# INFODIDACT'2020

## 13. Informatika Szakmódszertani Konferencia


## Előadáskötet

2020

Szerkesztette: Dr. Szlávi Péter, Dr. Zsakó László
Megjelenés: 2020. december



**ISBN** 978-615-80608-4-4

# Tartalom

# A távolléti oktatás tapasztalatai

Bakonyi Viktória[1], Szabó Dávid[2], Korom Szilárd[3], Dr. Illés Zoltán[4]

{[1]hbv,[2]sasasoft,[3]szicsa,[4]illes}@inf.elte.hu
ELTE IK

**Absztrakt.** A 2020 tavaszán bekövetkező járványhelyzet miatt néhány nap alatt az iskoláknak át kellett állni távolléti oktatásra úgy, hogy sem a diákok, sem pedig a tanárok nagy része nem rendelkeztek korábbról ilyen jellegű tapfasztalattal. Ráadásul az otthoni eszközellátottság is nagyon változó volt - akár egy-egy osztályon belül is - így a helyzet megoldása nagyfokú rugalmasságot kívánt mindenkitől. A tanárok rendkívül leleményesen megpróbálták felhasználni az általuk elérhető összes lehetőséget pl. Facebook csoportokat hoztak létre, e-mailt küldtek, online kvízeket, videókat gyártottak és online órákat tartottak. Az iskolai év vége felé felmérést készítettünk a gyakorló informatikai tanárok körében, hogy milyen tapasztalatokat szereztek, milyen problémákkal és esetleges előnyökkel szembesültek. Cikkünkben ennek a felmérésnek az eredményét értékeljük.

**Kulcsszavak:** közoktatás, módszertan, digitális oktatás

## 1. Bevezetés

A XXI. században kiemelt fontosságú, hogy az emberek rendelkezzenek digitális kompetenciával. 2016-ban elfogadták a Magyarország Digitális Stratégiáját [1], amely megfogalmazta, hogy milyen lépéseket kell megtenni a célok eléréséhez. Az iskolákban kialakítandó technikai háttér mellett az oktatás módszertani megújításával is foglalkoztak a Digitális Pedagógia Módszertani Központ [2] iránymutatásával. A munka elkezdődött, de maradt jócskán teendő, hiszen ennyi idő alatt a teljes tanári társadalom módszertani és informatikai továbbképzése még nem valósult, nem valósulhatott meg.

Ilyen előzmények után kellett egyik napról a másikra átállni digitális távolléti oktatásra 2020 márciusában a közoktatás minden szintjén [3]. Azok a tanárok is, akik korábban is használták a modern technika nyújtotta lehetőségeket egy teljesen új helyzetben találták magukat. Nem egyszerűen arról volt szó, hogy az osztálytermi óráikat a digitális eszközök, lehetőségek segítségével tegyék érdekesebbé, hatékonyabbá, hanem magát az óratartást kellett (volna) áthelyezni a virtuális térbe. Tovább nehezítette a dolgukat, hogy nem lehetett arra számítani, hogy minden diáknak megfelelő eszköze és informatikai tudása van az otthoni munkához, ugyanakkor számukra is biztosítani kellett a folyamatos oktatást. A tanárok nagy hányada megragadta az összes lehetséges eszközt, hogy elérhesse a rábízott diákokat a postai levelezéstől kezdve a virtuális osztálytermek használatáig. [4] Nem szabad elfelejteni azt sem, hogy milyen sok feladat hárult a szülőkre is, akik főleg a kisebbek esetén aktív szerepvállalásra kényszerültek az otthoni feladatok megoldásában.

A felmerülő nehézségek ellenére a digitális oktatás többségében sikeresen vizsgázott és az erőfeszítések eredményeképpen mind a két oldal, mind a tanárok, mind pedig a diákok új lehetőségekkel, új ismeretekkel gazdagodtak [5]. Ugyanakkor meg kell említeni, hogy a hátrányos helyzetű térségekben vagy tanulók esetében voltak problémák. [6]

Egyetemi oktatóként mi is megtapasztaltuk a digitális térben való oktatás specialitásait. A saját benyomásaink mellett természetesen kíváncsiak voltunk tanítványaink, kollégáink véleményére is. A velük készített interjúkat, illetve a tanítványaink összegyűjtött véleményét már feldolgoztuk korábban. [7] [8] Jelen cikkünkben a közoktatásban dolgozó informatika tanárok, tanári gyakorlatukat végző tanárszakos hallgatók véleményét dolgoztuk fel, akiknek az eszközök, új alkalmazások kezelése talán kevésbé, de a digitális oktatásban alkalmazható módszertan kialakítása nem volt egyszerű.

## 2. Adatgyűjtés

Készítettünk egy kérdőívet a Google segítségével, hogy biztosítsuk az anonimitást, „Fókuszban a digitális oktatás (2020)" címmel.

A kérdőív a következő linken található: https://bit.ly/30viDlW

A válaszolók létszáma: 64 (IK tanítási gyakorlatot folytató hallgató, levelező tanár, informatikai műveltségterületen tanítók és informatika tanárszakon végzett tanítványaink)

### 2.1. A kérdőív

A kérdőív kérdései magyarul:

1. Hol tanít? (Többszörös választás: Felsőoktatás / Középiskola / Felsőtagozat /Alsótagozat)
2. Milyen módon szervezte az óráit? Jelölje meg az összes formát, amit használt! (Többszörös választás: Streamelt óra / Előre felvett óra / Elektronikusan kitölthető feladatlap / Online kvíz / Postai levél /Egyéb)
3. Mennyi volt ebből az online, valós-idejű alkalom? (Választható: 1-5 1=egyik se, 5=mind)
4. Volt-e olyan diák, aki nem tudott bekapcsolódni az online órákba, ha egyáltalán tartott ilyet? (Választható: 1-5 1=senki, 5=mind)
5. Milyen alkalmazást, alkalmazásokat használt? (Szabadszöveges válasz)
6. Hogyan oldotta meg a számonkérést? (Szabadszöveges válasz)
7. A visszajelzések alapján a gyerekeknek melyik módszer tetszett a legjobban és a legkevésbé? (Szabad szöveges válasz)
8. Tapasztalatai szerint a személyes jelenlét közben vagy online módon voltak-e aktívabbak a tanítványai? (Választható: 1-5 1=hagyományos, 5=online)
9. Mivel próbálta a tanítványait aktivizálni az online órákon? (Szabadszöveges válasz)
10. A tananyag elsajátítása melyik módon sikeresebb? (Választható: 1-5 1=hagyományos, 5=digitális)
11. Voltak-e olyanok, akik az átlagnál jobban vagy rosszabbul teljesítettek, akár önmagukhoz, akár korábbi teljesítéseikhez képest? (Szabadszöveges válasz)
12. Mi lehet ennek az oka? (időgazdálkodás/szorgalom/családi háttér/közösség/egyéb)
13. Van-e olyan és ha igen, melyik a most kipróbált módszerek, lehetőségek közül, amelyet használna a későbbiekben is. (Szabadszöveges válasz)
14. Van-e egyéb észrevétele, mondanivalója a témával kapcsolatban? (Szabadszöveges válasz)

### 2.2. Mit mondanak az adatok?

A válaszolók egy része (16 fő) nemcsak egyetlen iskolában, iskolafajtában tanít lásd 1. táblázat.

| Fajta | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| Személy | 47 | 13 | 4 | 0 |

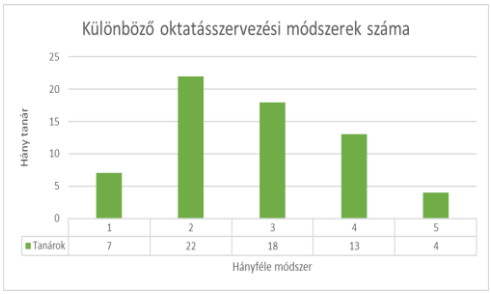
**1. táblázat:** Hány fajta iskolában tanít

Az egyes iskolafajtákban tanítók száma a válaszolók között a 2. táblázatnak megfelelően alakult:

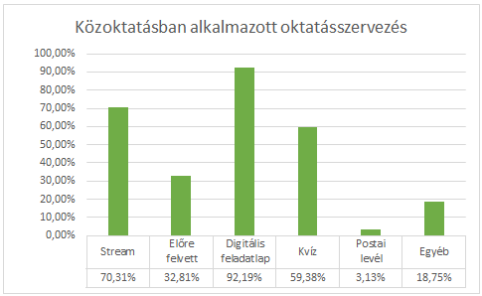
| Felsőoktatás | Középiskola | Általános iskola (felső) | Általános iskola (alsó) |
|---|---|---|---|
| 8 | 47 | 21 | 9 |

**2. táblázat**: Az egyes iskolafajtákban tanítók száma

Az iskolák különböző elvárásai, a korosztályok, az otthoni eszközök ellátottsága mind, mind befolyásolta a választott oktatásszervezési módszereket, amelyet az 1. ábra mutat. (A felmérésben lehetséges választások: Streamelt óra / Előre felvett óra / Elektronikusan kitölthető feladatlap / Online kvíz / Postai levél /Egyéb) A 2. ábrán látható, hogy a tanárok 70%-a használta a streamelési lehetőséget, és 32 % előre felvett videókat is használt. Majd mindenki, 92% használt digitális feladatlapokat is a felkészüléshez, dolgozatíráshoz. A hagyományos levelezés mindössze 3%-ban fordult elő – egyetlen tanár küldött postán is feladatokat, miközben digitális feladatlapokat is küldött. Ez azt bizonyítja, hogy a felmérésben résztvevő tanárok esetében ténylegesen megvalósult a digitális oktatás.

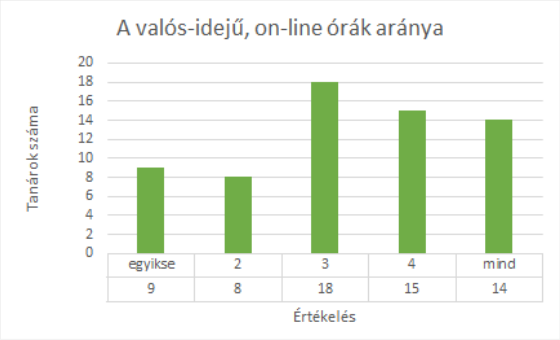



**1. ábra:** Módszerek száma




**2. ábra:** Az egyes módszereket alkalmazók aránya

Külön rákérdeztünk arra, hogy a valós-idejű virtuális osztályterem alkalmazásokat milyen számban használták a gyakorlatban, ami talán a legközelebb áll a hagyományos oktatáshoz. A 3. ábrán követhető, hogy 29-en (45%) majdnem mindig online órákat tudott tartani. 9 tanár volt, azaz 14 %, aki sohase streamelte az óráját.




**3. ábra:** Online órák

| Egyetem | Középiskola | Felső | Alsó |
|---|---|---|---|
| 8 / 8 (100%) | 33 / 47 (70%) | 12 / 21 (57 %) | 6 / 9 (66 %) |

**3. táblázat:** Az egyes iskolafajtákban tanítók stream / száma

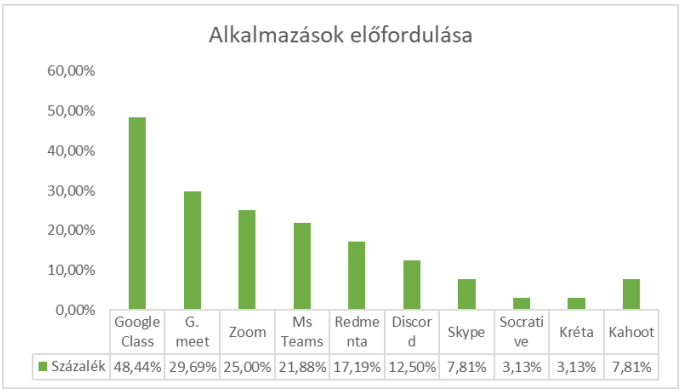
A különböző iskolafajták szerint is érdemes megfigyelni az online órák lehetőségét. Látható a 3. táblázaton, hogy mindenhol 55 % feletti a felhasználás.

A helyzetképhez hozzátartozik, hogy az iskolákban mennyi tanuló nem tudott bekapcsolódni az oktatásba. A 4. táblázat mutatja, hogy egy-egy osztályon belül általában akadtak olyanok, akik nem tudtak volna kapcsolódni az online órákhoz. Ez az eredmény azt az általános véleményt húzza alá, hogy a hátrányos helyzetű gyerekek esetenként kimaradtak néhány lehetőségből. (Egyes iskolákban egyáltalán nem tartottak online órát, így ezek nyilván a mind kategóriát választották.)

| Értékelés | senki | 2 | 3 | 4 | mind |
|---|---|---|---|---|---|
| Darabszám | 34 | 18 | 6 | 4 | 2 |

**4. táblázat:** A digitális lehetőségből kimaradók.

Az előzőekben felvázoltuk a digitális oktatás iskolatípusok szerinti módszereit. A következőkben azt vizsgáltuk, hogy mely szoftvereket alkalmaztak a tanárok a megvalósítás során lásd a 4. ábrát. Szabadszöveges válaszból gyűjtöttük ki a leggyakrabban előforduló alkalmazásokat. Látható, hogy a Google Class volt a leggyakrabban említett, de a Zoom és a Teams is 20% felett került említésre. Igen kevesen említették a Krétát. A Kahoot (7,8%) és a Socrative (3%) is szerepelt, ami azt mutatja, hogy a CRS (Classroom Response System) is kezd bekerülni a tanári gyakorlatba [9].




**4. ábra:** Az alkalmazások előfordulása

Az oktatás egyik fontos része a számonkérés, ami a hagyományos módon nem bonyolítható le a távolléti oktatási helyzetben. A következő kérdés arra vonatkozott, hogy hogyan folyt az értékelés. A szabadszöveges választ elemezve a következő jellemző lehetőségek kerültek elő: Redmenta használat, valamilyen LMS (Canvas, Moodle) használata, Skype, valamelyik CRS (Kahoot, Socrative), házi feladatok értékelése (egyéni/projekt), űrlapok, kvízek, órai munka, online feleltetés, feladatlapok visszaküldése, lefényképezése (videó készítés) lásd 5. táblázat.

| Feladatlap | Fénykép, videó | Redmenta | Űrlap | HF | Teszt | Kvíz |
|---|---|---|---|---|---|---|
| 25 | 18 | 17 | 16 | 10 | 10 | 9 |
| LMS (Canvas,Moodle) | Online órán felelés | Órai munka | CRS (Kahoot, Socrative) | Gondolkodtató feladatok | Skype | |
| 8 | 7 | 4 | 4 | 2 | 1 | |

**5. táblázat:** Számonkérési módszerek

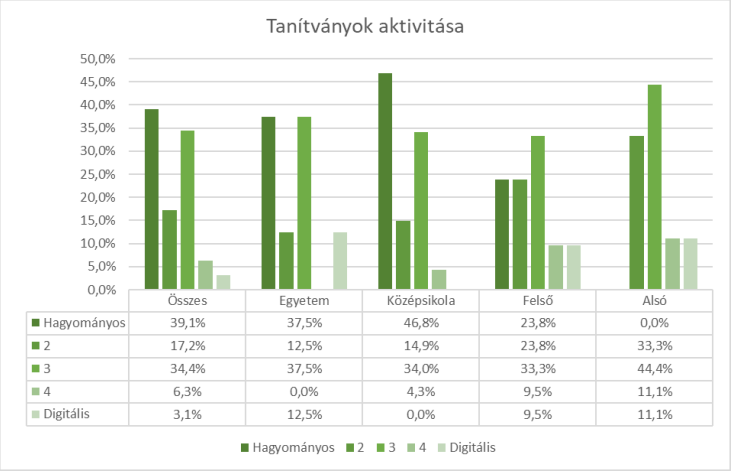
(A válaszokból nem derült ki minden esetben, hogy például a Moodle-t milyen módon használták: feladat kiadásra, beszedésre vagy tesztekre.) A leggyakrabban előforduló módszer a feladatlapok használata volt, amelyet változatos módon juttattak el a diákokhoz, e-mailben vagy valamely más

felületen keresztül pl. GoogleClassroom, Kréta. Sokan használták a papíron elkészített feladatok befényképezését, de voltak, akik egyes esetekben videó készítést is kértek, amiben a megoldást kellett elmagyarázni a diáknak. A harmadik leggyakoribb módszer a válaszoló tanárok körében a Redmenta volt. Jellemzően figyelembe vették az órai munkát, a szorgalmi feladatokat, beadandókat, de nem elhanyagolható számban online feleltetést is alkalmaztak. Rákérdeztünk arra is, hogy a módszereik közül véleményük szerint mi állt a legközelebb a diákokhoz. Sokan, sokfélét válaszoltak az online kvízektől, a tanár által készített videókon át a páros és projektmunkáig jelöltek kedvenceket, de a legjellemzőbb mégis az online óra említése volt: 23 esetben (35 %). Korosztályokra bontva a következőket kapjuk a 6. táblázatban:

| Egyetem | Középiskola | Felső | Alsó |
|---|---|---|---|
| **3 / 8 (37 %)** | **0 /47 (0 %)** | **19 / 21 (90%)** | **4 / 9 (44 %)** |

**6. táblázat:** Online óra a kedvenc

Saját tapasztalataink alapján is észrevehető volt, hogy a diákság aktivitása különbözött a hagyományos formákhoz képest. Egyfelől a gyerekek számára is új volt ez a fajta tanítási környezet, másfelől a mai fiatalok a mindennapokban egyre inkább a beszéd helyett a szöveg alapú kommunikációt részesítik előnyben, így egy ilyen jellemzően digitális környezetben talán inkább hallgatnak [10], amit az oktató passzivitásnak érzékelhet. Az 5. ábra mutatja, hogy több mint fele (25 + 11 ember ) inkább a hagyományos oktatásban tartja nagyobbnak az aktivitást, 22 ember szerint nincs jelentős különbség és csak 6 ember gondolja intenzívebbnek az online órákon a gyerekek szerepvállalási hajlandóságát.




| | Összes | Egyetem | Középsikola | Felső | Alsó |
|---|---|---|---|---|---|
| Hagyományos | 39,1% | 37,5% | 46,8% | 23,8% | 0,0% |
| 2 | 17,2% | 12,5% | 14,9% | 23,8% | 33,3% |
| 3 | 34,4% | 37,5% | 34,0% | 33,3% | 44,4% |
| 4 | 6,3% | 0,0% | 4,3% | 9,5% | 11,1% |
| Digitális | 3,1% | 12,5% | 0,0% | 9,5% | 11,1% |

**5. ábra:** Aktivitás tanári értékelés

Természetszerűleg adódik a kérdés, hogy milyen módszereket alkalmaztak ahhoz, hogy ösztönözzék a tanulóikat a közös munkára. A 6. ábrán azt mutatjuk meg, hogy milyen módszereket alkalmaztak a tanóráikon. 56 %-ban a személyes figyelmet említették a tanárok, mint a leginkább célravezető eszközt, de szerepelt a többször említettek között a dicséret, a jutalom és az érdekes feladatok kitűzése is.


**6. ábra:** Motivációs eszközök

Végezetül ejtsünk szót arról, hogy a digitális oktatásra való áttérés igen nagy erőfeszítést jelentett mindenki számára, hiszen a napi munka mellett sokaknak új alkalmazásokkal kellett megismerkedniük, illetve kidolgozni ezekben az egyes anyagrészeket. Ennek ellenére szinte mindenki (93 %) úgy ítélte meg, hogy a kényszerű átállás és plusz munka eredményeképpen létrejött anyagait, az új alkalmazásokat vagy egy részüket a normál osztálytermi oktatásban is használni fogja. 48,4 %-uk valamelyik virtuális osztálytermi alkalmazást is tovább használná. Összesen 33-an írtak valamilyen észrevételt. Vegyes volt a reakció, de szinte kivétel nélkül látták a digitalizáció előnyeit is, miközben hiányolták a személyes jelenlétet. Talán az egyikük által írt mondat adja vissza leginkább a véleményüket: „*a személyes jelenlét, kommunikáció pótolhatatlan a tanításban, közös élmények kellenek*”.

## 3. Összefoglaló

Kérdőívet állítottunk össze annak kiderítésére, hogyan élték meg az informatika tanárok a tavaszi osztálytermen kívüli távolléti oktatást. Vitathatatlan, hogy a digitális oktatásra való hirtelen átállás komoly erőfeszítést kívánt a tanároktól, de a befektetett munka eredményeképpen nagy lépést tettünk meg az oktatás digitalizációja irányában. Megismertek rengeteg alkalmazást, amely hagyományos keretek között is változatosabbá és hatékonyabbá teszik a tanítási módszereiket. A válaszokból az is kitűnt, hogy ezeket a kényszerhelyzet szülte új lehetőségeket a többség fel kívánja használni a későbbiekben is. Természetesen voltak menet közben problémák, például az aktivitás és a személytelenség tekintetében, de ezek talán csak a kezdeti nehézségek voltak, amit finom hangolni lehet a későbbiekben, ha szükség lenne rá.

## Irodalom

1. Digitális Oktatási Stratégia
https://digitalisjoletprogram.hu/hu/tartalom/dos-magyarorszag-digitalis-oktatasi-strategiaja (utoljára megtekintve: 2020.10.31.)
2. Digitális Pedagógiai Módszertani Központ
https://dpmk.hu/ (utoljára megtekintve: 2020.10.31.)
3. Digitális munkarend
https://koronavirus.gov.hu/cikkek/emmi-uj-munkarend-koznevelesi-es-szakkepzesi-intezmenyekben
(utoljára megtekintve: 2020.10.31.)
4. A digitális oktatás első tapasztalatai
https://www.mixonline.hu/Cikk.aspx?id=174633 (utoljára megtekintve: 2020.10.31.)
5. Sikeres volt a digitális oktatás
https://roviden.hu/2020/07/09/maruzsa-zoltan-bejelentette-hogy-sikeres-volt-a-digitalis-oktatas (utoljára megtekintve: 2020.10.31.)

6. A hátrányos helyzetű gyerekek más módon kapcsolódnak a digitális oktatáshoz https://eduline.hu/kozoktatas/20200428_hatranyos_helyzetu_gyerekek_digitalis_tanrend (utoljára megtekintve: 2020.10.31.)
7. Bakonyi Viktória, Illés Zoltán: *Real-time and Digital Solutions in Education During Emergency Situation in Hungary*, In: Abonyi-Tóth, Andor; Stoffa, Veronika; Zsakó, László New Methods and Technologies in Education, Research and Practice, Budapest, Magyarország : ELTE Informatikai Kar, (2020) pp. 231-240. , 10 p.
8. Bakonyi Viktória, Illés Zoltán, Chaman Verma: *Real-time Education in Emergency Situation*, elbírálás alatt 2020 IEEE First International Conference on Advances in Electrical, Computing, Communications and Sustainable Technologies (IEEE ICAECT 2020)
9. H., Bakonyi Viktória; Illés, Zoltán: Valós idejű oktatási rendszerek, In: Szlávi, Péter; Zsakó, László (szerk.) InfoDidact 2018, Budapest, Magyarország: Webdidaktika Alapítvány, (2018d) pp. 51-58., 8 p.
10. Larry Alton: Phone Calls, Texts Or Email? Here's How Millennials Prefer To Communicate, Forbes, May 11, 2017, 08:00am EDT, elérhető: https://bit.ly/2Ygdtt2 utolsó elérés dátuma: 2020. 10. 31.

# Algoritmusoktatás online oktatási rendszerben

Bende Imre

beiraai@inf.elte.hu
Eötvös Loránd Tudományegyetem, Informatika Kar

**Absztrakt.** A jelenlegi helyzetben egyre fontosabb, hogy készen álljunk az online oktatásra, meglegyenek az előzetes feltételek annak sikerességéhez. Az alapvető algoritmusok (programozási tételek) tanítása, megértése nagyon fontos alapköve a programozásoktatásnak, enélkül a továbbhaladás komplexebb algoritmusokra, feladatok megoldása sem lehetséges. Első lépésben a tanárok, diákok elé álló problémákat tárnám fel, majd pedig ezek megoldásához szeretnék irányt adni.

**Kulcsszavak:** online oktatás, programozásoktatás, algoritmus, algoritmus vizualizáció

## 1. Bevezetés

A jelenlegi helyzetben egyre fontosabb, hogy készen álljunk az online oktatásra, meglegyenek az előzetes feltételek annak sikerességéhez. Az alapvető algoritmusok (programozási tételek) tanítása, megértése nagyon fontos alapköve a programozásoktatásnak, enélkül a továbbhaladás komplexebb algoritmusokra, feladatok megoldása sem lehetséges. Első lépésben a tanárok, diákok elé álló problémákat tárnám fel, majd pedig ezek megoldásához szeretnék irányt adni.

## 2. Főbb módszerek offline oktatás során

Normál oktatási körülmények között különböző segédeszközök állnak rendelkezésünkre az alapvető utasítások, algoritmusok megértéséhez. Mindenképpen fontos megemlíteni hiszen a legfontosabb a tanári irányítás: a tanári magyarázat különböző illusztrációkkal (gondolok itt egyszerűbb példákra is mint az osztálytermi táblán való algoritmus bemutatás), gép melletti segítség a feladatok megoldásához, javításához. Fiatalabb diákok esetén megjelenhetnek különböző robotok, amelyek előre programozható utasításokat tudnak végrehajtani. Később előjöhetnek különböző játékos feladatok, melyek megoldásához az algoritmikus szemléletet kell felhasználni, ezek lehetnek online feladatok, valamint offline típusúak is (ezek akár kisebb átalakítás után az online oktatás során ugyancsak fel lehet használni). A feladatok bonyolultsága/komplexitása széles skálán mozoghat, így azok, a megfelelő eszköz kiválasztásával könnyen skálázhatóak.

## 3. Az online oktatás előnyei és hátrányai

Az online oktatás a 21.században jelent meg és lett egyre elterjedtebb, ahogy egyre több ember számára vált elérhető az Internet. Manapság egyre népszerűbb ez a tanulási forma, mivel így bárki, bárhol a világon létrehozhat, részt vehet ezeken. „Az online oktatási módszer nagyon hatékony alternatív oktatási módszer lehet az érett, önfegyelmezett, motivált, jól szervezett és magas fokú időgazdálkodási képességekkel rendelkező hallgatók számára, azonban ez nem megfelelő tanulási környezet a kevésbé önálló tanulók számára, és azoknak akiknek nehézségeik vannak az online tanfolyamok által megkövetelt felelősség vállalásában.”[4] A következőkben pontokba szedve nézem meg, hogy mikre kell odafigyelni az online oktatás során, milyen előnyei és hátrányai vannak.

## 3.1. Előnyei

- A tanóra és azon megjelenő eszközök mindenki számára elérhetőek az óra után is. Így, ha valaki lemaradt, nem tudott részt venni az órán, akkor is vissza tudja azt nézni a későbbiekben ugyanolyan feltételek mellett.
- Segíti az önálló tanulás kialakulását, fejlesztését, amely fontos szerepet tud játszani a továbbtanulásban, valamint az élet több területén is kifejezetten fontos kompetencia. Kisebb korban ez még nehezebben alakul ki, így náluk jobban oda kell figyelni, több időt kell felhasználni a közös tanulásra.
- Fejleszti a diákok egyéb online oktatáshoz szükséges digitális kompetenciáit, amit normál körülmények között nem kéne használni.
- A diákoknak nem kell utazniuk az iskolába, így kipihentebben tudnak már az első (reggeli) órán is részt venni, több idejük marad, amit tanulás mellett más „szabadidős" tevékenységgel is ki tudnak tölteni. Emiatt a tanulás folyamata is hatékonyabb lesz. Ez persze a másik fél részére is igaz, a tanároknak is több idejük jut az órákra való felkészülésre, pihenésre.
- A platformból adódóan több lehetőségünk van a diákokkal való közös kommunikációra, amely segíti a tanár munkáját és a diákok tanulását is.

## 3.2. Hátrányai

- Online oktatáshoz szükséges előfeltételek hiánya: Az online oktatáshoz mindkét félnek szüksége van egy számítógépre, webkamerára, mikrofonra, viszonylag magas sávszélességre (le- és a feltöltés egyaránt megjelenik a kétoldalú beszélgetés során). Ezek azonban nem mindenki számára elérhetőek.
- Oktatást segítő eszközök hiánya: Tábla, robotok, különböző segédeszközök nem állnak rendelkezésre. Ha vannak is használható eszközök, akkor is ezek drágák, beszerzésük nehézkes.
- Platformi nehézségesek: A gyakran használt online platformok, habár egy előadás bemutatását lehetővé teszik, de nem tudják visszaadni az osztálytermi környezetet, ahol lehetőség van külön egy-egy diákhoz odamenni, segíteni neki; egyszerre nem tudunk több eszközt használni/bemutatni.
- Kommunikációs hiányosságok: Nincs lehetőség az élőbeszédre. Sok esetben a diákoknak nincsen webkamerájuk/mikrofonjuk/videóbeszélgetésre alkalmas sávszélességük. Ez megakadályozza a rendes oktatás menetét, valamint probléma esetén nehézzé teszi a tanári segítségnyújtást. Ehhez kapcsolódó probléma, hogy a diákok egy része nem szívesen kommunikál ilyen formában, ami miatt nem csak a tanároktól, hanem a diáktársaitól is eltávolodhat.
- Tervezési feladatok: Nincsenek bevált módszerek, nincsenek előzetes tervezetek arra vonatkozólag, hogy online oktatás során, hogyan lehetne/lenne érdemes oktatni. Ezt időközben kell kitalálni a tanárnak, hogy a saját osztályával/csoportjával, hogyan is tud haladni, milyen eszközökkel tudja ezt megtenni. Ez gyakran rengeteg pluszmunkát jelenthet.
- Számonkérés mikéntjei: Hasonlóan az előző ponthoz, a tanárnak kell eldönteni, hogy mely számonkérési formát választja ki, amelyhez minden előfeltétel rendelkezésre áll és mindkét félnek megfelelő visszajelzést ad.

# 4. Online oktatás

## 4.1. Eszközök

Online oktatás során az alapvető eszközök hiánya megnehezíti az oktatás menetét, a megszokott segédeszközök nem állnak rendelkezésre. Ehhez köthető probléma, hogy ha találunk hasznos pót-eszközöket, akkor is drágák, valamint az iskolai beszerzés körülményes, túl sok időt vesz igénybe. Az ezek által nyújtott funkciók jelentős részét azonban imitálhatjuk online megoldásokkal, néha pedig egyszerűbb, olcsóbb megoldások is rendelkezésünkre állnak, amelyek akár eddig is a birtokunkban lehettek.

Sok esetben az osztálytermi táblát használjuk algoritmusok felírására, példák vizuális megjelenítésére. Otthoni környezetben erre azonban nincs lehetőség. Ha van is egy nagyobb felület otthonunkban, amit erre a célra tudnánk használni, akkor is egy olyan webkamera kéne annak rögzítésére, ami jó minőségű képet tud arról adni. Ezután pedig még előjön a sávszélesség kérdése is: a tanárnak folyamatos nagy feltöltési sebességre van szüksége, a diákok részéről pedig nagyobb letöltési sebességre, amin követni tudják a táblán felírtakat. Ennek egy egyszerűbb megoldása lehet, ha a tanár egy papírra írja le a dolgokat, amit telefonján lévő kamerával közvetít. A mai mobiltelefonok ehhez a célhoz már megfelelő kamerával rendelkeznek, valamint a közvetítendő tárgy kisebb, közelebb van a lencséhez, így nem is kell akkora képfelbontás a célhoz, mint egy rendes tábla közvetítéséhez. Ebben az esetben már csak arra van szükség, hogy a kamera képét megosszuk a számítógépünkkel. Egy másik irányzat lehet, hogy ha egy tabletet, vagy ehhez hasonló eszközt használunk, amelyen keresztül ábrázolunk. Ez nem csak abban segít, hogy könnyebben olvasható, látható lesz az átküldött "táblakép", hanem ezek tárolhatóak is, így a diákok bármikor vissza is nézhetik azokat.

Sajnos a normál körülmények közt használt eszközök nem állnak rendelkezésre, így további lehetőségünk még az online szoftverek, weboldalak használata, valamint előjöhetnek olyan offline játékok, amelyekhez mindenki egyszerűen és gyorsan létre tudja hozni a szükséges kellékeket.

## 4.2. Kommunikáció

„Az online kommunikáció célja megegyezik az élő kommunikáció céljával: kapcsolat, információcsere, meghallgatás, megértés." [3] Online formában azonban a kommunikáció nehézkes. Folyamatos mindenki által használt webkamera nélkül nincsen szemkontaktus, nem látjuk a nonverbális jeleket. Ha lenne is, akkor is inkább egyoldalú lenne ez, hiszen a tanár a kijelzőn nem tudja megjeleníteni, figyelemmel kísérni az összes diákot egyszerre, valamint a platformot/felhasználók internet sebességét is jelentősen leterhelné. Ez könnyen elidegenítheti az oktatás folyamatától a benne szereplőket, így erre mindenképpen különösebb figyelmet kell fordítani.

Tanácsok, amivel fejleszteni, személyesebbé tudjuk tenni az online kommunikációnkat a diákokkal:

- Használjunk megszólítást! Ez személyessé teszi az üzenetet, kérdésre adott választ, amely ezáltal a diáknak is értékesebb lesz.
- Ha lehetséges az órán kívüli üzeneteket is audió formában töltsük fel. Így a kapcsolat az online világ ellenére megvan, és a tanár hangja erősíti a kapcsolatot a diákokkal. (ez persze sokban függ a csoport összetételétől és a korosztálytól)
- Legyen egy felület, amelyen keresztül naprakészen tudjuk tartani a diákok eredményeit! Ez nem csak a dolgozatok eredményeire vonatkoznak, hanem az órai aktivitásukra is. Példának okáért az ELTE-n Webprogramozás kurzuson nyilvántartunk egy táblázatot, amelyen a diákok már az óra elején láthatják, mely önálló feladatokról lesz szó aznap, ha pedig készen vannak eggyel, akkor azt megjelölik (akár részmegoldást is). Ezáltal az oktató látja, hogy mely feladatok megoldását lehet ellenőrizni, melyekkel van probléma, a diákok pedig visszamenőleg is láthatják saját, valamint

társaik előrehaladását. Ennek célja nem a verseny kialakítása, hanem csak egy visszajelzés mindkét fél számára. Ez teljes mértékben önbevalláson alapszik és semmiféle értékelésben nem szerepel.

- Igyekezzünk minél több esetben kommunikációra, interakcióra bíztatni a diákokat! Ez lehet akár írásban, akár szóban is. Enélkül könnyen egyoldalúvá válhat a folyamat, ami nemcsak a diákok teljesítményét befolyásolhatja negatívan, hanem a tanár számára is nehezebbé teheti az oktatás folyamatát.
- Fontos, hogy úgy építsük ki az osztály és a tanár közti csatornát, hogy az mindkét fél számára ideális legyen. Így érdemes a diákokat is megkérdezni arról, hogy mely módszer volt számukra működő, vagy mely módszeren lehetne módosítani. Az oktatás kooperáció a tanár és a diákok között, így fontos, hogy minden szereplő hatékonyan tudja ellátni feladatait.

## 4.3. Oktatás menete

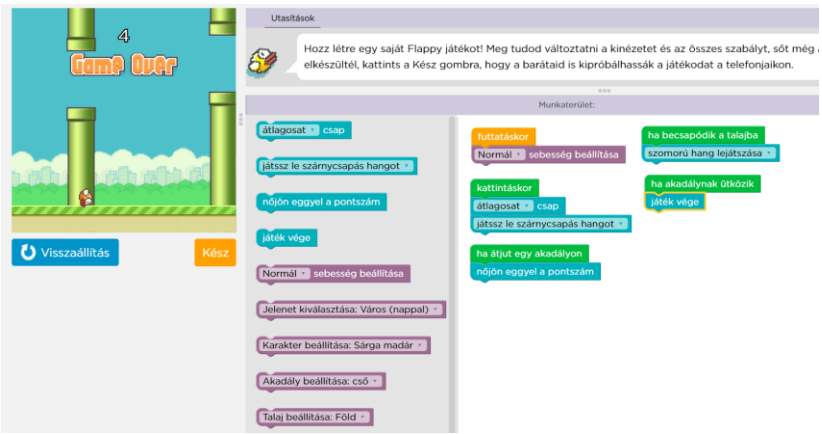
Természetesen, ha a megfelelő eszközök mind a diákok, mind a tanár részéről rendelkezésre állnak, akkor az oktatás mehet a szokásos módon, annyi különbséggel, hogy az oktatás nem szemtőlszembe megy, hanem webkamerán és monitoron keresztül. Azonban ez nagyon sok esetben nem tud megvalósulni: nincs megfelelő internetkapcsolat a folyamatos videókapcsolathoz, oktatást segítő eszközök, kommunikációs eszközök hiánya miatt. Ezért olyan eszközöket érdemes keresnünk az algoritmusoktatáshoz, mely nem feltétlenül igényli a folyamatos kommentárt (ez erősíti az önálló tanulást, valamint, ha valaki nem tud becsatlakozni egy órába, akkor sem marad le) és teljes mértékben be tudja mutatni az aktuális témát, algoritmust. Különböző korosztályoknál más követelményeknek kell megfelelni egy ilyen eszköznek, a következőkben ezt fogom áttekinteni.

### 4.3.1. Algoritmus vizualizációk, játékok

Általános iskolában még nem tanulnak a diákok programozási tételeket (legalábbis nem jellemző a fogalom használata, holott az egyszerűbbekkel már valószínűleg találkoztak, használták őket), számukra a fontos az alapvető utasítások, változók, struktúrák megértése és azok használata. A legjobb mód erre, ha a diák ezt valamilyen játékos formában sajátítja el, ahol az utasításokat ő rakosgatja össze és az eszköz látható formában futtattja le azt, majd jeleníti meg a végeredményt. Ebben az esetben megjelenhetnek a már eddig is használt szoftverek (itt érdemes figyelni arra, hogy a szoftverek ingyenes legyenek és ne legyen szükségük nagy erőforrásra a használatukkor, vagy pedig akár böngészőben is lehessen használni azokat telepítés nélkül): Logo, Scratch[1], amelyekben egyszerűbb utasításalapú játékokat, feladatokat lehet megoldani. Ehhez hasonló elven alapuló Blockly alkalmazások is szóba jöhetnek, melyek hasonló elven működnek, mint a Scratch és több feladatgyűjtemény is rendelkezésre áll[2].

[1] A Scratch online verziója: https://scratch.mit.edu/

[2] Például: https://code.org/, https://hourofcode.com/, https://blockly.games/




**1. ábra:** Flappy Bird imitáló játék készítése egyszerűbb utasításokkal[3]

Általános iskola végén, középiskolában már megjelennek a programozási tételek, rendezések, sőt versenyeken elég magas szintű algoritmusok is megjelenhetnek (gráfalgoritmusok, mohó algoritmus, backtrack …). A diákoknak már van kellő ismereteik a programozásról, így ezekre tudunk építkezni. Egy bemutató eszköz választásánál fontos, hogy az jól be tudja mutatni az egyes algoritmusokat (szövegesen leírással, valamint egy általános algoritmusleíró nyelven), vizuális elemekkel tegye érthetővé az algoritmus menetét, valamint rendelkezzen implementációval a diákok által tanult programozási nyelven. Gyakorlásnál pedig megjelenhetnek azok a weboldalak, programok, melyen keresztül játékos környezetben gyakorolhatják programozási készségeiket (melyek gamifikációs elemeket tartalmaznak, vagy akár vizuális megjelenítéssel segítik a feladatok megoldását). Ilyenek például a: CodinGame[4], Code Combat[5], Codewars[6], CodeChef[7].

Ezek mellett megjelenhetnek az offline játékok is, amik nem szükségeltetnek egyéb eszközök meglétét, viszont kicsit kiemeli a diákokat a gép mögül, úgy, hogy mindeközben az algoritmikus szemléletüket fejlesztik. A Computer Science Unplugged [1] pont ezt a szemlélet követi, amely rengeteg hasznos és érdekes feladattal, majd ezek alapján újabb lehetőségekkel, irányzatokkal látják el a tanárokat. Több esetben a diákok könnyen elkészíthetik a szükséges eszközöket hozzájuk, melyek például a kézügyességüket is fejlesztik. Példának okáért egy rendezéses algoritmus elsajátításához nincs másra szükség, mint egy mérlegre és pár hasonló méretű, de különböző súlyú tárgyra. Ez alapján aztán a diákok megpróbálhatják azokat rendezni, majd pedig a bemutatott elvek mentén el tudják sajátítani a rendezések működését. Mindezt úgy, hogy számítógépre egyáltalán nem volt szükség! Kisebbeknél hasznos tud lenni az ilyen, hiszen a folyamatos online órák mellett néha szükséges, hogy „kimozdítsuk" őket a gép elől.

[3] https://studio.code.org/flappy/10

[4] https://www.codingame.com/

[5] https://codecombat.com/

[6] https://www.codewars.com/

[7] https://www.codechef.com/

### 4.3.2. Oktatási segédanyagok önálló tanuláshoz

Az eredeti oktatási formához visszatérve, de kikerülve a szükséges folyamatos internetkapcsolatot előjöhet a szöveges/videó tutorial fogalma is. A lényege az lenne, hogy minden témához létrehozunk egy rövid anyagot (csatolva hozzá a szükséges mellékleteket), így a diákok ezeket bármikor elolvashatják/megnézhetik és ezeken keresztül el tudják sajátítani az újabb készségeket. Az óra ez esetben az anyagok feldolgozásáról szólna, valamint a felmerülő kérdések írásos/szóbeli megválaszolásáról. Itt érdemes összedolgozni a kollégákkal, hiszen ez esetben az elkészült munkákat mindenki felhasználhatja és a jövőben is hasznos segédletet nyújthat. Dokumentum formájában érdemes rövidebb kulcsszavakban összefoglalni a fogalmakat, míg videóanyagban egy mintafeladat megoldását célszerű bemutatni lépésről lépésre. A segédanyagok végére érve érdemes elérhetővé tenni a diákok számára az említett, megoldott programokat is, hiszen, ha a diákok esetleg problémába ütköznek, elakadnak időközben, akkor azt később is elő tudják venni, így nem fognak lemaradni emiatt. Egy harmadik ehhez hasonló opció lehet még egy olyan dokumentum, felület (algoritmus vizualizációs eszköz) használata is mely egyszerre írja le egy algoritmus működését, mutatja be azt, valamint mintaprogrammal is rendelkezik. Több ilyen vizualizációs eszköz is létezik[8], amely ezeknek a kritériumoknak megfelel, ezeket akár egy az egyben is felhasználhatjuk, vagy kisebb kiegészítésekkel is bemutathatjuk, felhasználhatjuk.


**2. ábra:** Algoritmus vizualizációs eszköz leírással, mintakóddal, algoritmussal, valamint lépésenkénti vizualizációval[9]

A tanári segítség normál körülmények között úgy történik, hogy a tanár odamegy a diák gépéhez és segít megoldani a felmerülő hibát. Online környezetben ez azonban nem megvalósítható. A diák persze megoszthatja a képernyőjét, de ebben az esetben az mindenki számára láthatóvá válik, nagyobb sávszélességet igényel, valamint plusz időt vesz igénybe a képernyő megosztása. Erre lehet egy megoldás például a Visual Studio[10] Live Share[11] bővítménye. A diákok minden óra elején megosztják a munkafolyamatukat, a tanár pedig szükség esetén csak erre csatlakozik rá, így azt (megfelelő jogosultsággal elindított megosztás esetén) nem csak megtekinteni tudja, hanem akár szerkeszteni is tudja a szóbeli segítség mellett. Ez jóval kevesebb erőforrást igényel, hiszen nem folyamatos képi megosztást igényel, hanem csak a szöveges változtatásokat tölti fel a szerver fele. Akár a számonké-

[8] Például: https://visualgo.net/en, http://ermi46.web.elte.hu/dev/algotan/, http://anim.ide.sk/

[9] http://ermi46.web.elte.hu/dev/algotan/maximumkivalasztas.html

[10] https://visualstudio.microsoft.com/

[11] https://visualstudio.microsoft.com/services/live-share/

rés ellenőrzésében is segíthet, habár sok diák esetén nem lehet figyelemmel követni az összes diák munkáját.

## 4.4. Számonkérés

Személyes jelenlét hiányában nem tudjuk ellenőrizni, hogy a diákok ténylegesen, hogyan is oldják meg a számukra adott feladatokat. Adott esetben külső segítséget is használhatnak, amelyek miatt az eredmények nem tükrözik a tényleges tudásukat. Itt gondolok tárgyi segítségre (tankönyv, internetes segédanyagok, fórumok, szoftverek), valamint személyi segítségre is (osztálytársak megoszthatják a megoldásaikat, ismerős segíthet a feladat megoldásában). Ezek sajnos 100%-ban nem kerülhetőek el, de bizonyos mértékben tudunk rajtuk segíteni.

Az osztályon belüli másolás egyik ellenszere lehet, ha a diákok különböző feladatok kapnak. Ez persze többletmunkát igényel: közel azonos nehézségű feladatok kiválasztása, javítás hosszabb ideje. Erre megoldás lehet egy feladatbank, mely nem csak egy adott témában tartalmaz különböző feladatokat, hanem egy értékelővel is rendelkezik. Ez esetben a feladatok részletes leírása, minta be- és kimenetek már megvannak, valamint a tesztelő a funkcionális tesztelést, értékelést is elvégzi. Ilyen használható rendszer lehet például a mester[12], amely a versenyeken használt bíró értékelő rendszer publikus változata. Ez nem csak régebbi versenyfeladatokat, hanem egyszerűbb programozási tételre visszavezethető feladatokat is tartalmaz (többek között régebbi ELTE-s elsőéveseknek szánt beadandók és zárthelyik is szerepelnek a kínálatába). Az előző részben több weboldalt, alkalmazást is megemlítettem, amelyek különböző nehézségű, típusú feladatokat tartalmaznak. Ezeket számonkérés során is felhasználhatjuk, ezzel segítve a tanár feladatát, valamint a diákok számára is élvezhetőbb, ezáltal motiválóbb környezetben tudják bizonyítani tudásukat. Így akár olyan dolgokra is juthat időnk, mint a kódtisztaság vizsgálata, beszédes változónevek használatának figyelése, személyes visszajelzések.

Egy másik irányzat lehet tudásfelmérésre a beadandó írása, készítése prezentálással, bemutatással. A diákok egy általuk választott témát (algoritmust) próbálnak bemutatni egy vizualizáció segítségével diáktársaiknak. A diák hozzájuk közel álló platformot/témát választhatnak, amellyel szívesebben foglalkoznának. Ez nem csak a tudásokat ellenőrizné, hanem mélyebb tudást is szereznek az adott témakörben, valamint egyéb kompetenciáikat is fejlesztik vele (bemutató készítése, prezentációs készség, valamint választott platformtól függően egyéb készségeiket is fejlesztik a diákok). A vizualizáció felhasználhat offline és online elemeket is, mely lehet egy egyszerűbb prezentáció képek sorozatából, de akár egy összetettebb program, weboldal készítése is. A cél, hogy az elkészített munka segítségével mélyebben megértse az algoritmus működését, fejlessze az előbb említett kompetenciáit, és a többiek tanulását is segíti. Karavirta által összegyűjtött kutatásokból [2] pedig látszik, hogy minél magasabb az interakciós szint ([5] által meghatározott szintek: nincs megtekintés, megtekintés, válaszadás, változtatás, összerakás, bemutatás) az algoritmus vizualizáció használatánál, annál inkább lesz hatékony a tanulás folyamata.

[12] http://mester.inf.elte.hu/

# 5. Összegzés

Az online oktatásnak megvannak az előnyei és hátrányai is. A sikeres tanulási folyamathoz szükség van megfelelő mennyiségű felkészüléshez mind eszköz, mind tervezés szintjén. Emellett viszont újabb kapuk nyílnak meg, amiket érdemes kihasználnunk. Kényszerhelyzetben sajnos hirtelen történik a változás, amiből igyekszünk kihozni a legtöbbet úgy, hogy az mindenki számára megfelelő legyen. A cikkem során igyekeztem segédkezet nyújtani, irányokat bemutatni arra vonatkozólag, hogy tudjuk az algoritmusok oktatását, majd annak számonkérését az online környezetben a lehető legjobban megvalósítani, melyeket példákkal is illusztráltam. Kiemelten foglalkoztam az algoritmus vizualizációs eszközök használatáról, hiszen ebben a szituációban az a legkönnyebben felhasználható, hatékony oktatási eszköz, módszer (habár normál körülmények között is erősen ajánlott a használatuk, hiszen rengeteget segít az algoritmusok megértésében). A tudásátadás mellett azonban fontos az is, hogy ezt hogyan tesszük meg, milyen eszközöket használunk, milyen platformokat használunk, hogyan kommunikálunk. Így ezekre is hangsúlyt fektettem, melyek általánosságban segíthetik az online oktatásra való áttérést, valamint annak sikerességét. A feltételek mindenki számára mások, azonban a megfelelő lépésekkel egy jó környezetet tudunk kialakítani, amelyben mindkét fél jól érzi magát, aminek hatására a diákok is motiváltabbak lesznek és a fejlődésüket is segítik.

## Irodalom

1. Mike Fellows et al.: *Computer Science Unplugged*, Computer Science Unplugged, 2015.
2. Ville Karavirta: *Facilitating Algoritm Vizualization Creation and Adoption in Education*. Helsinki University of Technology, 2009.
3. Whitney Kilgore: *Humanizing Online Teaching and Learning*. Whitney Kilgore, 2016.
4. Dhirenda Kumar: *Pros and Cons of Online Education*. North Carolina State University, 2015.
5. Thomas L. Naps, G. Rössling, V. Almstrum, W. Dann, R. Fleischer, C. Hundhausen, A. Korhonen, L. Malmi, M. McNally, S. Rodger, J. Á. Velázquez-Iturbide: *Exploring the role of visualization and engagement in computer science education*. ITiCSE on Innovation and Technology in Computer Science Education (ITiCSE'02). ACM Press, pp131–152, 2002.

# Lányok az informatikában – Módszerek és esettanulmányok a nők eredményesebb bevonására

Bernát Péter[1], Szlávi Anna[2]

[1]bernatp@inf.elte.hu, [2]dr.szlavi@gmail.com
ELTE Eötvös Loránd Tudományegyetem Informatikai Kar

**Absztrakt.** Nemzetközi felmérések szerint a természettudományos egyetemi szakokon a hallgatóknak csupán 30-40%-a lány, Magyarországon pedig mindössze 23%-a. Az informatikai képzések nemi aránya még ennél is jobban el van tolódva hazánkban: programtervező informatikus szakon például 15%-hoz közelít a női hallgatók aránya. A hazai és nemzetközi kutatások arra mutatnak rá, hogy elsősorban a nemi sztereotípiák – és az ebből adódó kellemetlen közoktatási élmények – az okai annak, hogy kevés lány választja az informatikus szakmát. Cikkünk első részében olyan módszereket tekintünk át, amelyekkel felülírható a „nem tudom" és a „nem akarom" élmény, és ezáltal a lányok számára is elérhetőbbé és élvezetesebbé tehető az informatika. A második részben pedig két lehetséges módszernek, az interaktív prezentációnak és a játékfejlesztésnek a 2020-as Lányok Napján megvalósult gyakorlati alkalmazását mutatjuk be részletesen esettanulmányunk keretein belül.

**Kulcsszavak:** informatikaoktatás, gender, példaképek, játékfejlesztés, motiváció

## 1. Bevezetés

A 2019-20-as koronavírus-járvány, amely az egész világot megbénította, egyértelműen rámutatott, hogy a tudomány és a technológia nemcsak a jövő, hanem a jelen legfontosabb befektetése. Normális életvitelünk szigorú keretek közé szorult, így digitális lehetőségek nélkül a legtöbb munkafolyamat és szociális kapcsolat lehetetlenné vált volna. Visszatérésünk a „normális kerékvágásba" a tudomány és a technológia kezében van. 2020-ban tehát különösen lényeges kérdéssé vált, hogy a STEM[1] területeken maximalizáltuk-e kapacitásunkat.

Az informatika területén – már a pandémia előtti kimutatások szerint is – munkaerőhiány van. Ezzel összefüggésbe hozható, hogy a kutatások szerint a nők jelenléte nagyon alacsony, mindössze 16,5% az EU28 zónában. [1] Ez a hiány már az oktatásban is kimutatható. Az UNESCO átfogó felmérései szerint a természettudományos egyetemi szakokon a hallgatóknak csupán 30-40%-a lány [2], Magyarországon pedig mindössze 23%-a [2].

A hazai és nemzetközi kutatások arra mutatnak rá, hogy elsősorban a nemi sztereotípiák – és az ebből adódó kellemetlen közoktatási élmények – az okai annak, hogy kevés lány választja az informatikus szakmát [3, 4, 5]. Cikkünk célja, hogy alternatív megközelítést javasoljon közoktatásban oktató tanároknak arra, hogyan vonják be eredményesebben lány diákjaikat. A cikk első felében olyan módszereket tekintünk át, amelyekkel felülírható a „nem tudom" és a „nem akarom" élmény, és ezáltal a lányok számára is elérhetőbbé és élvezetesebbé tehető az informatika. A cikk második felében pedig, egy esettanulmány keretein belül, két lehetséges módszernek, az interaktív prezentáci-

[1] STEM = Science (természettudomány), Technology (technológia), Engineering (mérnök-tudomány), Mathematics (matematika)

ónak és a játékfejlesztésnek a 2020-as Lányok Napján megvalósult gyakorlati alkalmazását mutatjuk be részletesen.

## 2. Nemi arányok az informatikában

Az utóbbi 50 évben egyre több tudományterület fordította figyelmét a nemi viszonyok vizsgálatára, köztük például a filozófia [6], a pszichológia [7], a nyelvészet [8], a szociológia [9] és a pedagógia. [10] Az elmúlt 20 évben azonban egyre több tanulmány születik a STEM-en belül is ebben a témában [11, 12, 13, 14], hisz a természettudományok is felismerték a tudományterületen belüli nemi alapú egyenlőtlenségeket és azt az igényt, hogy ez változzon. A nők hiánya ugyanis nemcsak társadalmi és gazdasági hátránnyal jár, hanem szakmai-technológiai következményei is vannak. [15]

Nemi alapú egyenlőtlenségeket találunk az informatika munkaerőpiacán, a felsőoktatásban és a közoktatásban is. Ami az informatikai munkaerőpiacot illeti, Kirkup átfogó vizsgálata arra mutatott rá, hogy mind vertikális, mind horizontális munkamegosztást találunk a szektoron belül. [16] Amellett, hogy kevés a nő ezen a pályán, aki mégis bejut, az tipikusan alacsonyabb fizetést kap és lassabban jut előre, mint férfi kollégái. A pozíciók típusát illetően is különbségeket talált: a nők tipikusan kevésbé vannak jelen technikai-műszaki állásokban; inkább a kevesebb szaktudást igénylő pozíciók nyitottak előttük.

Az egyik fő oka annak, hogy az informatikai állásoknak miért csak 16,5%-át töltik be nők, az, hogy már az egyetemi informatikai képzéseken is jóval kisebb arányban vannak jelen a nők, mint a férfiak. Ramirez és Kwak kutatása rávilágít, hogy míg általában az egyetemre beiratkozottak többsége nő, a természettudományos szakok esetében már egyértelmű kisebbségben vannak a lányok. [2] Egyes országokban, köztük Magyarországon, mindössze 23% az arányuk; sőt a vizsgálat azt is megmutatta, hogy némileg csökkenő tendencia figyelhető meg országunkban a lányok természettudományos pályaválasztási hajlandóságát illetően.

2012-ben az egyik magyar egyetem úgy döntött, hogy utánajár, miért ilyen alacsony a STEM területek iránt érdeklődők száma a magyar középiskolás lányok között [4, 5]. A kutatás, amely kérdőívek és interjúk formájában zajlott, mind a diáklányokat, mind a tanárokat megkérdezte a STEM területekről alkotott véleményükről és azok kompatibilitásáról lányokkal. A vizsgálat legtöbb konklúziója az volt, hogy a lányok nem érzik úgy, hogy „lányoknak valók" ezek a szakmák és nem is érzik érdekesnek a területet – főként a középiskolai természettudományos órákon szerzett tapasztalatok alapján. A lányok hozzátették, ennek fő oka, hogy úgy érezték, másképp bántak velük a tanárok, mint a fiúkkal, amelyet néhány, meginterjúvolt tanár maga is alátámasztott.

## 3. Megközelítések a nemi arányok javítására

Ha társadalmi, gazdasági és technikai előnye is lenne annak, ha több nő választaná az informatikát, akkor minél nagyobb erőkkel és minél hatékonyabban kell ezen dolgoznunk. A szakirodalom három különböző megközelítésről beszél. Bonder az alapján különbözteti meg az eddigi megközelítéseket, amelyek a nők bevonását célozták meg, hogy mire fókuszálnak. [17]

Az első megközelítés a számokra összpontosít, vagyis, hogy kevés a nő az egyetemi szakokon, munkahelyeken, tehát ezt kell orvosolni („fix the numbers"). Éppen ezért a beiratkozási vagy felvételi számokat próbálják javítani ezen megközelítés képviselői. Azzal azonban nem számolnak, hogy mennyire fenntartható csupán a bemenetre fókuszálni. Sok esetben az intézmény maga nem tudja a megfelelő feltételeket biztosítani ahhoz, hogy lányok és nők számára hosszútávú megoldás lenne a felvételük.

A második megközelítés már ezt is figyelembe veszi és az intézményeket állítja górcső alá („fix the institutions"). Például a túlnyomó többségű férfi oktatógárdát, ami elidegenítőként hathat a frissen felvett női hallgatóra, női mentortanárokkal lehet ellensúlyozni. Vagy baba-mama szobák

létrehozásával a kismamák számára is lehetővé teszik, hogy ne kelljen karrier és család között választani. De továbbra is kérdéses, hogy ez a megközelítés mennyire képes az egyenlőtlenségek alapjait megváltoztatni.

A harmadik megközelítés azt állítja, hogy csak ha az alapokra – az egyenlőtlenségeket legelementárisabb szinten okozó társadalmi sztereotípiákra – fókuszálunk, akkor lesz hosszú távú és fenntartható a változás („fix the knowledge"). Ha felülírjuk az olyan társadalmi berögződéseket, hogy a lányok nem értenek a tudományokhoz, nem gondolkodnak logikusan, nem lehetnek önálló pénzkeresők, egyedüli felelősségük a család, akkor válik lehetségessé, hogy egyenlő esélyei legyenek az iskolapadban, majd a munkahelyen. Vagyis, az oktatásnak központi szerepe van abban, hogy a lányok számára elérhetőbbé váljon az informatika és általában a tudomány.

# 4. Módszerek az informatikaoktatásban

Mint a fentebbi kutatások rámutattak, a lányok és nők elsősorban a – tudományt a férfi(asság)hoz kapcsoló – nemi sztereotípiák miatt maradnak távol a STEM területektől. A sztereotípiák mind a tanárok viselkedését, mind a diákok magukról és egymásról alkotott képét befolyásolják. Az, hogy a lányok nagy része nem nyitott az informatikára, nem elsősorban képességbeli alkalmatlanságról szól. Sok esetben két pillére van ennek a passzivitásnak: az a benyomás, hogy az informatika nem érdekes a számukra, valamint az az érzés, hogy lányokként ők erre nem képesek. Vagyis, a „nem akarom" és a „nem tudom" szubjektív berögződése áll a kimaradásuk mögött. Tanárként ennek a két meggyőződésnek a megváltoztatásán kell dolgoznunk.

Az alkalmazható módszereket két csoportra oszthatjuk aszerint, hogy a tanulók elsősorban befogadóként vagy alkotóként vesznek-e bennük részt.

## 4.1. Bemutatók

A módszerek első csoportjába tartoznak a hagyományos vagy interaktív – kvízkérdésekkel vagy más rövid játékos feladatokkal tarkított – előadások (például Kahoot kvízek és interaktív Kahoot prezentációk), valamint a filmvetítések és az intézménylátogatások, amelyeket beszélgetések vagy játékos feladatok követhetnek. Ezek a módszerek, amelyek elsődleges célja, hogy női példaképekkel és sikersztorikkal ismertessék meg a lányokat, azért szükségesek és hatékonyak, mert az informatikáról és a női informatikus karrierről alkotott képüket árnyalja („tudom") és az érdeklődésüket felkelti („akarom"). Megmutatják a résztvevőknek, hogy számos nő képes volt sikereket elérni az informatika területén, amivel a nemi sztereotípia okozta berögződést képesek feloldani, hogy pusztán azért, mert lányok, nekik nem való ez a terület.

## 4.2. Foglalkozások

A második csoportba tartoznak azok a rendszeres vagy alkalmi foglalkozások, amelyek keretén belül oktató segítségével a lányok ki is próbálhatják magukat a programozásban, és a pozitív benyomásokon túl („akarom") sikerélményeket is szerezhetnek („tudom"). Ezek a kedvező hatások akkor várhatók, ha az oktató a lányokat (is) mozgósítani képes programozási feladatokat tűz ki, amelynek a feltételeiről lentebb, programozási területeként írunk.

### 4.2.1. A foglalkozások színterei

Ideális esetben a lányokat is bevonni képes foglalkozásoknak a meghatározó színterei az iskolai informatika órák. A 2020-as Nemzeti Alaptantervhez illeszkedő kerettantervekben a korábbiakhoz képest jelentősen emelkedtek a programozásra (valamint a programozói gondolkodást megalapozó algoritmizálásra) fordítható óraszámok: a 3-4. osztályban 8-ról 16-ra, az 5-8. osztályban 20-ról 48-ra, a 9-10. osztályban pedig 10-ről 25-re. Emellett az általános iskolai programozásoktatásban tartalmi

bővítésre is sor került: már nemcsak a teknőcgrafika és a robotika, hanem az animációkészítés és a játékfejlesztés is előírt programozási területek [18].

A lehetséges *rendszeres* foglalkozások közé tartoznak még az iskolai programozószakkörök, amelyeket az iskola informatika tanárai vagy meghívott oktatók is tarthatnak, valamint a piaci alapon működő intézmények programozókurzusai.

A lányok informatika iránti érdeklődését felkelteni szándékozó *alkalmi* foglalkozásokat elsősorban tudománynépszerűsítő rendezvényeken, illetve cégek és alapítványok szervezésében tartanak. Az előbbire hazai példa a Lányok Napja, az utóbbira pedig a Technológiai Oktatásért Alapítvány Skool nevű projektje, amely különböző technológiai cégekkel együttműködve szervez néhány órás vagy akár egész napos programokat, nemcsak fiatal lányok, hanem hátrányos helyzetű gyermekek számára is [19].

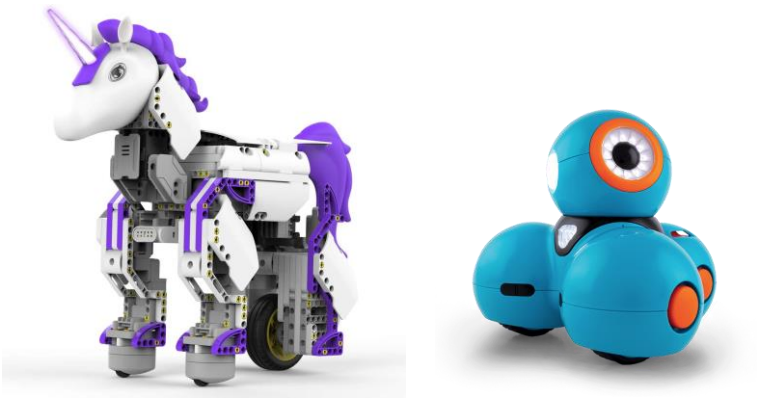
### 4.2.2. A foglalkozások tartalma

A bevezető programozástanítás során olyan oktatási célú programozási nyelvek használata a jellemző, amelyekkel a hétköznapokból is ismerős utasításokkal programozhatók könnyen megszemélyesíthető objektumok: például robotok, amelyek adott feladatok elvégzésére taníthatók be, vagy a képernyőn megjelenő szereplők, amelyek megfelelő vezérlésével animációk vagy grafikus játékprogramok is készíthetők. Mivel a robotika, az animációkészítés és a játékfejlesztés középpontjában is objektumok állnak, ezért közérthető módon vezethetők be velük az objektumorientált programozás alapjai, amely napjainkban a szoftveripar egyik legnépszerűbb programozási paradigmája [20].

**Robotika**

Az oktatási célú programozható robotok általában egyenesen haladni és kanyarodni képes szerkezetek, amelyekhez érzékelők is tartozhatnak. Az érzékelő nélküli változatok csak a külső hatásoktól függetlenül képesek működni, az érzékelőkkel rendelkezők viszont reagálhatnak a környezet bizonyos változásaira. Néhány fajtájuk készre szerelt és felépítésük nem módosítható, más típusok azonban a rendelkezésre álló elemekből mindig az aktuális problémának megfelelően építhetők össze.

A robotok szó szerint kézzel foghatók és kapcsolatba kerülhetnek a valós világ tárgyaival, ezért „életre keltésük" bizonyos tanulók számára a képernyőn futó programok írásánál jóval motiválóbbak lehetnek [21].



**1. ábra:** A lányoknak szánt UnicornBot és a mindenkinek készült Dash robot

A gyártók egy része kifejezetten a lányok számára is előállít robotokat – ezeket a reklámjaikban lányok használják –, amelyek között van például katicabogár [22] vagy unikornis alakú [23] (1. ábra, bal oldal). Más cégek inkább nemi sztereotípiáktól mentes robotokkal kívánják a gyermekek minél szélesebb körét bevonni a programozásba. Így például a Wonder Workshop, amelynek Dash nevű

robotja [24] először kevéssé érdekelte a lányokat, mert sokukat autóra vagy teherautóra emlékeztette, miután azonban a végső változatában elrejtették a kerekeit, már osztatlan sikert aratott a gyermekek körében [25] (1. ábra, jobb oldal).

Master, Cheryan, Moscatelli és Meltzoff megállapították, hogy már az első osztályos amerikai tanulók körében is mérsékeltebb a lányok érdeklődése és önbizalma a robotikában és a programozásban, amely különbség azonban a kutatók által szervezett robotprogramozás kurzuson való részvételt követően eltűnt [26]. Természetesen idősebb lányok is motiválhatók ezzel a programozási területtel – például Screpanti és társai kutatásában 11-13 éves lányok váltak érdeklődőbbé az informatika iránt a kéthetes foglalkozás végére [27].

**Animációkészítés**

Két- vagy akár háromdimenziós animációk számos oktatási célú környezetben létrehozhatók, amelyekhez hátterekre és az előttük mozgó szereplőkre van szükség (2. ábra). A hátterek és a szereplők kiválaszthatók a környezetbe épített galériából, de sok esetben meg is rajzolhatók. Programozásra a szereplők vezérléséhez és a jelenetváltásokhoz van szükség.

Szinte bármilyen tantárgyi vagy akár hétköznapi témakörben készíthetők történetek, szemléltetések vagy bemutatók, ezért az animációkészítés különösen alkalmas arra, hogy a nem informatikai érdeklődésű tanulók érdeklődését is felkeltse a programozás iránt [28].


**2. ábra:** Egy kétdimenziós animáció a ScratchJr
és egy háromdimenziós az Alice programozási környezetben

Kelleher és Pausch kutatásukban sikeresen motiváltak felső tagozatos lányokat programozásra animációkészítéssel. Megállapításaik szerint a történetmesélő animációk programozása azért ösztönző a lányok számára, mert lehetőséget ad saját történeteik elmesélésén keresztül az önkifejezésre, különböző társadalmi szerepek és emberi viszonyok végiggondolására, és nem utolsó sorban szert tenni a programozásban akár nem jártas kortársak és felnőttek elismerésére is [29].

**Játékfejlesztés**

A grafikus játékprogramokra tekinthetünk úgy, mint interakciókkal kiegészített animációkra: ezek is két- vagy háromdimenziósak lehetnek (3. ábra), a létrehozásukhoz pedig szintén hátterekre és szereplőkre van szükség, működésük során azonban folyamatos interaktív kapcsolatban van az egy vagy több játékos a játékprogrammal, valamint a belső szereplők egymással, amely a szoftverfejlesztésben szintén gyakran alkalmazott eseményvezérelt programozással valósítható meg.

A játékfejlesztés mindenekelőtt a játékok tipikus megvalósítási módszerei iránt érdeklődő tanulók számára lehet nagyon ösztönző. Az animációkészítéshez hasonlóan játékok is készíthetők sokféle tantárgyi, vagy akár hétköznapi témakörben, ám ehhez nagyobb kreativitásra van szükség [30].

**3. ábra:** Egy kétdimenziós játék a Scratch [31]
és egy háromdimenziós a Kodu [32] programozási környezetben

Az utóbbi néhány évben végzett felmérések alapján a számítógépes játékosok nagyjából fele nő, jelentősen eltér azonban a nemek aránya az egyes műfajokon belül, amelyet érdemes figyelembe vennünk, ha a játékfejlesztéssel a lányok motiválása a célunk. 2017-es felmérések szerint a nők körében az akció-kalandjátékok, a logikai játékok, a szerepjátékok és a platformjátékok a legnépszerűbbek [33, 34].

Ugyanakkor koedukált csoportokban az elkészítendő játékprogramok nemsemlegességére is törekedhetünk. A tanulók minél szélesebb körét megszólító játékprogramok erőszakmentesek, a játékosok logikus gondolkozását vagy ügyességét teszik próbára, inkább célorientáltak mint nyílt végűek, és ez a cél pozitív (például építeni kell valamit és nem lerombolni), valamint lehetőséget biztosítanak a társas interakciókra [35].

Carmichael kutatásában 8. és 9. osztályos lányok egy egyhetes játékfejlesztő minikurzus végén úgy nyilatkoztak, hogy nagyobb valószínűséggel fognak informatikai tanulmányokat folytatni a középiskolában [36].

## 5. Esettanulmány

A következőkben egy esettanulmányon keresztül a gyakorlatban is be szeretnénk mutatni néhány lehetséges alkalmazását a korábban ismertetett módszereknek, hogy eredményesebben be tudjuk vonni a lányokat is az informatikába.

### 5.1. Lányok Napja

A „Lányok Napja” elnevezésű program és kampány egy nemzetközi kezdeményezés, melynek célja a STEM területek népszerűsítése általános és középiskolás lányok körében [37]. 2012 óta minden évben megrendezi az eseményt a Nők a Tudományban Egyesület (NaTE), tipikusan áprilisban, de 2020-ban, a koronavírus-járványra való tekintettel, októberben és online került megrendezésre. Minden évben számos felsőoktatási intézmény, kutatóintézet és vállalat vesz részt az eseményen fogadóintézményként: idén körülbelül 60 intézménybe látogathattak el a lányok virtuálisan. A résztvevő lányok száma folyamatosan nőtt az évek során, az utóbbi években körülbelül 200 lány vett részt a programon[2].

[2] Jóllehet az idei, online megrendezésű eseményen egy kicsit kevesebben voltak.

Az esemény honlapján részletes statisztikákat olvashatunk az utóbbi 4 év programjairól [38], kezdve a résztvevő lányok, iskolák és intézmények számáról, egészen a fókuszban levő szakmákig. Sokatmondó adat például, hogy az elmúlt 9 év során 12.500 résztvevő inspirálódhatott a programokon. Ahogy – szintén az éves adatokat feldolgozó – infografikákon olvashatjuk, mind a lányok, mind a (tipikusan női) szakemberek hasznosnak találták az együttdolgozás lehetőségét.

## 5.2. Lányok Napja 2020 az ELTE IK-n

Az Eötvös Loránd Tudományegyetem Informatikai Kara évek óta részt vesz a Lányok Napja kezdeményezésen. A 2020-as Lányok Napjára – online jellegéből adódóan – egy félnapos programmal készült, az elején egy, az egyetemet és a kart bemutató prezentációval, valamint két rövidebb előadással egy-egy női oktatóval. Ezt követően elkezdődtek a játékos, interaktív programok, amelyek kifejezetten a fentebb bemutatott módszertan alapján lettek összeállítva. A félnapos eseményen 40-en vettek részt.

A következő két alfejezet ezeket a programokat mutatja be részletesen, majd a végén röviden kitér a diákok visszajelzéseire. Az első program egy, kvízkérdéseket is tartalmazó, interaktív prezentáció volt inspiráló tech-nőkről, míg a második egy játékprogramozó foglalkozás volt.

### 5.2.1. Interaktív Kahoot prezentáció

Ahogy említettük, a lányok idegenkedése az informatikától a társadalmi nemi sztereotípiákból fakad, amelyek azt sugallják nekik, hogy a lányok passzívak, nem értenek a tudományhoz, és nem is nekik való. Éppen ezért ahhoz, hogy hatékonyabban népszerűsítsük az informatikát a lányok körében is, először is lényeges, hogy ténylegesen bevonjuk őket, vagyis aktív és interaktív feladatokat találjunk ki. Másodszor pedig fontos, hogy ahhoz az élményhez juttassuk őket, hogy (lányként is) képesek rá és hogy (lányként is) érdemes ezzel foglalkozniuk, vagyis a „nem tudom" és a „nem akarom" berögződést próbáljuk felülírni.

Példaképek – vagyis női sikertörténetek – megismerése egy lehetséges módja annak, hogy elhiggyék, ők is képesek lehetnek az informatikában sikereket elérni. [39, 40, 41] Minél jobban bevonja a diákokat az esemény, annál maradandóbb tud lenni az élmény. A hagyományos előadásformánál interaktívabb, így hatékonyabb egy kvíz-szerű megközelítés, amelyben a diákok egyrészt maguk „rakják össze" az információkat, ahelyett, hogy passzívan befogadnák, másrészt pedig egymással versenyezve haladnak előre, ahelyett, hogy tét nélkül hallgatnák az előadást.

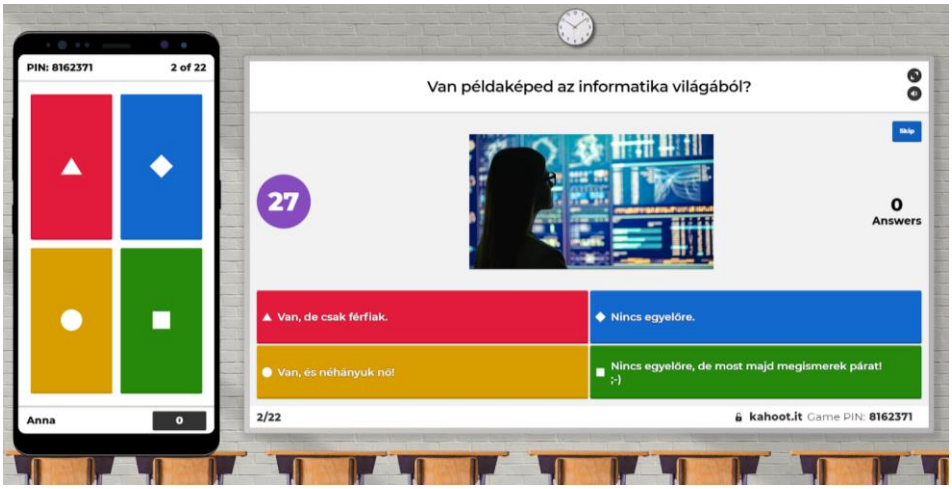
Az egyik lehetséges megvalósítása ennek a Kahoot. A Kahoot [42] egy ingyenesen elérhető, játék alapú tanuló platform, amely böngészőben és okostelefonos applikáción keresztül is elérhető. A platform alapötlete a gamifikáció, vagyis, hogy játékos megközelítésen keresztül hatékonyabb a tanulás és az információcsere. A Kahoot játékszerű elemei a verseny (amelyet az időközi részeredmények megjelenítése, majd a végső pódium segít elő), a vidám, laza hangulat (amelyet a vicces, automatikusan generált versenyző-nevek, valamint a színes és zenés felhasználói felület biztosít), valamint a modern, korosztályra jellemző technológia használata (okostelefon és okostábla/laptop/táblagép).

A 2013 óta működő platformnak több mint 1,2 milliárd felhasználója van; a 2019. év végén kitörő koronavírus járvány és az ennek hatására globális trenddé váló online oktatás miatt 23%-kal megnőtt a regisztrációk száma az előző évhez képest [43]. Tavasszal a Prémium szolgáltatások is ingyenessé váltak pedagógusok és oktatók számára az online oktatás idejére[3]. Manapság a magyar diákság jórésze ismeri (használta már) a Kahootot, főleg a kvíz változatát, hisz használata egyszerű. Ahhoz,

[3] A gyakorlatban ez 2020. október elejét jelentette, hisz a 2020/2021-es tanév őszi félévétől már nem kevésbé volt jellemző az online vagy hibrid oktatás a közoktatásban, mint a 2019/2020-as tanév tavaszán.

hogy egy Kahoot játékban részt vegyen valaki, regisztrációra sincs szükség, mindössze egy okostelefonra vagy egy laptopra, amelyen az adott játékhoz generált kódot beírja (4. ábra), majd a játék majd a játékmoderátor által valamilyen felületen (kivetítővásznon, megosztott képernyőn vagy laptopkijelzőn) megjelenített kérdésére a saját telefonján vagy laptopján megnyomja a helyesnek gondolt válasz ikonját (5. ábra).


**4. ábra:** Egy Kahoot-játék kezdőfelülete
(bal oldalt: okostelefon a játékosnak, jobb oldalt: számítógép-kijelző a játék moderátornak)

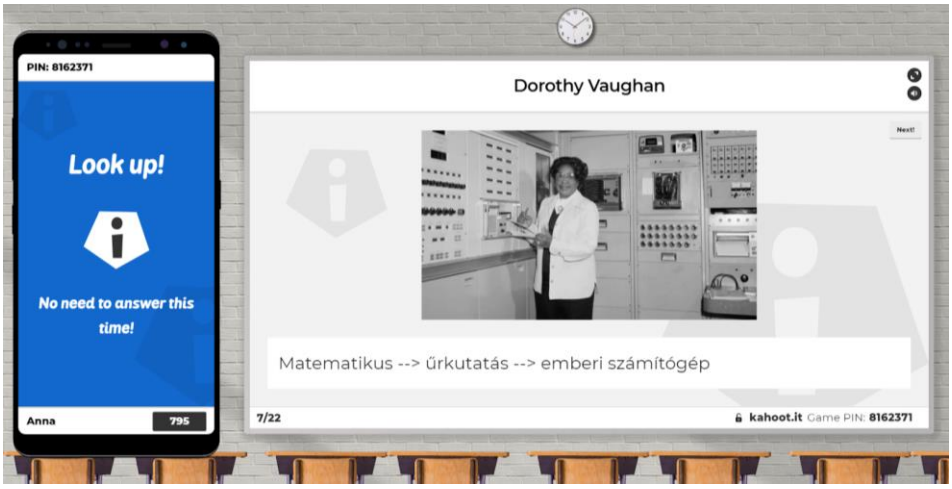



**5. ábra:** Egy Kahoot-játék játékos, illetve moderátori felülete
(bal oldalt: a játékos okostelefonja, jobb oldalt: a moderátor számítógép-kijelzője)

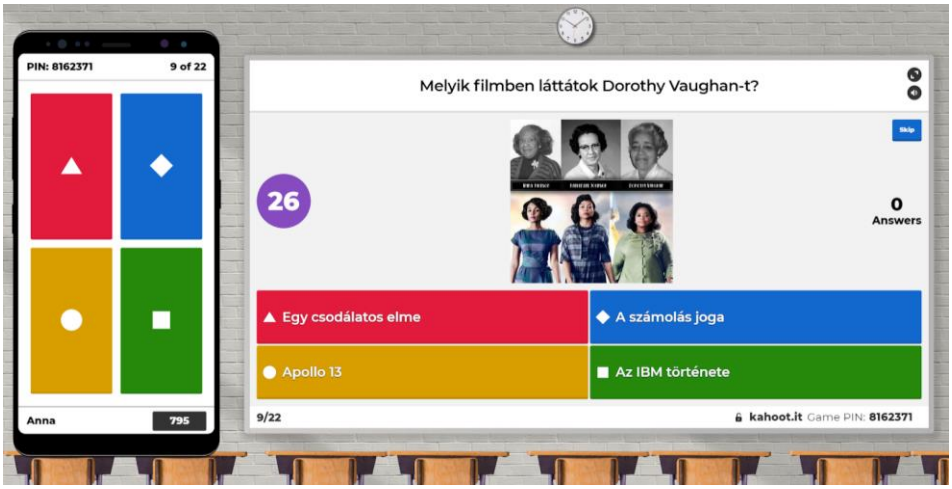
A Lányok Napjára összeállított játék egy „interactive presentation" (vagyis interaktív prezentáció)[4], amely egyesíti a Kahoot három fő funkcióját, a „quiz" (kvíz), a „poll" (szavazás) és a „slide"

[4] Az interaktív prezentáció jelenleg a Prémium szolgáltatáson belül elérhető.

(dia) funkciókat. A játék tervezett időtartama 20-25 perc volt, amelyhez 22 oldalt terveztünk, körülbelül fele-fele arányban elosztva az információ átadó (diás) oldalakat és az interaktív (kvízes vagy szavazós) oldalakat. A megfontolás az volt, hogy kapjanak új információt, mint egy hagyományos prezentációban, de maguknak is aktívan hozzá kelljen adniuk az információfeldolgozáshoz. Úgy lett tehát a prezentáció felépítve, hogy egy pár perces előadás (tehát néhány „dia") után mindig egy pár perces aktivitás (tehát „kvíz" vagy „szavazás") következzen (6-7. ábra). A kérdések vagy visszakérdeznek az anyagra, tehát megerősítik az előbb elmondottak legfontosabb tudnivalóját, vagy ahhoz hozzátesznek – a legtöbb esetben kitalálható – új információkat. Az ilyen formájú interaktivitás fenntartja a diákok figyelmét, illetve sikerélményt okoz, hisz maguktól jutnak el az információkhoz.




**6. ábra:** „Dia" egy Kahoot-játékban (balra: játékos kijelzője, jobbra: moderátor kijelzője)

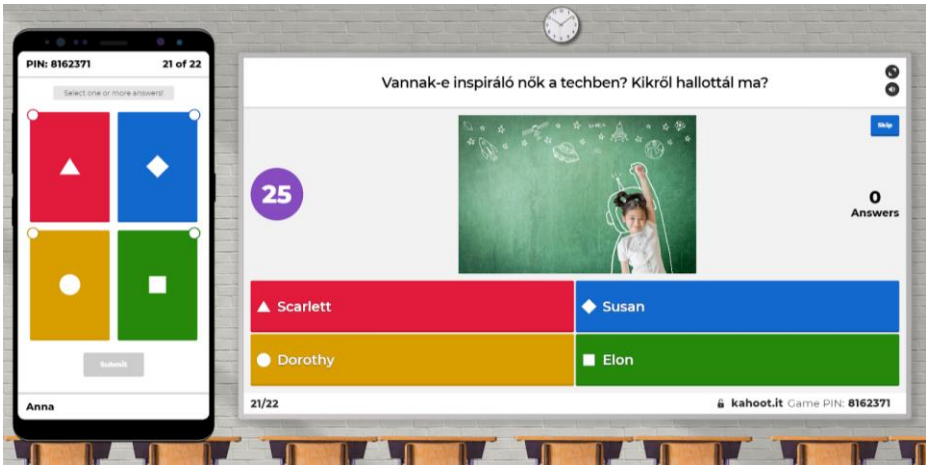



**7. ábra:** „Kvíz" egy Kahoot-játékban (balra: játékos kijelzője, jobbra: moderátor kijelzője)

A játék célja az volt, hogy ízelítőt adjon a lányoknak, hogy mennyi(féle) női sikertörténet van a tech világban. Az időkeretek miatt három terület egy-egy képviselőjét választottuk: a tudomány világából Dorothy Vaughant, akit a populáris médiából ismerhetnek a diákok; a tech ipar világából Susan Wojcickit, aki a YouTube, az egyik legnépszerűbb közösségi média platform, vezetőjeként

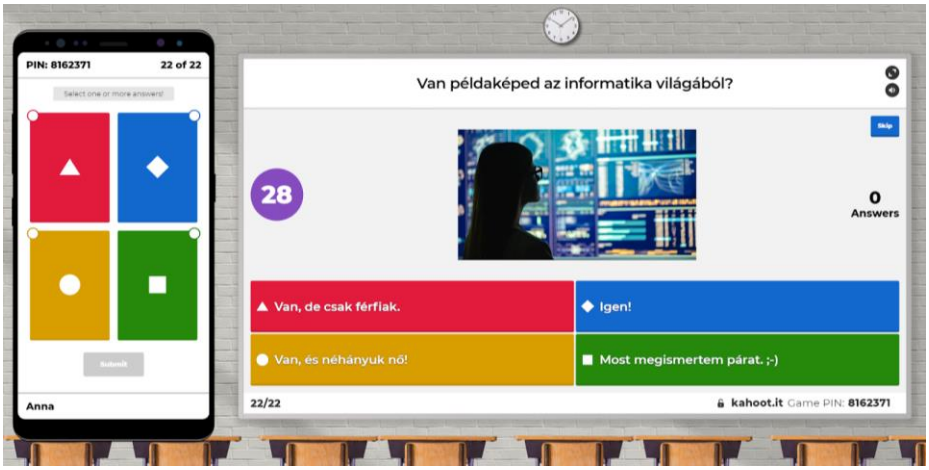
lehet releváns példakép; és a lányok körében is egyre népszerűbb gaming világából Sasha Hostynt, a legsikeresebb profi gamert.

A három sikertörténet interaktív bemutatását egy összegzés zárta le, amelynek funkciója az volt, hogy megismételje és megerősítse a prezentáció célkitűzését: annak a tudatosítását, hogy vannak sikeres nők a tech világában. Először egy dupla-pontot érő, összefoglaló kvíz-kérdés jött (8. ábra), majd a prezentáció elején is szereplő szavazás ismétlődött meg némileg módosított válaszlehetőségekkel (9. ábra).




**8. ábra:** Záró „kvíz”-kérdés

A prezentációt nyitó, majd záró, keretként szolgáló szavazás azt kívánta tudatosítani a lányokban, hogy ha eddig nem is volt olyan tech-nő a látókörükben, aki példaképként vagy inspirációként szolgálhatna, akkor most megismertek hármat. Az interaktív utazás, amelyben felépítették három különböző korú, etnikumú, hátterű és érdeklődésű nő történetét, aki képes volt érvényesülni, megadta a lányoknak azt az élményt, hogy az informatika egy olyan világ, amiben nekik is helyük van.




**9. ábra:** Záró „szavazás”-kérdés

### 5.2.2. Játékfejlesztés a Scratch-ben

A felső tagozatos és középiskolás lányoknak szánt egyórás gyakorlati foglalkozással informatikai karunk népszerűsítésén túl kiemelt célunk volt lehetőséget biztosítani a résztvevőknek arra, hogy saját pozitív tapasztalatokat szerezhessenek a programozásról. Habár az alkalmazott fejlesztői környezetben – más oktatási célú nyelvekhez hasonlóan – leegyszerűsített és játékos formában lehet programozni, a közös munka során az objektumorientált programozás számos kulcsfontosságú fogalmát ismerhették meg és alkalmazhatták a lányok, és ezáltal valós benyomásokat és sikerélményeket szerezhettek.

A résztvevőktől előzetes programozói tudást nem, csak számítógéphasználati alapismereteket vártunk el. Otthoni számítógépükre semmilyen alkalmazást nem kellett előre feltelepíteniük, ugyanis az alkalmazott Scratch programozási környezet internetes böngészőben futtatható a hivatalos honlapján [44]. Annak érdekében, hogy saját munkájukat a foglalkozás közben és végén menteni tudják, a lányokat előre megkértük, hogy regisztráljanak a Scratch-be.


**10. ábra:** A Fall Guys játék néhány pillanatfelvételét bemutató dia

A lehetséges programozási területek közül azért a játékfejlesztés mellett döntöttünk, mert (például a robotikával szemben) online formában is jól kivitelezhető, és mert így az interaktivitás nemcsak a foglalkozást, hanem az elkészült programot is jellemezhette. A Scratch mellett szólt, hogy ez napjaink egyik legismertebb ingyenesen használható oktatási célú programozási környezete, valamint hogy megtalálhatók benne a játékfejlesztéshez szükséges programozási nyelvi elemek. A 2020 augusztusában megjelent Fall Guysra pedig a hirtelen szerzett hatalmas népszerűsége mellett azért is esett a választásunk, mert rendelkezik a tanulók minél szélesebb körét – így a lányokat is – bevonni képes játékoknak a cikk módszertani részében említett ismérveivel: erőszakmentes, a játékosok ügyességét teszi próbára, a cél világos és pozitív színezetű, és többen is játszhatják [35].

**11. ábra:** A Fall Guys játéknak a foglalkozás során elkészített egyszerűsített változata, amely a prezentációban képernyővideón jelent meg

A foglalkozás bevezetésében egy háromdiás prezentációt használtunk, amelynek a kezdődiáján felidéztük a résztvevőkkel közösen a Fall Guys-t (10. ábra). A második dián bemutattuk egy képernyővideón a játéknak a foglalkozás során elkészítendő egyszerűsített változatát (11. ábra). A harmadik dián pedig egy-egy példával rávilágítottunk arra, hogy a szoftveripar a játékfejlesztésen belül, és azon kívül is előszeretettel alkalmazza az objektumorientált programozást (12. ábra).


**12. ábra:** Annak az illusztrációja egy dián, hogy az objektumorientált programozást a szoftveriparban a játékfejlesztésen belül (bal oldal) és azon kívül is alkalmazzák (jobb oldal)

Az eredeti háromdimenziós játékban az online játékosok cukorka alakú szereplők bőrébe bújva mérhetik össze ügyességüket az élénk színű kipárnázott akadálypályákon (10. ábra). A lányok ennek egy egyedül vagy ketten játszható offline felülnézeti változatát készíthették el a Scratch-ben. A 11. ábrán látható pályán a játékosok zöld, illetve piros figurája a bal alsó sarokból indul, és anélkül kell nekik a másikat megelőzve eljutniuk a jobb felső sarokban található célba, hogy beleesnének a

pályát körülvevő rózsaszín tengerbe, vagy hozzáérnének a mozgó akadályokhoz – mindkét esetben ugyanis azonnal visszakerülnek a startvonalhoz.

A játék grafikai elemeit előkészítettük, és néhány akadály mozgását előre leprogramoztuk, és mindezt egy kiindulási változatban egy linken keresztül kapták meg a résztvevők. A program lényegi részét azonban ők valósították meg a saját számítógépükön úgy, hogy a szükséges lépéseket folyamatosan követték a megosztott képernyőnkön. Vagyis a segítségünkkel ők valósították meg (ebben a sorrendben), hogy

1. a püspöklila akadályok a tengelyeik körül folyamatosan forogjanak,
2. a két játékos a saját figuráját képes legyen a billentyűzet megfelelő gombjaival irányítani,
3. a rózsaszín tengerhez vagy a mozgó akadályokhoz érve a szereplők azonnal visszakerüljenek a startvonalhoz,
4. és hogy amint az egyik játékos szereplője áthalad a célvonalon, véget érjen a játék.

A közös programozás során több alkalommal a résztvevő lányok nevezték meg a következő logikus lépést, az utolsó percekben pedig számos továbbfejlesztési ötletet is bedobtak: például, hogy a játékosoknak a célba érés előtt a pálya különböző pontjain elhelyezett tárgyakat is össze kelljen szedniük, vagy hogy a több pályán keresztül gyűjtött pontszámok alapján alakuljon ki a verseny végeredménye.

A lányok aktív részvételén túl az interaktív prezentáció és a játékprogramozó foglalkozás sikerességét mutatja az is, hogy utólag többen megfogalmazták a NaTE felmérésében, hogy a kar programjai közül ezek tetszettek nekik a legjobban.

## 6. Befejezés

Cikkünk célja az volt, hogy felhívja a figyelmet a nők és a lányok alacsony számára az informatika területén, valamint, hogy rámutasson, hogyan lehetne ezen a gazdasági, társadalmi és technológiai hátrányokkal járó helyzeten változtatni. Kutatásunk szerint a közoktatásnak és az oktatóknak központi szerepe van ebben a folyamatban, így cikkünk ehhez kíván útmutatót nyújtani. Először olyan módszereket vonultattunk fel, amelyek alkalmasak lehetnek arra, hogy hatékonyabban meg tudjuk szólítani a (közoktatásban tanuló) lányokat és az informatika világára fogékonyabbá tegyük őket. A cikk második felében pedig a 2020-as Lányok Napja esemény két programját, egy interaktív prezentációt és egy játékkészítő foglalkozást mutattunk be részletesen. Célunk ezzel az volt, hogy demonstráljuk, hogyan lehet a gyakorlatban is alkalmazni a fentebb felsorolt módszereket, amelyek – a résztvevő lányok utólagos visszajelzése alapján – élvezetessé és elérhetőbbé tették számukra az informatikát.

## Irodalom


1. Eurostat. (2019). *ICT specialists in employment.* https://ec.europa.eu/eurostat/statisticsexplained/index.php/ICT_specialists_in_employment#ICT_specialists_by_sex (utoljára megtekintve: 2020.11.15.)
2. Ramirez, F. O. & Kwak, N. (2015). *Women's Enrollments in STEM in Higher Education: Cross-National Trends, 1970–2010.* In Pearson, Jr. W., Frehill, L. M. & McNeely, C. L. (Szerk.) Advancing Women in Science. An International Perspective (pp. 9-26). Springer.
3. Nosek, B.A., Smyth F. L., Sriram, N., Lindner, N. M., Devos, T., Ayala, A., BarAnan, Y., Bergh, R., Cai, H., Gonsalkorale, K., Kesebir, S., Maliszewski, N., Neto, F., Olli, E., Park, J., Schnabel, K., Shiomura, K., Tulbure, B. T., Wiers, R. W., . . . Greenwald, A. G. (2009). *National differences in gender-science stereotypes predict national sex differences in science and math achievement.* PNAS June 30, 2009 106 (26) 10593-10597; https://doi.org/10.1073/pnas.0809921106

4. Papp, G. & Keszi, R. (2013). *A műszaki felsőoktatás a nemek tükrében – különbségek a pályaválasztás és az egyetemi tapasztalatok területén.* Zárótanulmány. In Szekeres V. & Krolify Intézet (szerk.) „Ti ezt tényleg komolyan gondoltátok?" Nők és a műszaki felsőoktatás. Óbudai Egyetem, pp.214–314
5. Nagy, B. (2014). *Háttérben: Kísérlet egy szervezeti nemi rend feltárására.* L'Harmattan.
6. de Beauvoir, S. (1969). *A második nem.* Gondolat.
7. Kovács, M. (Szerk.) (2017). *Társadalmi nemek. Elméleti megközelítések és kutatási eredmények.* Eötvös kiadó.
8. Huszár, Á. (2009). *Bevezetés a gender-nyelvészetbe.* Tinta.
9. Butler, J. (2007). *Gender trouble.* Routledge.
10. Sperling, G. B. & Winthrop, R. (2016). *What Works in Girls' Education: Evidence for the World's Best Investment.* Brooking Institution.
11. Crutzen, C. (2005). *Questioning Gender in E-learning and its Relation to Computer Science. Space for design, working, and learning.* In Braidotti, R. & van Baren, A. (Szerk.) The Making of European Women's Studies Vol. VI. University of Utrecht, pp.40-59.
12. Mansour, N. & Wegerif, R. (2013). *Science Education for Diversity: Theory and Practice.* Springer.
13. Frehill, L. M. & McGrath Cohoon, J. (2015). *Gender and Computing.* In Pearson, Jr. W., Frehill, L. M. & McNeely, C. L. (Szerk.) Advancing Women in Science. An International Perspective (pp. 237-264). Springer.
14. Rosser, S. V. (2017). *Academic Women in STEM Faculty.* Palgrave Macmillan.
15. Buolamwini, J., & Gebru, T. (2018). *Gender Shades: Intersectional Accuracy Disparities in Commercial Gender Classification.* Proceedings of Machine Learning Research 81:1–15. http://proceedings.mlr.press/v81/buolamwini18a/buolamwini18a.pdf (utoljára megtekintve: 2020.11.15.)
16. Kirkup, G. (2002). *ICT as a tool for enhancing women's education opportunities, and new educational and professional opportunities for women in new technologies.* United Nations Division for the Advancement of Women (UNDAW)
17. Bonder, G. (2015). *Foreword.* In Pearson, Jr. W., Frehill, L. M. & McNeely, C. L. (szerk.) Advancing Women in Science. An International Perspective (pp. v-viii). Springer
18. *Kerettantervek (2020)* https://www.oktatas.hu/kozneveles/kerettantervek/2020_nat (utoljára megtekintve: 2020.11.15.)
19. *Skool* https://skool.org.hu/ (utoljára megtekintve: 2020.11.15.)
20. Broy, M. (2003). *Object-oriented programming and software development — a critical assessment.* In: McIver A., Morgan C. (eds) Programming Methodology. Monographs in Computer Science. Springer, New York, NY. https://doi.org/10.1007/978-0-387-21798-7_10
21. Bernát, P. (2015). *Robotika az általános iskolában és a RoboMind programozási környezet* In: Szlávi, Péter; Zsakó, László (szerk.) INFODIDACT 2015, Webdidaktika Alapítvány
22. *Kamigami is a cute robot bug you build yourself* https://www.engadget.com/2017-10-11-mattel-kamigami-robot-kit.html (utoljára megtekintve: 2020.11.15.)
23. *UnicornBot Kit* https://www.ubtrobot.com/products/unicornbot-kit (utoljára megtekintve: 2020.11.15.)
24. *Dash - Wonder Workshop – US* https://www.makewonder.com/robots/dash/ (utoljára megtekintve: 2020.11.15.)
25. *Can Robots Help Get More Girls Into Science and Tech?* https://www.wired.com/story/can-robots-help-get-more-girls-into-science-and-tech (utoljára megtekintve: 2020.11.15.)
26. Master, A., Cheryan, S., Moscatelli, A. & Meltzoff, A. (2017). *Programming experience promotes higher STEM motivation among first-grade girls.* In: Journal of Experimental Child Psychology, Vol. 160, pp 92-106.

27. Screpanti, L. et al. (2018) *An Educational Robotics activity to promote gender equality in STEM Education.* In: International Conference on Information, Communication Technologies in Education (ICICTE 2018), pp. 336-346.
28. Bernát, P. (2020). *Teaching introductory programming by creating animations with Scratch.* In: New Methods and Technologies in Education, Research and Practice 2020. ELTE Informatikai Kar. pp. 30-39.
29. Kelleher, C. & Pausch R. (2007). *Using Storytelling to Motivate Programming.* In: Communications of the ACM. Vol. 50, No. 7, pp 59-64.
30. Bernát, P. (2017). *Feladattípusorientált játékfejlesztés a Scratch-ben.* In: INFODIDACT 2017, Webdidaktika Alapítvány, pp. 1-12.
31. *Parachute Flight on Scratch* (2019) https://scratch.mit.edu/projects/276366240/ (utoljára megtekintve: 2020.11.15.)
32. *Xbox Indies - Games - Kodu Game Lab.* http://xboxindies.com/game/kodu-game-lab/ (utoljára megtekintve: 2020.11.15.)
33. *Male and Female Gamers: How Their Similarities and Differences Shape the Games Market* https://newzoo.com/insights/articles/male-and-female-gamers-how-their-similarities-and-differences-shape-the-games-market/ (utoljára megtekintve: 2020.11.15.)
34. *Which games are women and girls playing?* https://www.polygon.com/2017/1/20/14337282/games-for-women-and-girls (utoljára megtekintve: 2020.11.15.)
35. American Association of University Women. Educational Foundation. Commission on Technology, Gender, and Teacher Education (2000). *Tech-savvy: educating girls in the new computer age.* American Association of University Women Educational Foundation.
36. Carmichael, G. (2008). *Girls, computer science, and games.* In: ACM SIGCSE Bulletin 40(4), pp. 107-110. DOI: 10.1145/1473195.1473233
37. *Lányok Napja* http://www.lanyoknapja.hu (utoljára megtekintve: 2020.11.15.)
38. *Lányok Napja - Korábbi évek számokban* http://lanyoknapja.hu/infografikak/ (utoljára megtekintve: 2020.11.15.)
39. Margolis, J. & Fisher, A. (2002). *Unlocking the Clubhouse. Women in Computing.* Cambridge Mass, MIT Press.
40. Szlávi, A. (2019). *Nemi sztereotípiák az informatika oktatásban.* InfoDidact 2019. Webdidaktika Alapítvány.
41. Szlávi, A. (2020). *Introducing the Gender Aspect into IT Education.* CEJNTREP Vol 2/2.
42. *Kahoot* http://www.kahoot.com (utoljára megtekintve: 2020.11.15.)
43. *Kahoot - Kahoot! AS reports first quarter 2020 results* https://kahoot.com/investor/announcements/kahoot-as-reports-first-quarter-2020-results/ (utoljára megtekintve: 2020.11.15.)
44. *Scratch – Imagine, Program, Share* https://scratch.mit.edu/ (utoljára megtekintve: 2020.11.15.)

# Scrum retrospektív technikák egyetemi kurzusokhoz

Bornemisza Barbara

hbddn5@inf.elte.hu
ELTE IK

**Absztrakt.** Napjainkban az agilis módszertanok, köztük a Scrum módszertan egyre nagyobb teret hódít a szoftverfejlesztési iparágban. Emiatt indokoltnak tartjuk, hogy az informatika szakos egyetemi hallgatók is megismerkedjenek ezzel a modellel. 2020 tavaszán 6 hallgatói Scrum csapattal valósítottunk meg Scrum módszertanos fejlesztést. Ebben a cikkben azokat a tapasztalatokat taglaljuk, melyeket a Scrum retrospektívek kapcsán szereztünk, és ajánlásokat fogalmazunk meg az egyetemi csoportokban hatékonyan levezethető retrospektíveket illetően.

**Kulcsszavak:** Retrospektív, Scrum, Agilis módszertanok

## 1. Bevezetés

Az agilis módszertanok új megvilágításba helyezik a szoftverfejlesztést. Ezek a módszertanok a régi, lineáris szoftverfejlesztési módszertanokkal szemben, új - a megrendelők esetlegesen változó igényeit még jobban fókuszba helyező - módszertanok. Főbb alapelveik, hogy üzleti értéket képviselő szoftvert készítsenek az ügyfeleknek minden egyes munkaszakasz végére, és a fejlesztők nyitottak legyenek az idővel változó elvárásokra. Hangsúlyozzák az egyének és a személyes kommunikáció fontosságát, valamint a megrendelővel történő együttműködést.

A Scrum egy agilis keretrendszer, mely munkaszakaszokra (futam) ütemezi a munkát, és különböző szerepkörök (pl. Scrum mester, Termékgazda, Fejlesztőcsapat), események és résztermékek elkülönítésével járul hozzá a hatékony munkához. A Retrospektív (Visszatekintő) egy az események közül, mely minden munkaszakasz végén valósul meg, és melynek célja reflektálni a közös munkára és optimalizálni azt. A Retrospektívek hozzájárulnak a Scrum csapat fejlődéséhez a csapattagok bevonásával.

Oktatási intézmény révén fontosnak tartjuk, hogy a hallgatóknak naprakész tudásuk legyen az iparban is alkalmazott keretrendszerekről, ezért alkalmazzuk a Scrum-ot egyre több gyakorlati órán. Emellett abban is hiszünk, hogy a konkrét gyakorlati órákon - ahol csoportban zajlik a fejlesztés - is hozzájárulhat a teljesítményhez, és elősegíti, hogy a csapatok hatékonyabban tudjanak működni.

Kíváncsiak voltunk arra, hogy az oktatás keretei között milyen Retrospektív technikákat lehet használni, annak érdekében, hogy megtanítsunk Retrospektív technikákat, és hogy hatékonyabbá tegyük a hallgatói csapatok munkáját. Több módszert is kipróbáltunk az egyetemen oktatott „Szoftvertechnológia" tárgyon belül. Összesen 6 csoporttal vezettünk le Retrospektívet, 3 különböző időpontban, 3 különböző módszerrel.

Ebben a cikkben ismertetjük, összehasonlítjuk, és kiértékeljük a gyakorlati órákon alkalmazott Retrospektív technikákat. Ezek alapján ajánlásokat fogalmazunk meg az egyetemi kurzusok keretein belül levezetendő Retrospektíveket illetően.

## 2. Kapcsolódó munkák

A Scrum oktatásba való bevezetése megoldást jelenthet olyan problémákra, mint a kevés motiváció és a rövid távú elköteleződés. A visszajelzések alapján hatékonyabbá teszi a kommunikációt, és jobban előtérbe kerül a csapatmunka[7].

A [2] cikk beszámol arról, hogy a tanórák keretein belül a Scrum keretrendszerrel való fejlesztés során több teljesen működő funkció készült el a megvalósítandó szoftverből, mint a hagyományos modellel való fejlesztésnél. A hallgatók saját bevallása szerint a teljesítményük javulna, ha rendszeresebben lennének számonkérések [3]. A Scrum keretrendszer a rendszeres napi Scrum-jai miatt erre alkalmas lehet.

A Scrum bevezetésére több módszert is találunk a szakirodalomban. Az[5] cikkben tényleges Scrum szerepkörök kiosztásával valósították meg a Scrum-ot, és hangsúlyozzák a megfelelő személyek kiválasztásának fontosságát az egyes szerepekre. A folyamatos visszajelzések, illetve a több, rövid fejlesztési szakasz növeli a csapatok hatékonyságát[1].

A Retrospektívek során használható játékokat több szempontból is lehet csoportosítani [4]. A kollaboratív játékos Retrospektívek jobb eredményre vezetnek, mint azok, amelyek során nem használunk játékokat, illetve fejlesztik a résztvevők kommunikációját és kreativitását is[10].

A szakirodalomban sok olyan cikk jelenik meg, ami kimondottan a Scrum Retrospektívekre fókuszál. A [6] cikk a Retrospektívek hatását elemzi ipari környezetben.

A Scrum keretrendszer folyamatait, köztük a Retrospektívet is, labdajátékkal tanították meg a résztvevő diákoknak [9]. A játék eredményéből levont konklúziók azt mutatták, hogy a csapatok teljesítményét javítja, ha van előre megadott időkeret arra, hogy a csapatok megbeszélhessék az előző körben bekövetkezett hibáikat, és újragondolják a stratégiát.

## 3. A kísérlet keretei

A fentiekben láthattunk példát arra, hogyan van jelen az oktatásában a Retrospektív. A továbbiakban bemutatjuk a mi kísérleteinket.

A „Szoftvertechnológia" tárgy célja, hogy a hallgatók megismerjék a szoftvertechnológia alapjait, és áttekintsék a szoftverfejlesztés folyamatait (szoftverspecifikáció, szoftvertervezés, szoftver validáció, szoftver evolúció). Lényeges az is, hogy ezt a folyamatot egyben lássák. A tárgy 5 kredites, és hetente körülbelül 10 óra munkát jelent. A heti egyszeri, 1,5 órás gyakorlati óra keretein belül a hallgatók kisebb (3-4 fős) csoportokban dolgoznak egy előre meghatározott szoftver projekten. Egy tantervi átalakításnak köszönhetően Scrum módszertanból vett elemekkel szervezik meg a csoportmunkát [8]. A reguláris órák során heti Scrum-ot tartanak az oktató kíséretében. (A Scrum szerepkörök nincsenek kiosztva a hallgatók között.) Időközönként - 3 hetente - prezentálják az addigi munkájukat (futam bemutatóhoz hasonlóan). Ezt a bemutatót a tárgy keretein belül mérföldkő bemutatónak nevezzük. A félév végére egy üzleti értéket képviselő szoftvert kell elkészíteniük – hűen az Agilis elvekhez.

### 3.1. A Retrospektívek helye a „Szoftvertechnológia" tárgyban

A „Szoftvertechnológia" keretein belül alap esetben a Retrospektívek nem képezik az órák részét, mivel erre nem áll rendelkezésre elég idő/energia. 6 csapattal kipróbáltuk, hogyan lehet megvalósítani a Retrospektívet, és azoknak milyen hatása lesz. A Retrospektíveket 3 alkalommal tartottuk meg, mindet az egyes mérföldkő bemutató utáni héten. A Retrospektíveken résztvevő csapatokat nem mi oktattuk, velünk csak a Retrospektív alatt találkoztak a hallgatók. Ez hozzájárulhat ahhoz, hogy a csapattagok őszintébben, nyíltabban elmondják a véleményüket, tekintve, hogy nem a mi feladatunk értékelni őket a tárgyból. Ez viszont az ellenkezőjét is kiválthatja, hisz érezhetik úgy is, hogy kívülállók vagyunk. A Retrospektívek során inkább az előbbi valósult meg a csapatoknál.

A Korona vírus okozta kijárási korlátozások miatt a Retrospektíveket online formában tartottuk meg, a Microsoft Office Teams nevű alkalmazáson keresztül, konferenciahívásban. Minden egyes csapatnak saját, névvel ellátott csatornája volt, ezeken keresztül zajlottak a hívások. Ilyen körülmények között ki kellett zárnunk a szociometriás gyakorlatokat, hiszen nem tudtak volna szabadon, minden résztvevő számára jól látható és érzékelhető módon mozogni a térben. Nehezebb volt összegyűjteni az résztvevőket a Retrospektívek elején, mert várni kellett, hogy mindenki ott legyen a gépnél, illetve esetlegesen megoldjuk a jelentkező mikrofon problémákat. Ez hozzájárult ahhoz, hogy a tervezett időkeretből 2 alkalommal is kicsúsztunk a 3 különböző Retrospektív során, illetve, hogy 2 alkalommal egy teljes csoporttal kevesebbel tudtuk csak elvégezni a Retrospektívet.

Egy csoportra körülbelül 15 perc jutott. Ez az idő jelentős mértékben rövidebb, mint a 2-4 hetes futamokhoz előírt 3 órás Retrospektív. Emiatt olyan módszereket kellett választani, amik ilyen rövid idő alatt is hatásosak lehetnek, és érdekesek is, emellett az online térben is kivitelezhetők. Mivel oktatási intézmény vagyunk, ezért fontosnak tartjuk azt, hogy megismertessük, és elmélyítsük a hallgatókban a Scrum értékeket. A Retrospektívek során ez a szempont is szerepet játszott a módszerek kiválasztásában, megalkotásában.

Fontos volt számunkra az is, hogy egy-egy Retrospektív során rá tudjunk látni a munkaszakaszokkal és Retrospektív technikákkal kapcsolatos visszajelzésekre, és azokra tudjuk építeni a további elképzeléseinket.

# 4. Az adatgyűjtés módja

A Retrospektívek hatásairól adatokat gyűjtöttünk, kíváncsiak voltunk a hallgatók visszajelzéseire. Az adatgyűjtéshez egy kérdőíveket állítottunk össze, melyeket minden Retrospektív után kitöltettünk a hallgatókkal, összesen 3 darabot.

## 4.1. Adatgyűjtés a csapatok teljesítményéről

A kérdőív első részében a csapatok működésére és teljesítményére vonatkozóan végeztünk méréseket. Ehhez olyan kérdéseket tettünk fel, melyekben a kitöltők 1-5-ig terjedő skálán (1 - Egyáltalán nem, 5 - Teljes mértékben) pontozhatták a munkájukra jellemző tulajdonságokat az előző futam fiegyelembevételével. A szempontok, amelyeket értékelniük kellett:

- Közérzetem a csapatban
- A csapat együttműködése
- A csapat teljesítménye
- A munka megfelelő megosztása
- A csapat kommunikációja
- A munkavégzés gyorsasága
- Az időbeosztás
- A halogatás elkerülése
- A kitűzött feladat megértése

Ezek a szempontok a csapatmunka fejlődését voltak hivatottak felmérni.

## 4.2. Adatgyűjtés a Scrum értékekről

A kérdőív második részében a Scrum értékek fejlődését mértük a csapatokban. Ehhez olyan kérdéseket tettünk fel, melyekben a kitöltők 1-5-ig terjedő skálán (1 - Egyáltalán nem, 5 - Teljes mérték-

ben) pontozhatták a Scrum értékek jelenlétére vonatkozó kijelentések igazságtartalmát. Az értékelendő állítások:

- „Meg mertem mondani, ha valami szerintem nem volt kivitelezhető abban a formában, amit előzetesen kigondoltunk, vagy azt, ha máshogy gondoltam valamit." (Bátorság értékre vonatkozott)
- „Nem szóltam bele más feladat-választásába, alkalmazott megoldási módszereibe." (Tisztelet értékre vonatkozott)
- „A csapattársaim tisztelettel bántak velem, nem nézték le a képességeimet." (Tisztelet értékre vonatkozott)
- „A futam során sikerült szem előtt tartanom a megbeszélteket, nem foglalkoztam olyan dologgal, ami nem volt kitűzve, mint futam cél." (Fókusz értékre vonatkozott)
- „Kötelességemnek éreztem, hogy elvégezzem a rám eső feladatokat, illetve azt, hogy hozzájáruljak valamivel a csapat munkájához, és ezért mindent meg is tettem." (Kötelezettségvállalás értékre vonatkozott)
- „Befogadóan kezeltem a megvalósítással kapcsolatos új ötleteket, illetve a változtatásokra vonatkozó javaslatokat." (Nyitottság értékre vonatkozott)

A kitöltőknek minden egyes Scrum értéket illetően 1 állítást kellett pontozniuk. Ez alól kivétel a Tisztelet, mint Scrum érték, mert ott 2 állítás volt, mert mindkét szempont fontos: hogy ők mennyire tisztelték a csapattársaikat, a saját meglátásuk szerint őket mennyire tisztelték a többiek. A kérdőívben az egyes állítások után nem volt odaírva, hogy mely Scrum értéket hivatottak felmérni, ez azért történt így, hogy ne befolyásolja a választásukat az érték konkrét megnevezése.

## 4.3. Adatgyűjtés egy adott Retrospektívről

A kérdőív harmadik részében az aktuális Retrospektívet értékelték ki a kitöltők. A kérdőívet kitöltőknek 1-5-ig terjedő skálán (1 - Egyáltalán nem, 5 - Teljes mértékben) kellett pontozniuk az adott Retrospektívet a megadott kérdések alapján. Továbbá le kellett írniuk a következő futamra javasolt javító ötleteiket, melyeket a Retrospektív segítségével meg tudtak fogalmazni. Az értékelendő kérdések:

- Mennyire érezted jól magad a Retrospektív alatt?
- Mennyire tartottad hasznosnak ezt a Retrospektívet?

Összehasonlítás szempontjából fontosnak tartottuk, hogy minden különböző módszer után friss élmények alapján tudják elmondani a véleményüket a hallgatók.

## 4.4. Adatgyűjtés a Retrospektívek közötti összefüggésekről

Az utolsó kérdőívben elhelyeztünk egy negyedik részt, mely az összes Retrospektívet hasonlítja össze egymással. A feltett kérdések:

- Melyik Retrospektív volt szerinted a leghasznosabb?
- Melyik Retrospektív volt szerinted a legérdekesebb?
- Melyik Retrospektív volt szerinted a legélvezetesebb?

Továbbá a kitöltőknek 1-5-ig terjedő skálán (1 - Egyáltalán nem, 5 - Teljes mértékben) kellett értékelniük a Retrospektívekre általánosságban vonatkozó kérdéseket:

- Mennyire segítette a munkátokat az, hogy voltak Retrospektívek?
- Mennyire segítették a Retrospektívek, hogy jobb rálátást kapj a csapatmunka működését befolyásoló tényezőkre?

- Jelöld meg, hogy mennyire tartod jónak/hasznosnak, hogy résztvettél a Retrospektíveken!

Ezen kívül lehetőséget biztosítottunk arra, hogy szabadon leírhassák észrevételeiket, megjegyzéseiket a Retrospektívekkel kapcsolatban.

A kérdőívek eredményét az egyes Retrospektívek eredményeinél elemezzük ki, illetve az alkalmazott Retrospektív technikák összehasonlításánál.

# 5. Az alkalmazott Retrospektív módszerek

## 5.1. Az Autós játék és az Elkezdeni-Folytatni-Befejezni módszer

### 5.1.1. Leírás

A legelső Retrospektív kiválasztásánál olyan szempontok érvényesültek, minthogy: egyszerű levezényelni, a hallgatók is gyorsan megértik, és oldja az első Retrospektívből adódó esetleges feszengést bennük. Fontosnak tartottuk, hogy bátran ki merjék mondani a véleményüket, ezért olyan játékot kerestünk, ahol nem rögtön egymás előtt kell elmondani a véleményüket az előző futamról, hanem át tudják gondolni, mindezt játékos formában.

Ehhez egy úgynevezett Autós játékot választottunk, melynek lényege, hogy a résztvevők 3 perc alatt leírnak egy autómárkát, amely szerintük a legjobban jellemezné a futamot. Példa: Ferrari - Mindent nagyon gyorsan el tudtunk végezni. Trabant - Mindent lassan végeztünk el. Amennyiben valamelyik résztvevő nem volt jártas az autómárkákban, neki megengedett volt jellemzőket írni a futamról. Ez azért volt szükséges, hogy senki ne legyen kirekesztve a csapatból, mindenki el tudja mondani az önálló véleményét. Az első lépésben a kigondolt autómárkát elküldték nekünk személyes üzenetben. Erre a lépésre azért volt szükség, hogy ne befolyásolják egymást. A Retrospektív második feladatát az elküldés után ismertettük.

A második feladat az Elkezdeni-Folytatni-Befejezni (Start-Stop-Continue) nevű játék volt. Az Elkezdeni-Folytatni-Befejezni játék lényege, hogy létrehoztunk három oszlopot, melyeket megcímkézünk ezzel a három szóval. Az „Elkezdeni" oszlopba olyan javaslatokat írunk, amit a következő futamban kéne csinálnunk, de a legutóbbi futamban nem csináltuk. A „Folytatni" oszlopba olyan javaslatokat írunk, amit jól csináltunk a legutóbbi futamban, és szerintünk a következőben is alkalmazni kéne. A „Befejezni" oszlopba olyan javaslatok kerülnek, amit rosszul tettünk az előző futamban, és a következőben el akarjuk ezeket kerülni. Ezután egyesével minden résztvevő elmondta, hogy milyen autómárkát választott, és miért azt választotta, majd az indoklását elhelyezte a megfelelő oszlopban (mi az oszlopokat tartalmazó dokumentumot képernyőmegosztással mindenki számára láthatóvá tettük). Miután mindenki végzett, lehetőség volt újabb javaslatok hozzáadására az egyes oszlopokhoz.

### 5.1.2. Tapasztalatok

Az autós játék során 1 csapat kivételével minden csapatban volt olyan tag, aki nem autómárkát írt, hanem jellemzőket a futamról, tehát jól tettük, hogy megengedtük, hogy lehessen ilyeneket is írni, így mindenki el tudta mondani a véleményét. Az Elkezdeni-Folytatni-Befejezni játékhoz tartozó táblázatban a résztvevők el tudták helyezni a választott autóhoz tartozó futam jellemzőket, de további javaslatokat nem nagyon tudtak mondani rávezető segítség nélkül (pl.: halogatás, egyeztetések gyakorisága stb.). A 15 perces időlimitet egyik csapatnál sem tudtuk tartani, mind a résztvevőkre való várakozás, mind az ötlethiány miatti lassabb haladásból adódóan. Annak ellenére, hogy 5 csapat vett részt a Retrospektíven a tervezett 6 csapat helyett is 10 perccel többet tartott az óra. Ez átlagban 20 perces Retrospektívet jelent csapatonként.

### 5.1.3. Eredmények

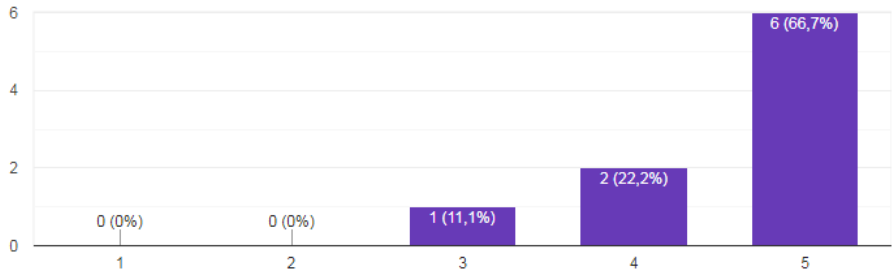
A Retrospektíven a 6 csapatból 5 csapat vett részt, a kérdőívet 9 ember töltötte ki a résztvevő 13 főből.

A visszajelzések alapján a legtöbb csapatnak a legnagyobb gondot a munka halogatása, illetve az jelentette, hogy nem volt előre egyeztetett időpont a megbeszélésekre. A Retrospektíven részt vett hallgatók közül a legtöbben az egymás segítését, és a gyors feladatmegoldást tartották a legkiemelkedőbbnek a munkájuk során. A résztvevők saját bevallása szerint ez a mérföldkő a feladat szempontjából még nem volt nagy kihívás. A választott autómárkák túlnyomó többségben a feladatmegoldás gyorsaságát hangsúlyozták, de akadtak olyanok is, melyek a késői munkakezdést szimbolizálták.

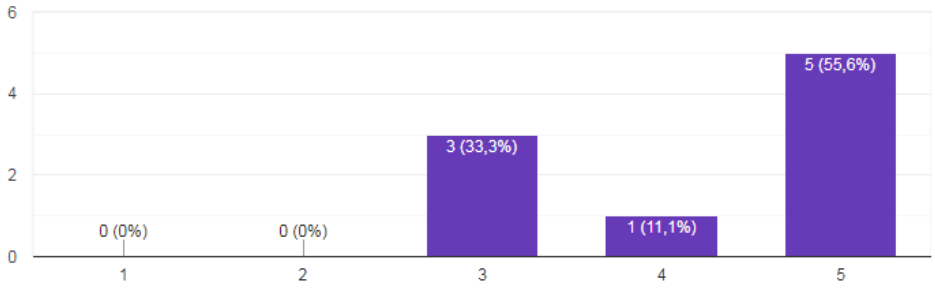
A kérdőívet kitöltők 66.7%-a teljes mértékben jól érezte magát a Retrospektív alatt. A legroszszabb értékelés a Retrospektív alatti közérzetre 3-as (közepes) volt. Ezek az adatok az 1. ábráról leolvashatók.

A kérdőívet kitöltők 55.6%-a teljes mértékben hasznosnak érezte ezt a Retrospektív technikát, és a legrosszabb értékelés a hasznosságra 3-as (közepes) volt. Ezek az adatok a 2. ábráról leolvashatók.

Fontosnak tartjuk kiemelni, hogy a 3-as (közepes) értékelést háromszor többen választották a hasznosság megítélésére, mint a közérzetre.




**1. ábra:** Közérzet az Autós játék és az Elkezdeni-Folytatni-Befejezni Retrospektív technika alatt




**2. ábra:** Az Autós játék és az Elkezdeni-Folytatni-Befejezni Retrospektív technika hasznossága

## 5.2. Bankrablós játék, Scrum értékek fejlesztése

### 5.2.1. Leírás

A második Retrospektívre olyan módszert választottunk, amely fejleszti az előző Retrospektív után felmért Scrum értékeket. Ez volt az a gyakorlati óra, mely előtt életbe lépett a vírus miatt elrendelt online oktatás. Ez az új helyzet új eszközök és módszerek használatát követelte meg a hallgatók és az oktatók részéről is egyaránt. Az előző retrospektív után beérkezett kérdőívek alapján a Tisztelet

értéket értékelték a legalacsonyabbra a résztvevők. Ennek fejlesztésére talált játékok az online oktatás keretében nehezen - vagy egyáltalán nem - lettek volna megvalósíthatók, ezért egy sajátos módszert dolgoztunk ki. Egy rögtönzött csapatmunkát igénylő játékot, mely során az volt a célunk, hogy megfigyeljük az egyes csapatok dinamikáját, a Scrum értékek jelenlétét, és fejlesszük őket. A játékosoknak 3 perc alatt kellett megtervezniük egy bank kirablását, úgy, hogy minden csapattagnak kellett feladatot kiosztani, és a menekülésről is gondoskodniuk kellett. Ehhez a feladathoz kaptak egy térképet, amin látni lehetett a bank alaprajzát, az ajtók, ablakok, kamerák, őrök és a riasztó pontos elhelyezkedésével együtt. A térképet képernyőmegosztással minden résztvevő számára láthatóvá tettük. Ezzel rá voltak kényszerítve, hogy tiszteljék egymás véleményét, és legyenek nyitottak az időközben felmerülő újabb nehézségek áthidalására. Az idő rövidsége miatt a fókusz és az elköteleződés értékeket is fejlesztette a játék, illetve a bátorságot is, hiszen minden ötletre szükség volt, hogy kitaláljanak egy stratégiát, mielőtt lejár az idő, mindezt úgy, hogy mi is ott voltunk a háttérben. Az idő lejártával kérdésekre kellett válaszolniuk, először a játékra, majd pedig a futamra vonatkozóan.

A feltett kérdések:

- Mennyire voltatok fókuszáltak?
- Mennyire tudtátok tiszteletben tartani a csapattársaitok ötleteit?
- Mennyire tudtátok bátran kimondani a felmerülő ötleteiteket?
- Mennyire éreztétek úgy, hogy nem érdekel titeket a feladat?
- Szükségetek volt egy vezetőre, hogy teljesítsétek a feladatot/mérföldkövet?
- Milyen érzés volt, hogy mindenkinek egyformán részt kellett vennie a feladat megoldásában?
- Volt olyan, aki elnyomva érezte magát?
- Milyen lépéseket lehetne tenni annak érdekében, hogy mindenki egyformán be legyen vonva a feladat megoldásába?
- Miben változott a csapatmunkátok az idő előrehaladtával? A jövőre nézve milyen tanulságot tudtok ebből levonni? Mi az a pár dolog, amin esetleg változtatni kéne?

### 5.2.2. Tapasztalatok

Eleinte a csapattagok szokatlannak tartották a Bankrablós játékot, nem nagyon tudták elhelyezni hogyan kapcsolódik ide, de a hozzá kapcsolódó, a futamra átvezető kérdéssorozat által jobban sikerült megérteniük a feladat hátterét. Az időt mi mértük, így ők nem látták a hátralevő idejüket, így az 5 résztvevő csapat közül 3 nem tudta betartani a rendelkezésre álló időt. Sok csapatnál eleinte csak 1-2 ember beszélt a 3-4 fős csapatból. Ez az ember aztán megkérdezte a társait, hogy nekik is jó-e az általa elmondott ötlet, ekkor a többiek is válaszoltak, esetleg módosítottak az ötleten. A kisebb létszámú csapatokban több szerep jutott az egyes csapattagoknak, az egyes tagok nem tudtak a háttérben maradni. Az egy csapatra szánt 15 perces időlimitet a legtöbb csapatnál nem sikerült tartani, hisz vagy túllépték a feladatra szánt időt, vagy a kérdések megbeszélése húzódott el. Mivel 5 csapat vett csak részt ezen a Retrospektíven a tervezett 6 helyett, így sikerült beleférnünk a 1,5 órába, tehát átlagban 18 percet vett igénybe egy Retrospektív.

### 5.2.3. Eredmények

A Retrospektíven a 6 csapatból 5 csapat vett részt, a kérdőívet 7 ember töltötte ki a résztvevő 11 főből.

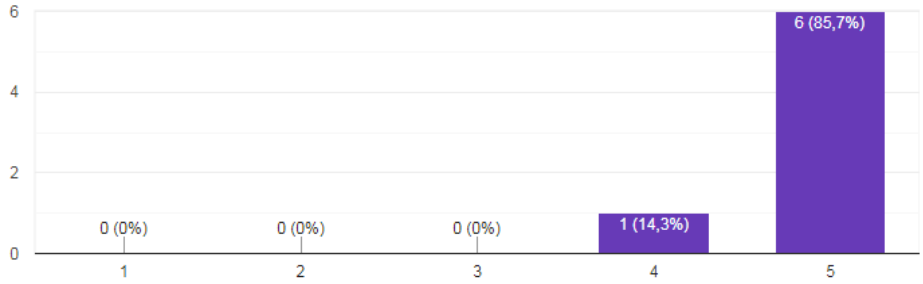
A fókuszáltságra, a tiszteletre, az érdeklődés megtartására, a csapaton belüli elnyomás érzésére adott válasz az összes csapatnál megegyezett mind a játék, mint a mérföldkő szempontjából. Az összes csapat fókuszált volt, csak néhány csapat nem teljesen az elejétől fogva. Egy csapatot kivéve minden csapat tiszteletbe tudta tartani a csapattársai véleményét. Két csapat furcsának találta a játé-

kot, és ezért nem teljesen figyelt oda, és ezek a csapatok a munka során is arról számoltak be, hogy nem végig volt kitartó az érdeklődésük. Egy csapat kivételével minden csapat bátor volt a játék és a mérföldkő alatt is. A kivételt képező csapat arról számolt be, hogy őket a játéknál feszélyezte, hogy mi is hallgatjuk, de a projektmunka során jó közösség alakult ki, így a csapattagok is bátrabbak voltak. A játék során minden csapatban úgy érezték, hogy mindenki be volt vonva a feladatba, ám a projektmunka alatt volt 2 csapat, ahol nem mindenki volt egyformán bevonva, ezek közül 1 csapatnál pedig 1 ember csinálta meg az egészet. Ennél a csapatnál az volt a javaslat annak érdekében, hogy mindenki be legyen vonva, hogy osszák le kisebb részfeladatokra a feladatot, és ezeket mindenkinek egyformán osszák el. A projektmunka során 4 csapatnál kialakult egy vezető, általában az, aki a legjobban értett a feladathoz. A játékban viszont csak 1 csapatnál alakult ki vezető, itt viszont a csapattársak szerint azért, mert csak neki volt jó a mikrofonja. 4 csapat természetesnek vette, hogy a csapatmunkában mindenkinek részt kell venni, és kifejezetten jónak is gondolták egymás segítése szempontjából. Ezeknél a csapatoknál a játékban is mindenki kivette a részét. A csapattagok bevonásánál említett kivételt képező csapatoknál a játékban mindenki kivette a részét, viszont a projektmunkát az egyik csapatnál csak egy ember csinálta. Az összes csapatban úgy érezték a csapattagok, hogy nincsenek elnyomva.

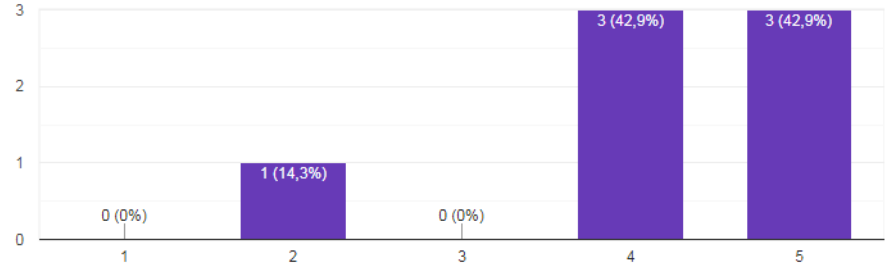
A csapatok úgy gondolták, hogy a legrosszabb a személyes kommunikáció hiánya volt. Ez az előző mérföldkőhöz képest már nem volt lehetséges a kialakult járványhelyzet miatt. A Retrospektíven kiderült az is, hogy az előző mérföldkőhöz képest bátrabbak lettek a munka során, illetve az egymás iránti empátiájuk is növekedett.

A kérdőívet kitöltők 85.7%-a teljes mértékben jól érezte magát a Retrospektív alatt, és a legrosszabb értékelés a Vissaztekintő alatti közérzetre is csak 4-es (jó) volt. Ezek az adatok a 3. ábrán olvashatók.

A kérdőívet kitöltők 42.9%-a teljes mértékben hasznosnak érezte ezt a Retrospektív módszert, és a legrosszabb értékelés a hasznosságra 2-es (kicsit) volt, ám ez is csak 14.3%-ot (1 embert) jelentett. Ezek az adatok a 4. ábrán olvashatók.




**3. ábra:** Közérzet a Bankrablós játék, Scrum értékek fejlesztése Retrospektív technika alatt




**4. ábra:** A Bankrablós játék és a Scrum értékek fejlesztése Retrospektív technika hasznossága

## 5.3. A három kismalac játék

### 5.3.1. Leírás

A harmadik Retrospektívre olyan módszert választottunk, amelyben nagyobb a csapat önállósága az egyes futamjellemzők kategorizálásában. Mivel a második Retrospektíven nem minden csapat figyelt a hátralevő időre, így nem is mindenki tartotta be pontosan a limitet, ezért itt is időlimites játékot akartunk alkalmazni. Okulva az előző időlimites Retrospektív tapasztalataiból, ebben a Retrospektívben már kivetítettünk egy visszaszámlálót, így minden csapat látta a még hátralevő idejét a feladat megoldására. Mivel az előző Retrospektívekhez tartozó kérdőívet nem mindenki töltötte ki, és ez a Retrospektív volt az utolsó, amit megtartottunk, ezért fontos tényező volt, hogy még a szokásosnál is kevesebb legyen a játékra szánt idő, hogy meg tudjuk várni, amíg kitöltik a kérdőívet a résztvevők. A három kismalac nevű játékot választottuk. A játék asszociál a „A három kismalac és a farkas" nevű mesére, mivel három oszlopot kell létrehozni, melyeket felcímkézünk szalmából, fából és téglából épült házakkal. A „Szalmaház" oszlopba olyan jellemzőket kell írni, amelyekről úgy éreztük, hogy nagyon gyenge lábakon álltak, épphogy megcsináltuk őket, tudtuk volna sokkal jobban is csinálni. A „Faház" oszlopba olyanokat, amit szerintünk nem csináltunk olyan rosszul, de azért tudnánk hozzájuk még javító ötleteket is mondani, jobban csinálni. A „Téglaház" oszlopba pedig olyan jellemzőket írunk, amiket saját bevallásunk szerint nagyon jól csináltunk, már nem tudnánk rajtuk javítani. A feladatot módosítottuk azzal, hogy előre megírt jellemzőket:

- Teljesítmény a csapatban
- Egymás segítése, támogatása
- Tehermegosztás
- Közérzet a csapatban
- Csapatmunka
- Nyitottság
- Bátorság
- Feladatmegértés
- Kommunikáció
- Tisztelet
- Munkavégzés sebessége

kellett elhelyezniük a csapattagoknak az egyes oszlopokba, és erre összesen 3 percük volt.

Lehetőségük volt a csapatoknak arra is, hogy új jellemzőket írjanak, amennyiben úgy érezték, hogy valami nem volt felsorolva, de ők fontosnak tartják, hogy beszéljenek róla. Az egyes oszlopokat, és jellemzőket képernyőmegosztással minden résztvevő számára láthatóvá tettük, és mi helyeztük el őket a megfelelő oszlopokba, kizárólag a csapattagok között lezajló kommunikáció alapján, ezzel párhuzamosan, annak érdekében, hogy ne emiatt fussanak ki az időből.

### 5.3.2. Tapasztalatok

Kedvezően hatott az, hogy kivetítettük a 3 perces időzítőt a csapatoknak az előző Retrospektívhez képest, hiszen csak 1 csapat nem tudta betartani az időlimitet a résztvevő 6 csapat közül. A résztvevő csapatok fele kihasználta az üresen hagyott jellemző cetlit, és írt még plusz jellemzőket a futamra vonatkozóan. Annak köszönhetően, hogy megvártuk, amíg mindenki kitölti a kérdőívet, több visszajelzést kaptunk, mint az előző Retrospektívek esetében. A Retrospektíven 6 csapat vett részt, és úgy is, hogy megvártuk, amíg kitöltik a kérdőívet (ennek időtartama max. 5 perc), sikerült befejeznünk 1,5 óra alatt a Retrospektíveket.

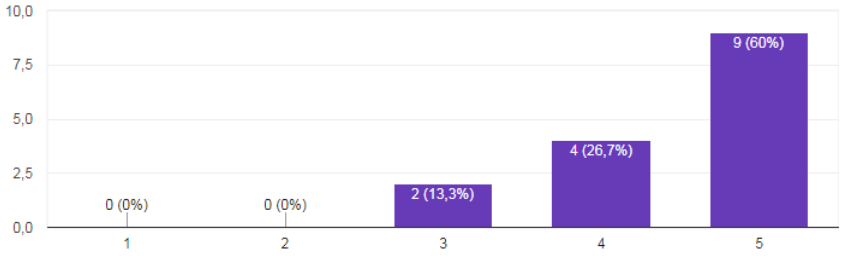
### 5.3.3 Eredmények

A Retrospektíven a 6 csapatból 6 csapat vett részt, a kérdőívet 15 ember töltötte ki a résztvevő 15 főből.

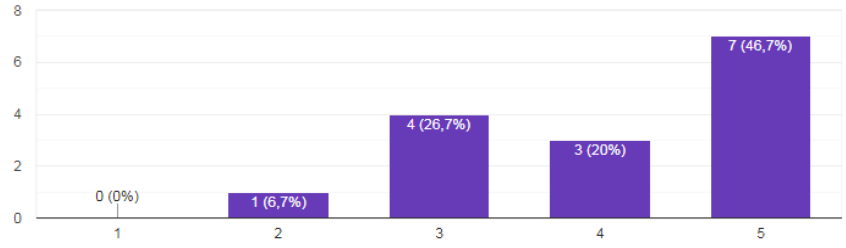
A legtöbb csapat a munkavégzés sebességét értékelte a legrosszabbra, de emellett 1-1 csapatnál megjelent a feladatmegértés és a „Teljesítmény a csapatban (végeredmény)" címke is. Az összes csapat a „Téglaház" oszlopba sorolta az egyes Scrum értékeket, tehát ezekről úgy vélekedtek, hogy maximálisan sikerült őket megvalósítaniuk az együttműködés során. A Scrum értékek mellett a csapatban való közérzetet is a legjobbra értékelték a csapatok.

A kérdőívet kitöltők 60%-a teljes mértékben jól érezte magát a Retrospektív alatt, és a legrosszabb értékelés a Vissaztekintő alatti közérzetre is csak 3-as (közepes) volt, amely 13.5%-ot jelent. Ez az 5. ábrán olvasható.

A kérdőívet kitöltők 46.7%-a teljes mértékben hasznosnak érezte ezt a Retrospektív módszert, és a legrosszabb értékelés a hasznosságra 2-es (kicsit) volt, ám ez is csak 6%-ot (1 embert) jelent. Az összes alkalmazott módszer közül ez kapta legdiverzebb értékelést a hasznosságát tekintve. Ez a 6. ábrán olvasható.




**5. ábra:** Közérzet A három kismalac játék Retrospektív technika alatt




**6. ábra:** A három kismalac játék Retrospektív technika hasznossága

# 6. Alkalmazott Retrospektív technikák összehasonlítása

A három különböző Retrospektívet és azok hatásait különböző szempontokból hasonlíthatjuk össze.

## 6.1. Csapatokra jellemző tulajdonságok fejlődése a félév során

A csapatok fejlődésére vonatkozó kérdésekre adott válaszokból megfigyelhetjük, hogy az átlag pontszámok a második Retrospektív után voltak a legmagasabbak. Ez a Retrospektív egy olyan mérföldkőnél volt, ahol még nem kellett, hogy kész legyen a teljes projekt, így a csapatokra lényegesen kevesebb nyomás helyeződött, mint a három kismalacos, végső verziót elváró mérföldkőnél. A közérzetre, a kommunikációra, a megfelelő időbeosztásra és a halogatás elkerülésére adott átlagos pontszám egyértelműen javult az első Retrospektíven kapott értékhez képest. Minden Retrospektíven próbáltuk hangsúlyozni a megfelelő időbeosztást, és a minél korábbi munkakezdés fontosságát, hiszen a

legtöbben ezeket hozták fel gyenge pontnak. Ennek javítása érdekében a félév során kiküldtünk egy ismertető leírást a feladat menedzselő alkalmazások hasznosságáról, illetve a Retrospektíveken is javasoltuk ezek használatát. Több résztvevő korábban nem hallott ezek létezéséről, és számukra hasznos volt ez az információ.

A 7. ábra szemlélteti az átlagos pontszámokra vonatkozó adatokat. Az egyes kategóriák szerinti minimum és maximum értékeket félkövér betűvel emeltük ki.

| | Közérzet | Együttműködés | Teljesítmény | Megfelelő munkamegosztás | Kommunikáció | Munkavégzés gyorsasága | Megfelelő időbeosztás | A halogatás elkerülése | Feladatmegértés |
|---|---|---|---|---|---|---|---|---|---|
| Start, Stop, Continue (1. Retrospektív) | **4.55** | 4.44 | **4.44** | 4.11 | **3.78** | **4.22** | **3.11** | **2.44** | 4.55 |
| Bankrablós játék (2. Retrospektív) | **4.86** | **4.57** | **4.29** | **4.29** | **4.71** | 4 | **3.71** | 3.14 | **4.86** |
| 3 kismalac (3. Retrospektív) | 4.6 | **4.33** | **4.4** | **4.07** | 3.8 | **3.93** | 3.6 | **3.4** | **4.4** |

**7. ábra:** Átlagos pontszámok a csapatok teljesítményére vonatkozóan az egyes Retrospektíveken adott visszajelzések alapján (1-5-ig lehetett pontokat adni)

## 6.2 Scrum értékek fejlődése a félév során

A félév során a csapattagok bátorsága nőtt, fókuszáltabbak lettek, és nyitottabbak az új ötletekre. A kötelezettségvállalásuk csökkent, ezt többen azzal magyarázták, hogy a félév előre haladtával megnőtt a rájuk nehezedő terhelés. A Retrospektíven használt technikák is leginkább a nyitottság, bátorság, és a fókuszáltság fejlesztését szolgálták azzal, hogy mindenkinek el kellett mondania a véleményét, és időre mentek a feladatok.

A 8. ábra szemlélteti az átlagos pontszámokra vonatkozó adatokat. Az egyes kategóriák szerinti minimum és maximum értékeket félkövér betűvel emeltük ki.

| | Bátorság | Tisztelet | Fókusz | Kötelezettségvállalás | Nyitottság |
|---|---|---|---|---|---|
| Start, Stop, Continue (1. Retrospektív) | **4.56** | 4.44 | **4.11** | **4.67** | **4.22** |
| Bankrablós játék (2. Retrospektív) | 4.86 | **4.71** | **4.86** | 4.57 | 4.71 |
| 3 kismalac (3. Retrospektív) | **4.93** | **4.2** | 4.2 | **4.53** | **4.8** |

**8. ábra:** Átlagos pontszámok a Scrum értékekre vonatkozóan az egyes Retrospektíveken adott visszajelzések alapján (1-5-ig lehetett pontokat adni)

## 6.3. A Retrospektívek hasznossága, érdekessége és élvezetessége

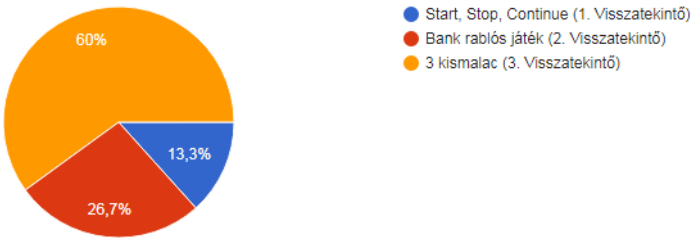
Mivel a „Bankrablós játék, Scrum értékek fejlesztése” módszert alkalmazó Retrospektíven kevesebben vettek részt, így arra értelemszerűen nem tudtak annyian szavazni, hisz azok, akik nem vettek rajta részt nem tudják összehasonlítani a többi módszerrel.

A kérdőívet kitöltők 60%-a a leghasznosabbnak „A három kismalac és a farkas” játékot alkalmazó Retrospektívet tartotta. A második helyen a bankrablós játékot alkalmazó, az utolsó helyen pedig az „Elkezdeni-Folytatni-Befejezni” módszer állt. Az első hely megválasztását az is befolyásolhatta, hogy közvetlenül ezt a módszert alkalmazó Retrospektívet követően kellett összehasonlítaniuk a résztvevőknek a Retrospektíveket, így még frissebb lehetett bennük ez az élmény, ám ez a százalék még ezt belekalkulálva is igen magas.

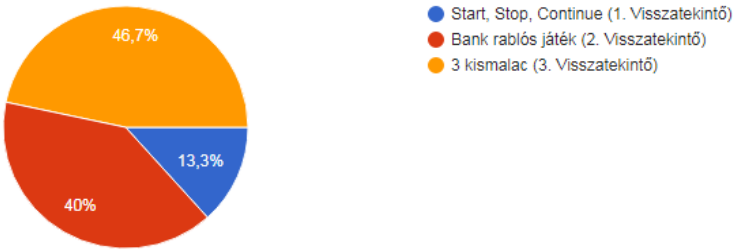
A kérdőívet kitöltők 46.7%-a a legérdekesebbnek „A három kismalac és a farkas” játékot alkalmazó Retrospektívet tartotta. A második helyen a bankrablós játékot alkalmazó, az utolsó helyen pedig az „Elkezdeni-Folytatni-Befejezni” módszer állt. A bankrablós játék módszert alkalmazó Retrospektív csak 6.7%-kal maradt le az első helyről, és ez sokkal kevesebb, mint a hasznosságnál jelentkező 33.3%.

A kérdőívet kitöltők 53.3%-a a legélvezetesebbnek a bankrablós játékot alkalmazó Retrospektívet tartotta. A második helyen „A három kismalac és a farkas” játékot alkalmazó, az utolsó helyen pedig az „Elkezdeni-Folytatni-Befejezni” módszer állt. A „A három kismalac és a farkas” módszert alkalmazó Retrospektív csak 13.3%-kal maradt le az első helyről. Az „Elkezdeni-Folytatni-Befejezni” módszerre csak egy ember szavazott ebben a kategóriában.

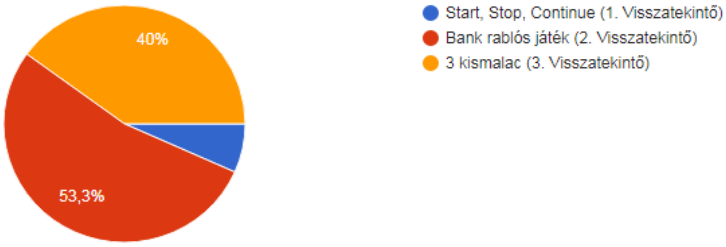
A fenti adatokat a 9., 10. és a 11. ábrák szemléletetik.




**9. ábra:** A leghasznosabb Retrospektív módszer




**10. ábra:** A legérdekesebb Retrospektív módszer




**11. ábra:** A legélvezetesebb Retrospektív módszer

# 7. Tapasztalatok elemzése és ajánlások

6 csapat 3 Retrospektívjének levezetését magába foglaló kísérletünk nyomán a következő ajánlásokat tudjuk megfogalmazni.

Amennyiben időlimites játékot szeretnénk játszani, akkor figyeljünk arra, hogy a rendelkezésre álló maradék idő a résztvevők számára is jól érzékelhető legyen, például stopper óra kivetítésével.

Ha előre megadunk Retrospektív szempontokat, ami szerint kell gondolkozzanak a csapatmunkájukról, az is nagyon segíti őket. Ez főleg az első Retrospektívek során lehet nagyon hasznos, a későbbiekben ezeket a szempontokat már maguktól is meg fogják tudni fogalmazni.

Az online formában való Retrospektív tartásnál érdemes előre küldeni a résztvevőknek emlékeztető üzenetet a csoportok által használt csatornákon keresztül, illetve e-mailben is, így minimalizálni tudjuk a csoporttagok csatlakozásához szükséges várakozási időt.

Amennyiben a Retrospektív alatt észleljük azt, hogy valamilyen módon többlet információkat biztosíthatnánk a hallgatók számára (például: munkamenedzsment eszközök ismertetésével), akkor van lehetőségünk dokumentációt készíteni arra vonatkozóan. Ajánlott ugyanakkor a következő Retrospektíven rákérdeznünk arra, hogy sikerült-e segíteni nekik ezzel. Amennyiben nem, akkor tovább próbálkozhatunk, hogy milyen más segítséget tudunk nekik adni.

Konkrétan az általunk használt játékokat illetően kiemelnénk, hogy az Autós játék nem biztos, hogy jó, mert nem mindenki ismer autómárkákat, ha valaki ismer, az is csak keveset. Ez az autós módszer nem tud annyira árnyalt képet adni.

A bankrablós játék elég jól leszimulálja a csapat valós dinamikáját. Tehát a Retrospektív során érdemes olyan csoport szituációba hozni a résztvevőket, ahol ezt meg lehet figyelni, így következtetéseket vonhatunk le a teljes csoportmunkára vonatkozóan, amit nem látunk, mert nem vagyunk ott.

A három kismalacos játék sok szempontból a legjobbnak bizonyult. A résztvevők érthetőnek tartották, be tudták tartani az időlimitet érdekesség és hasznosság szempontjából is nagyra értékelték.

A kísérlet során, a kutatási módszert illetően megtapasztaltuk, hogy amennyiben szeretnénk, hogy a résztvevők kitöltsék az általunk összeállított kérdőíveket, érdemes erre is időt hagyni, és megvárni, amíg minden résztvevő visszaküldi. Ezzel a módszerrel lehet a legnagyobb arányú kitöltést elérni.

A kísérletünknek volt egy hasznos melléktermék is. A Retrospektívek során kirajzolódni látszott előttünk egy lehetséges jó startégia a csapatmunkára. Az egyik résztvevő csapat arról számolt be, hogy kihasználva a távoktatás nyújtotta lehetőségeket, csoportosan fejlesztettek, a Microsoft Teams alkalmazás konferenciahívása segítségével. Az általuk megvalósítandó alkalmazás bonyolultabb funkcióit közösen, megbeszélve implementálták, az egyszerűbbeket egyedül, de eközben sem szakították meg a konferenciahívást. Ezek mellett feladat menedzselő alkalmazást is használtak, mely alkalmazásokról a kurzus folyamán küldtünk egy általános tájékoztató leírást.

## 8. Jövőbeli tervek

A továbbiakban célunk az, hogy ezen eredményeket felhasználva olyan Retrospektív segítésére képes alkalmazást fejlesszünk, mely automatizálja a Retrospektív folyamatát és ezek mellett ajánlásokat is képes kigenerálni a csapattagok által visszajelzett elakadásokat, gyengeségeket illetően.

## 9. Köszönetnyilvánítás

A kutatási projekt az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósult meg (EFOP-3.6.3-VEKOP-16-2017-00002).

## Irodalom

1. Charles Wallace, Sriram Mohan, Douglas Troy, and Mark E Hoffman: *Scrumacross the cs/se curricula: a retrospective.* InProceedings of the 43rd ACMtechnical symposium on Computer Science Education, pages 5–6, (2012).
2. Ramrao Wagh: *Using scrum for software engineering class projects* InAgile India, pages 68–71. IEEE, (2012)
3. Holló Csaba, Sárkány Rita, and Németh Tamás: *Hallgatói teljesítések javításának lehetőségei a felsőoktatásban informatikus szakokon* (2016)
4. Miloš Jovanović, Antoni-Lluís Mesquida, Nikola Radaković, and AntoniaMas: *Agile retrospective games for different team development phases.* Journal of Universal Computer Science, 22(12):1489–1508, (2016)
5. lyés Enikő: *Esettanulmány: Agilis szoftver-fejlesztés egyetemi kurzuson* (2017)
6. Jan Werewka and Anna Spiechowicz: *Enterprise architecture approach toscrum processes, sprint retrospective example.* In Federated Conferenceon Computer Science and Information Systems (FedCSIS), pages 1221–1228. IEEE, (2017)
7. Ilyés Enikő: *Agilis módszertan kutatás-fejlesztés laborban* (2018)
8. lyés Enikő, Pintér Balázs, Szendrei Rudolf, Cserép Máté: *A szoftver-technológia tárgy agilisra vált* (2019)
9. Jeffrey May, Jim York, and Diane Lending: *Play ball: bringing scrum intothe classroom.* Journal of Information Systems Education, 27(2):1, (2019)
10 Maciej Wawryk and Yen Ying Ng: *Playing the sprint retrospective*. In Federated Conference on Computer Science and Information Systems (FedCSIS), pages 871–874. IEEE, (2019)

# Hallgatók által vezetett Scrum Visszatekintők minősége

Cseh-Szabó Lilla

n8anic@inf.elte.hu
ELTE IK

**Absztrakt.** A Scrum módszertan egyre nagyobb térhódítása miatt, fontos, hogy már tanulmányaik során megismerkedjenek a hallgatók vele. A módszertant legeredményesebb gyakorlati úton elsajátítani. Kutatásom során egy egyetemi kurzus keretein belül vizsgáltam a Scrum oktatásának megvalósulását, ezen belül is a hallgatók által vezetett Scrum Visszatekintőket. Megfigyeléseim alapján ajánlásokat teszek a Visszatekintők minőségének javítására, ezáltal azok hatékonyabb oktatására vonatkozóan.

**Kulcsszavak:** Scrum, Visszatekintő, Agilis módszertanok

## 1. Bevezetés

Napjainkban egyre nagyobb teret hódítanak az informatikai iparban az agilis szoftverfejlesztési módszertanok, mint például a Scrum. [1] Ezért fontosnak tartjuk, hogy már az egyetemen megtörténjen a hallgatók oktatása erről. Így jött létre a „Scrum mester képzés" kurzus az egyetemünkön. Ezen kurzus keretein belül vizsgáltam a Visszatekintőket.

Cikkemben először ismertetem a Scrum módszertan lényegét. A második fejezetben leírok kapcsolódó kutatásokat. A harmadik fejezetben bemutatom a „Scrum mester képzés"-t, melynek kereteiben vizsgáltam a Visszatekintőket. A negyedik fejezetben leírom és elemzem a Visszatekintőket. Ötödik fejezetben megvizsgálom a hallgatók visszajelzéseit a kurzusról. A hatodik fejezetben ajánlásokat teszek a Visszatekintők minőségének javítására.

### 1.1. Scrum módszertan

A Scrum módszertant 1997-ban Sutherland és Schwaber [2] írta le. Ez egy agilis és iteratív szoftverfejlesztési módszertan. Annak nyomán jött létre, hogy felismerték: napjainkban a megrendelő hajlamos megváltoztatni a követelményeket a szoftverfejlesztés folyamata közben, erre pedig gyorsan kell tudni reagálni a termelékenység érdekében. A fejlesztést ennek érdekében futamokra osztják fel, melynek végén mindig egy üzleti értékkel rendelkező terméket kell bemutatni és melyet követően újra lehet aktualizálni a következő szakaszra vonatkozó követelményeket. A Scrum módszertan fő elemei a szereplők (a Fejlesztő csapat, a Termékgazda és a Scrum mester), az események (napi Scrum, Futamtervezés, Bemutató, Visszatekintő) és a termékek (Termék kívánságlista, Futam feladatlista, Inkrementum).

A Fejlesztő csapat keresztfunkcionális, így nem csak a termék fejlesztéséért, de teszteléséért is felelősek. A Termékgazda felel azért, hogy az ügyfélnek megfelelő funkciókkal és minőségben készüljön el a termék, ő képviseli az ügyfelet. A Scrum mester segíti a csapatot, hogy minél gördülékenyebben elkészüljön a vállalt feladat.

Mivel a fejlesztés futamokra van felosztva, minden futam egy Futamtervezéssel indul, hogy kiválasszák az adott idő alatt megvalósítandó feladatokat. Futam közben minden nap tartanak egy napi Scrum megbeszélést, melyen általában a "Mit csináltam tegnap? Mit fogok csinálni ma? Van-e valamilyen problémám?" kérdésekre válaszolnak röviden. A futamokat egy Bemutatóval zárják le, ahol a Termékgazdának bemutatják az elkészült funkciókat, aki azt átveheti. A Bemutató után a csapat a

Scrum mester vezetésével pedig tart egy Visszatekintőt, ahol megbeszélik milyen erősségeik és gyengeségeik voltak a futam alatt, min lehetne javítani.

A termékre vonatkozó elvárásokat a Termékgazda tartja karban a Termékkívánságlistában. Ez egy priorizált lista a feladatokról, amelyekből a Futamtervezés alatt választják ki az adott futamban megvalósíthatóakat. A kiválasztott elemeket átemelik a Futam feladatlistába. Ez csak az adott futamra vonatkozó feladatokat tartalmazza. Az Inkrementum a futam végén elkészült kész termék, mely az előző futamok eredményeinek kibővítése.

A Scrum módszertan nagy hangsúlyt fektet a csapatra és az együttműködésre. Így öt értéket szoktak kiemelni, melyek segítségével növelhetik a hatékonyságot. Ezek az elkötelezettség, a bátorság, a fókusz, a nyitottság és a tisztelet. Ezek a következőket jelentik:

- Elkötelezettség: a csapattársak és a termék irányába elköteleződni.
- Bátorság: felvállalni, ha nem tud valamit megcsinálni, kiállni egy rossz döntés ellen.
- Fókusz: segíti, hogy minőségi terméket szállítson a csapat a megadott határidőben.
- Nyitottság: megismerni mások véleményét, és megfontolni azokat.
- Tisztelet: tiszteletben tartani másokat, az idejüket, véleményüket.

### 1.2. Scrum mester

A Scrum mester a Scrum módszertan egyik szerepköre a Termékgazda és a Fejlesztő csapat mellett. A Scrum mester legfőbb feladata, hogy megértse és átadja a Scrum módszertan lényegét, értékeit, ezzel segítve a csapat munkáját. Megtanítja a többi szereplőnek az egyes Scrum elemeket, segít megérteni a módszertan lényegét.

### 1.3. Visszatekintő

A Visszatekintők lényege, hogy a csapat minden futam végén meghatározza az elmúlt időszak gyengeségeit és erősségeit.[3] Ezek közül általában kiválasztják a legfontosabbakat, amin változtatni kell és a következő futam során jobban figyelnek rá. A Visszatekintőt a Scrum mester moderálja. Ő határozza meg a menetét, állítja össze a feladatokat.

## 2. Kapcsolódó munkák

Egyre több egyetemen érzik már szükségesnek a Scrum oktatását. Sok helyen egy projekten keresztül mutatják be a hallgatóknak.

A [12] cikkben egy projekten keresztül mutatták be a hallgatóknak a Scrumot. Ez alapján ajánlásokat tesznek a kurzusra vonatkozóan, mikre érdemes különösen figyelni. Sikeresnek tartják ezt a fajta megközelítést, mivel az összes csoport megtanulta és alkalmazta a Scrum eszköztárat.

Egy másik tanulmány [10] szintén ezt a megközelítést alkalmazta az oktatásra, és a hallgatóktól pozitív visszajelzések érkeztek.

Ugyanakkor az egyetemi keretek között érvényesülnek olyan hátrányok is, melyek megnehezítik ezt a fajta oktatást.[11] Ilyenek például az adott időkeret vagy egyes hallgatók elérhetősége.

Visszatekintők segítésére különböző technikákat javasolnak a [9] és a [13] cikkben. A játékokat csoportosították aszerint, hogy miben tudnak segíteni, ezzel is segítve a csoportok egyéni igényeihez való igazodást.

# 3. A „Scrum mester képzés” kurzus

A 2019/2020 tanév tavaszi félévében indult el a "Scrum mester képzés" kurzus az ELTE Informatikai Karán. Korábban ilyen képzés ebben a formában nem került oktatásra az egyetemen. A kurzus kidolgozása során fontos volt, hogy a hallgatók ne csak az elméleti anyagokat hallgassák meg, hanem gyakorlatban is kipróbálhassák a Scrum módszertan működését.

A képzés során a hallgatók 3 fős csapatokban egy projektet vittek végig önállóan ebben a keretrendszerben. Minden csapat választott egy projekt témát, amit a Scrum eszköztár felhasználásával vezettek le a félév során. Ez lehetett egy szoftver elkészítése vagy akár egy esszé írása is. A félévet az oktatók három futamra osztották fel. Minden futamban rotálódtak a szerepkörök a hallgatók között, így mind a három szerepkörben kipróbálhatták magukat. Minden futam végén tartottak egy bemutatót, visszatekintőt, illetve egy futamtervezést a következő futamra. Ezek az órák csak ezzel teltek. Minden csapatnak jutott fél óra ezeknek a Scrum eseményeknek a levezetésére, ami A Scrum Útmutató [4] által megadott időkereteknél jelentősen kevesebb. A köztes órákon napi Scrumot tartottak. Mivel ez egy "Scrum mester képzés" volt, így ezeket az eseményeket nem az oktatók vezették, hanem a hallgatók, az aktuális szerepüknek megfelelően.

A kurzusra nagyarányú túljelentkezés volt, így két külön csoport indult. Az egyik csoportban 8, a másikban pedig 9 fővel zajlott az oktatás. Mindkét csoportban egy-egy csapat választotta, hogy szoftvert készít. A másik két-két csapat vállalta, hogy egy Scrummal kapcsolatos problémát körüljárva esszét ír.

Ebben a félévben nehézséget okozott a koronavírus miatti, félév közbeni digitális oktatásra való átállás. Ez az oktatóknak és a hallgatóknak is egy teljesen új szituáció volt. A hallgatók nem tudtak személyesen találkozni és együtt dolgozni, erre csak online formában volt lehetőség.

A kurzuson segítőként vettem részt. Az oktatót segítettem a tematika összeállításában, illetve egy-egy téma kapcsán kiselőadást tartottam.

## 3.1. „Retrospektív technikák” előadás

A Visszatekintők levezetéséhez egy külön előadást állítottam össze a hallgatók számára, ami ötletet adhatott a későbbi Visszatekintőik levezetéséhez. Az előadás során bemutattam Visszatekintő játékokat, technikákat, melyeket kutatócsoportunk gyűjtött össze. Ezek a következőek voltak:

- Autómárka [5]: Három kérdésre kell válaszolni. "Milyen autómárkára hasonlított a futam? Mi az álom autómárkád? Mit kellene tenni, hogy jobban hasonlítson rá?"
- Start - Stop - Continue [6]: 3-3 dolgot kell felírni az előző futamról, amit elkezdenénk, folytatnánk és abbahagynánk az egyes csapattagok a csapatot illetően.
- Valamit valamiért [7]: Tulajdonságpárok között egy vonalon jelezzük, mit mennyire tartunk fontosnak. Ez lehet akár termékkel kapcsolatban (könnyű kezelés) vagy csapat tulajdonságaival (gyorsaság) is.
- DAKI - Drop, Add, Keep, Improve – [7]: A csapattagoknak négy kategóriába, eldobni, hozzáadni, megtartani, fejleszteni, kell legalább egy dolgot írni a futammal kapcsolatban.
- Nirvana [7]: 3 fős csapatokban kell leírni, hogy milyen az ideális munkahely a számukra.
- Elismerő kérdések [5]: Pozitív kérdéseket teszünk fel a csapatnak, amiket megválaszolnak, annak érdekében, hogy megerősítsék a csapattagok önbizalmát, motivációját.
- Öt miért [5]: Egy konkrét probléma gyökerét megkeresendő feltesszük a miért kérdést többször egymás után. Majd a gyökér probléma megoldására próbálunk módszert találni.

Emellett megosztottam szakirodalmat, ahol még jobban után olvashattak a Visszatekintőknek. A hallgatók választhattak az általam bemutatottakból, de hozhattak új játékokat is.

# 4. A visszatekintők menete

Általánosságban elmondható, hogy a két csoportban jelentős eltérés volt a Visszatekintők minőségében.

Az első csoport nagyon összeszedett és felkészült volt. A Visszatekintőkre több játékkal is készültek, újakat is hoztak a bemutatottakon kívül. Ezért többször kicsúsztak a megadott időkeretből is. Minden alkalomra prezentációk készültek, vagy egy online eszközt hoztak a csapatok. A csapattagok visszanyúltak társuk munkájához, így a Visszatekintőknek egy íve volt.

A második csoport nem foglalkozott ennyit a felkészüléssel. Egy-egy kérdést tettek csak fel, hogy ki, hogy érezte magát a futam alatt. Volt, aki elfelejtett készülni, és az utolsó alkalommal volt, hogy teljesen el is maradt egy csapat Visszatekintője.

A továbbiakban részletesen bemutatom a csapatok munkáját. Közkedveltek voltak azok a játékok, melyek során kategóriákba kellett sorolni a futammal kapcsolatos visszajelzéseket. Ilyenkor néhány kategóriát sorolt fel a Scrum mester (pl. Start, Stop, Continue - Elkezdeni, Befejezni, Folytatni), és a csapat tagoknak ezekhez a kategóriákhoz kellett példákat mondaniuk a csapat szokásait illetően. A továbbiakban ezekre a játékokra a kategóriák felsorolásával hivatkozok. A csapatok megnevezése a továbbiakban a kurzus időpontjából adódott, az első csoportnak hétfőn (H), a második csoportnak csütörtökön (Cs) volt a gyakorlata.

## 4.1. H1 csapat

Az első Visszatekintőre két játékot hozott a csapat: Elismerő kérdések és Konstruktív kérdések.

A másodikat egy online eszköz segítségével tartották meg, ez a FunRetro [8] volt. Ez nagyon hasznos volt a digitális oktatás alatt. Egy link segítségével az oktatók és a hallgatók is hozzáférhettek a játékokhoz és láthatták a válaszokat. Itt először egy 1-től 5-ig tartó skálán be kellett jelölni mennyire érezték magukat biztonságban, majd Mad, Sad, Glad (mérges, szomorú, boldog) oszlopokba írni egy-egy dolgot.

A harmadik alkalommal szintén ezt az eszközt hívták segítségül. Első feladat a kész termékkel kapcsolatban kellett Happy, Meh és Sad (boldog, meh, szomorú) oszlopokba egy-egy dolgot írni, majd a közös munkára vonatkozóan a 3 kismalac mesében lévő szalma-, fa- és téglaháznak megfelelően egy-egy dolgot.

## 4.2. H2 csapat

Az első Visszatekintőn ráhangolódásként a Scrum mester először megkérdezte van-e bárkinek bármi, amit el szeretne mondani. Utána jellemezni kellett a futamot egy szóval, és megmagyarázni, miért azt a szót választották. Ezt követően különböző érzéseket megjelenítő emotikonokból kellett választani, ezzel jellemezve a futammal kapcsolatos domináns érzésüket. Utána adatgyűjtés és megértés szakasz következett, azaz a Scrum mester kérdéseket tett fel, mi ment jól, mit lehetett volna másképp? Harmadik feladatként a csapatnak egy pókháló diagramot kellett összeállítania, a neve Radar. Itt minden résztvevőnek hat kategória mentén (Az átadott termék minősége, Motiváció, a csapat lelkesedése, A csapaton belüli kommunikáció, Scrum szabályok betartása, A felhasználói történeteknek és a futam céljának érthetősége, A csapat teljesítménye) kellett osztályozni a futamot egy és tíz között. Ezt tantermi körülmények között nehéz kivitelezni, az eredményt megmutatni. A következő feladat címe "Mi tévők legyünk?", és célja, hogy ki mit mikor tud javítani. Lezárásként pedig újabb kérdések voltak: „Miben rejlik a csapat szuperereje?” és „Ezt tanultam, Ez volt vagány, Ezt csinálhatjuk jobban”. Ezen az alkalmon a csapat túllépte a megadott időkeretet, erre a későbbiekben jobban figyeltek és egyes feladatokra előre készültek. A Scum mester saját bevallása szerint sok energiát fektetett a Visszatekintő összeállításába, és ezt a későbbiekben a csapattársai ki is használták.

A második alkalomra az első alkalommal használt séma megmaradt, csak pár dolgot írtak át benne. A ráhangolódás és a lezárás szinte ugyanaz volt. A radar itt is megmaradt, csak már előre elkészítették és a kész diagramot mutatták be. Egyrészt az online órán nem tudták volna kivitelezni a megjelenítést, másrészt ezzel időt spóroltak tanulva az előző alkalom hibájából. Második feladatként pedig a DAKI (Drop, Add, Keep, Improve - eldobni, hozzáadni, megtartani, fejleszteni) játékot hozták.

Harmadik alkalomra szintén ugyanaz volt a kezdés és zárás. A radar szintén változatlanul megmaradt. A második feladat a Starfish – Tengeri csillag (Start, More, Keep, Less, Stop – elkezdeni, többet csinálni, folytatni, kevesebbet csinálni, befejezni) volt.

## 4.3. H3 csapat

Ennél a csapatnál az első futam megbukott, és menet közben szerepcsere történt. Feltehető, hogy ez volt az oka annak, hogy egy kérdés hangzott el az első Visszatekintőn: „Hogy érezted magad a futam alatt?”

A csapatot, saját elmondásuk szerint inspirálta a többi csapat munkája, így a második alkalomra készült egy prezentáció két játékkal. Az elsőnél mindenkinek kellett mondania egy igaz és egy hamis állítást a futammal kapcsolatban, és a többieknek ki kellett találnia, hogy melyik melyik. A második játék során egy pizzát osztottak szét, úgy, hogy mindenki kapott két számot egy és hat között. A pizza szeleteken kérdések, kérések voltak, melyekre válaszolnia kellett, annak a csapattagnak, akinél az adott szám volt.

Az utolsó Visszatekintő során a H2 csapattól inspirálódtak. Itt szintén egy prezentáció készült. Ráhangolódásként ki kellett választani egy csokit, illusztrálva, hogy mire hasonlított a futam. Utána egy online eszköz segítségével rajzolva az egész csapatnak együtt szobrot kellett emelnie a kész terméknek. Előre készítettek egy radarképet a teljes projektmunkáról. Lezárásként megbeszélték, hogy milyen eredményeket látnak a félévre visszatekintve.

## 4.4. CS1 csapat

Csak kérdések hangzottak el, nem készültek prezentációk. Hogy érezték a futamot, akadályokat és pozitív dolgokat kellett említeniük. Az utolsó Visszatekintő pedig elmaradt.

## 4.5. CS2 csapat

Itt is az első Visszatekintőn csak a "Hogy érezted magad a futam alatt?" kérdés hangzott el. A második alkalommal egy Word dokumentumba SWOT elemzést készítettek. Az utolsó alkalmon pedig egy szóval kellett jellemezniük a félévi munkát.

## 4.6. CS3 csapat

Az első alkalom hasonlóan zajlott a másik két csapathoz. A második Visszatekintőn szintén a "Hogyan érezted magad a futam alatt?" kérdésre kellett válaszolni. A harmadik alkalomra nem készültek így, a "Mi tetszett és mi nem?" kérdésre kellett válaszolni.

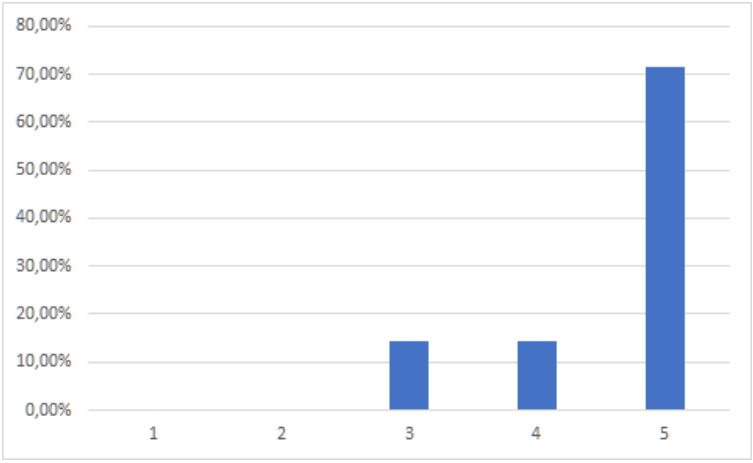
# 5. Visszajelzések

A félév végén készítettünk egy elégedettségi felmérést a kurzus kapcsán. Itt a kérdésekre válaszként egy egytől ötig terjedő Likert-skálán jelölhettek a hallgatók, egy jelentette a „legkevésbé sem” az öt pedig a „teljes mértékben” értéket. A kérdőívet az első csoportban 1 hallgató nem töltötte ki, míg a második csoportban mindenki kitöltötte.

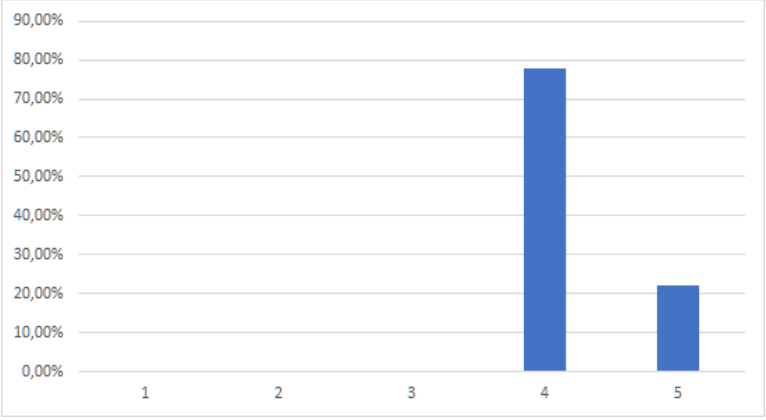
## 5.1. „Retrospektív technikák” előadás hasznossága

A kérdőívben megkérdeztük a hallgatókat, hogy melyik előadást mennyire érezték hasznosnak, köztük a "Retrospektív technikák"-at.

Az 1. ábrán látható, hogy a hétfői csoport 72 %-a tartotta nagyon hasznosnak az előadást, míg a többiek kevésbe. Ezzel szemben a 2. ábrán látható, hogy a csütörtöki csoportban kevésbé érezték hasznosnak ugyanazt az előadást.




**1. ábra:** Az előadás hasznossága a hétfői csoport számára



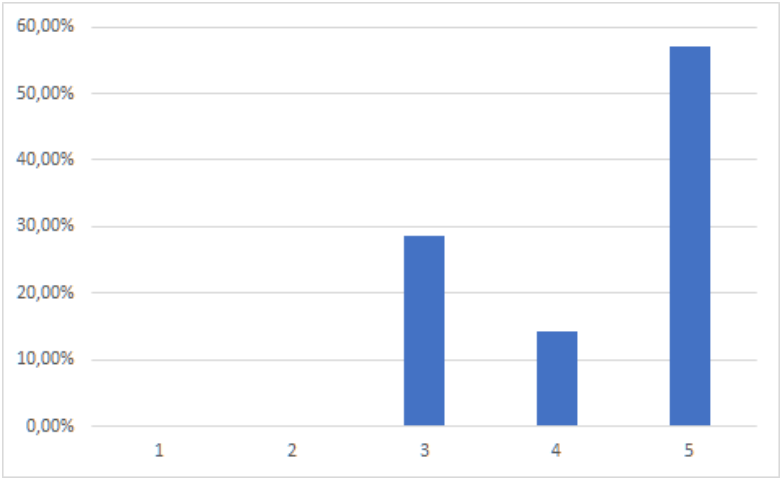

**2. ábra:** Az előadás hasznossága a csütörtöki csoport számára

A Visszatekintők minőségén is meglátszott ez a különbség. A hétfői csoport, akik hasznosabbnak érezték, ők sokkal színvonalasabb technikákkal vezették a Visszatekintőket. Nem csak felhasználtak egyet-egyet a bemutatott játékok közül, hanem újakat is hoztak, ezzel színesítve az órákat és egymás látókörét. Ezzel szemben a csütörtöki csoport, akik a Retrospektív előadást kevésbé érezték hasznosnak, nem is használták fel a bemutatottakat.

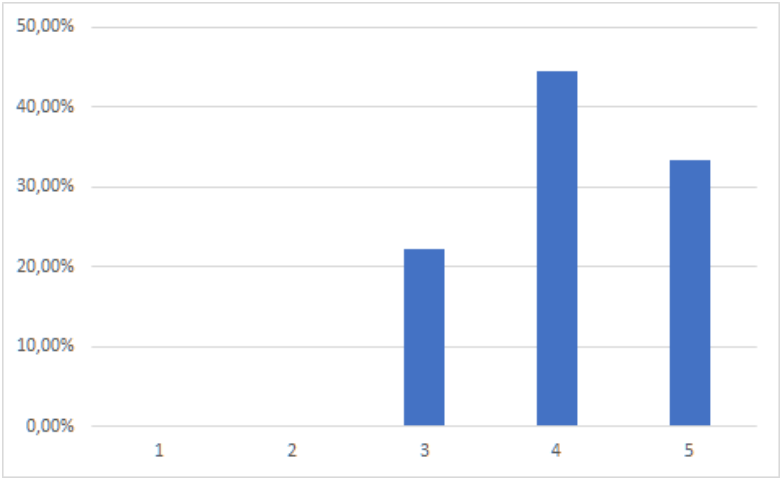
## 5.2. Scrum mester szerepkör kipróbálása

Mivel a Scrum mester egyik fontos feladata a Visszatekintők vezetése, megkérdeztük a hallgatókat, mennyire érezték hasznosnak, hogy kipróbálhatták magukat Scrum mester szerepkörben a félév során.

Az oktatók által alappillérnek tartott Scrum mesteri szimulációt a hétfői csoport 58 %-a tartotta fontosnak, ez látható a 3. ábrán. A 4. ábrán látható, hogy a csütörtöki csoportban ezt is kicsit kevésbé érezték hasznosnak.




**3. ábra:** A szimuláció hasznossága a hétfői csoport számára

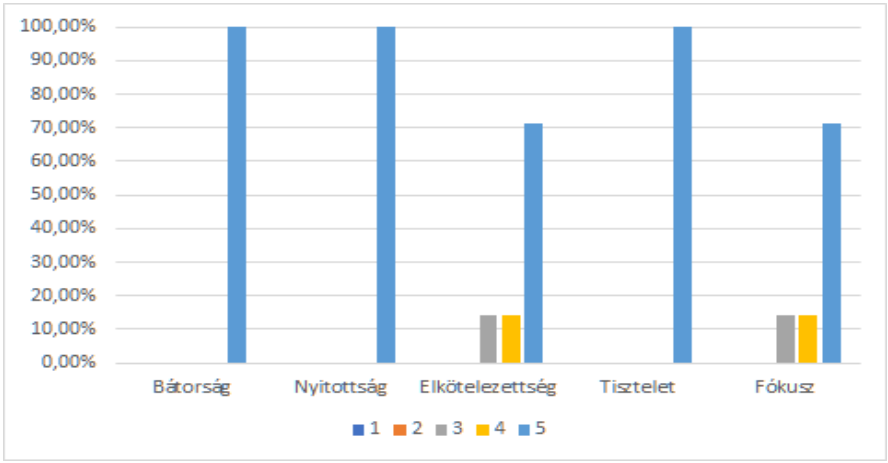



**4. ábra:** A szimuláció hasznossága a csütörtöki csoport számára

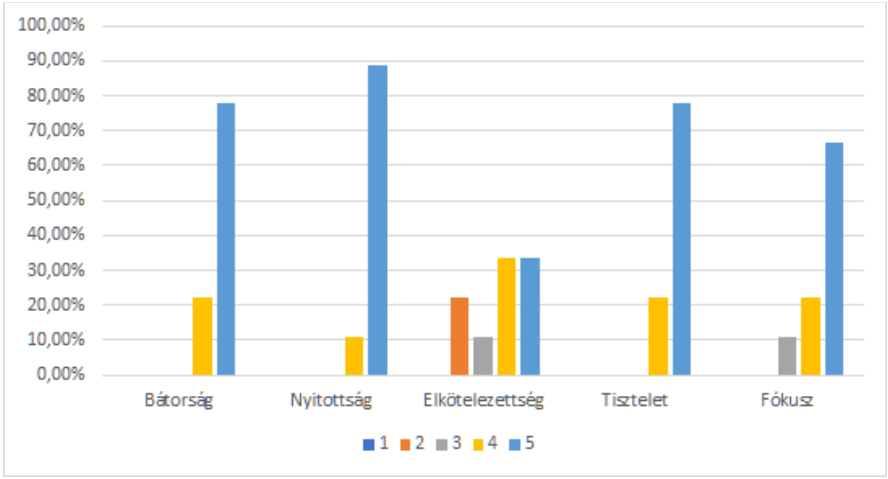
Itt nem tudok egyértelmű következtetést levonni, hiszen ugyanannyira érezték hasznosnak, mégis a Visszatekintők színvonala különbözött.

## 5.3. Scrum értékek

A kurzus során nemcsak elméleti tudás átadása volt a cél, hanem a gyakorlati oktatás és a Scrum értékek meghonosítása is. A Scrum értékek a bátorság, nyitottság, elkötelezettség, tisztelet és fókusz. Megkérdeztük a hallgatókat, hogy ezek a csapatmunka során mennyire teljesültek. Válaszaik az 5. és 6. ábrán láthatóak.




**5. ábra:** Az értékek jelenléte a hétfői csoportban




**6. ábra:** Az értékek jelenléte a csütörtöki csoportban

Érdekes megfigyelni, hogy egy nagy különbséget lehet észrevenni a csoportok között, mégpedig az elkötelezettség értékelésében. Felmerül a kérdés, hogy a csapatok elkötelezettsége és a Visszatekintők színvonala között általánosságban van-e valamilyen összefüggés, vagy ez csak a véletlen műve volt ebben az esetben.

## 6. Ajánlások

A cikkben bemutatott csapatok megfigyelésére alapozva a következőket ajánlom:

Ha nem elég minőségiek a hallgatók által tartott Visszatekintők, akkor érdemes mutatni nekik egy, az elvárt szinthez hasonló Visszatekintőt. Megfigyelhettük, hogy egy-egy Scrum csapat által megvalósított jó minőségű visszatekintő inspirálni tudta a többi csapatot is. Összeállítható egy prezentáció, amelyben 2-3 játék segítségével konkrétan bemutatható egy átlagos Visszatekintő menete, nem csak felsorolásszerűen a játékok, mint a fent említett „Retrospektív technikák" előadáson.

Amennyiben ez nem segít, felajánlható egy feladat lista, melyből össze kell válogassák a hallgatók a Visszatekintő feladatait. Egy a kurzushoz tartozó felületen publikussá tehető játékok pontos leírása, esetleg hivatkozással ellátva. Ezek külön kategóriákba sorolhatók, például: ráhangolódás, levezetés, lezárás. Kategóriánként egyet-egyet választhatnak a hallgatók.

Végső soron át lehet venni a hallgatóktól a Visszatekintők vezetését, hiszen abból is többet tanulhatnak, mintha egyáltalán nem készülnének a Visszatekintő levezetésére, esetleg emiatt az el is maradna.

## 7. Összefoglalás

A cikkben tehát egy egyetemi kurzus keretén belül vizsgáltam a hallgatók által vezetett Visszatekintőket. A félév során három Visszatekintőt kellett a hallgatóknak kitalálnia és levezetnie. Ezeket megvizsgálva ajánlásokat tettem ezek minőségének javítására.

A tapasztalatok alapján egy motivált csapat már feljebb tudja húzni az egész csoport Visszatekintőinek színvonalát. A Visszatekintőket megfigyelve látható volt, hogy a hallgatóknak gondot okoz az időkorlátok betartása, de ez tapasztalatot gyűjtve javul. Megfigyelhető volt, hogy egy csoportban különböző csapatok inspirálni tudják egymást, ezzel növelve a színvonalat. Érdekes összefüggésre utalhat a csapatok elkötelezettségének és a Visszatekintők minőségének az együtt-mozgása: ahol kevésbé volt elkötelezett a csapat, ott kevésbé színvonalasak voltak a Visszatekintők.

További kutatási lehetőség az ajánlásokat figyelembe véve megtartani a kurzust, megfigyelni milyen eredményeket hoznak a javaslatok. Egy másik lehetőség a csapat elkötelezettségének és a Visszatekintők színvonalának összefüggésének vizsgálata.

## 8. Köszönetnyilvánítás

A kutatási projekt az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósult meg (EFOP-3.6.3-VEKOP-16-2017-00002.

Továbbá szeretném megköszönni a segítséget témavezetőmnek, Ilyés Enikőnek, az inspirációt kutatótársamnak, Bornemisza Barbarának és a részvételt a hallgatóknak.

## Irodalom

1. 14th Annual State of Agile Report, https://stateofagile.com/#ufh-i-615706098-14th-annual-state-of-agile-report/7027494 (utoljára megtekintve: 2020.09.27.)
2. Ken Schwaber: *Scrum development process*. Business object design and implementation (1997) 117–134.
3. Esther Derby, Diana Larsen, and Ken Schwaber: *Agile retrospectives: Making good teams great*. Pragmatic Bookshelf (2006)

4. Ken Schwaber, Jeff Sutherland: *A Scrum útmutató*, https://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-HU.pdf (utoljára megtekintve: 2020.09.27.)
5. Luis Goncalves and Ben Linders: *Getting value out of agile retrospectives: a toolbox of retrospective exercises.* Ben Linders Publishing (2015)
6. Mike Cohn. *A simple way to run a sprint retrospective*, https://www.mountaingoatsoftware.com/blog/a-simple-way-to-run-a-sprint-retrospective (utoljára megtekintve: 2020.09.27.)
7. Paulo Caroli and Taina Caetano: *Fun retrospectives-activities and ideas formaking agile retrospectives more engaging*. Leanpub, Layton (2015)
8. https://funretro.io/
9. Miloš Jovanović, Antoni-Lluís Mesquida, Nikola Radaković, and AntoniaMas: *Agile retrospective games for different team development phases*. Journal of Universal Computer Science, 22(12):1489–1508, (2016)
10. Viljan Mahnic: *Teaching Scrum through Team-Project Work: Students' Perceptions and Teacher's Observations*. International Journal of Engineering Education 26.1 (2010): 96.
11. Zainab Masood, Rashina Hoda, Kelly Blincoe: *Adapting Agile Practices in University Contexts*. Journal of Systems and Software 144 (2018): 501-510.
12. David F. Rico, Hasan H. Sayani: *Use of agile methods in software engineering education*. 2009 Agile Conference. IEEE, (2009).
13. Christoph Matthies, Franziska Dobrigkeit, Alexander Ernst: *Counteracting Agile Retrospective Problems with Retrospective Activities*. Springer, Cham, (2019).

# Online robotika foglalkozások a karantén időszak alatt – Kihívások és tapasztalatok

Gaál Bence

gaalbence@inf.elte.hu
ELTE Informatikai Kar

**Absztrakt.** A koronavírus okozta helyzet ellehetetlenítette a hagyományos, jelenléti nyári táboraink és foglalkozásaink megtartását, ezért úgy döntöttünk, hogy a foglalkozásokat online formában tartjuk meg. A lelkes diákok így olyan robotika foglalkozásokon vehettek részt, amelyek teljes mértékben az online térben valósultak meg. Cikkemben ezen foglalkozások tapasztalatait, kihívásait, valamint a megvalósítás módját és a hozzá szükséges eszközök tárházát kívánom bemutatni.

**Kulcsszavak:** robotika, Covid-19, távoktatás, informatikaoktatás, szakkör

## 1. Bevezetés

A cikkem tárgyát képző, robotika szakkörök az ELTE T@T kuckó szervezésében valósultak meg 2020 nyarán három turnusban. Mivel a célcsoport a 13-16 éves korosztály volt, a kódolást már nem blokkalapú programozási környezetben kívántuk megvalósítani. A foglalkozásokat alapvetően olyan diákok számára terveztük meg, akik még nem programoztak Python nyelven, és/vagy nem találkoztak még a micro:bittel. A szakkör célja tehát az eszköz megismerésén kívül a programozás alapvető elemeinek elsajátítása volt. A három hét folyamán mindegyik foglalkozás teltházzal (14 fő) indult, lemorzsolódás pedig egyáltalán nem volt. A szakkörhöz kapcsolódóan megfogalmazott hipotézisünk az volt, hogy az online térben történő robotika szakkör, éppúgy eredményes és hatékony lehet az érdeklődés és a motiváció serkentésére a robotika és a programozás terén, mint a jelenléti foglalkozások.

A programozási nyelv kiválasztásánál fontos szempont volt, hogy olyan nyelvet használjunk, amely kezdők számára is ideális. A Python nyelv előnyei közé sorolható többek között az egyszerűsége, a biztonsága, valamint az objektumorientáltság támogatása.[1] Kutatás bizonyítja azt is, hogy a diákok szórakoztatóbbnak találták és könnyebben elsajátították a Python nyelvet, mint azokat, amelyeket korábban használtak.[2] Fontos azonban, hogy nem csak előnyei vannak a nyelv használatának. Mivel egy szkriptnyelvről beszélünk, a hosszabb programok futtatásakor teljesítményvesztés jelentkezik. Hátrány még az információ titkosításának hiánya, valamint a dinamikus típusmegadások is (egy változóhoz több típus hozzárendelése).[2] Oktatási szempontból viszont kiemelendő a nyelv strukturális felépítettsége, ugyanis megköveteli a felhasználótól azt, hogy whitespace elemekkel tördelje a kódot. Itt ugyanis az egyes behúzások fogják jelölni a kódrészletek kezdetét és végét. Más nyelven az olvashatóság kérdése a programozóra van bízva, míg itt a nyelv megköveteli azt, hogy olvashatóan és „szépen" kódoljunk. [3] Ennek segítségével a diákokban rögzül az ilyesfajta kódolás, ami más nyelvek esetén is tökéletesen kamatoztatható.

A micro:bit kiválasztásánál két fő szempontot mérlegeltünk. Az egyik az oktatásban való kiváló hasznosíthatósága, a másik pedig az eszközök ára. A résztvevőknek ugyanis saját maguknak kellett az eszközöket beszerezni, amelyekkel részt tudtak venni a szakkörökön. Az eszköz megvásárlásán felül más költség nem merült fel a résztvevők számára, a foglalkozások ingyenesek voltak. Az eszköz sokoldalúságának köszönhetően más-más érdeklődésű gyermekek számára is alkalmas lehet és több műveltségi területbe integrálhatók és készíthetők a témához kapcsolódó programok. [4][5] Ezen

felül egy kompakt eszközről beszélhetünk, aminek kiterjesztése a kezdő felhasználók számára is könnyen abszolválható.

## 2. A szakkör tananyaga

A szakkör tananyaga nagy részben az alábbi két tananyagból épült fel:

- Programozzunk micro:biteket Python nyelven - Szűcs Norbert [6]
- Programozzunk micro:biteket - Abonyi-Tóth Andor [7]

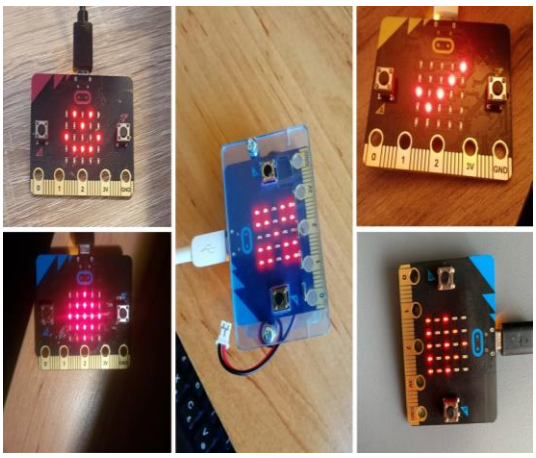
A fentebb említett segédanyagokat át kellett alakítani úgy, hogy illeszkedjenek a szakkör időtartamához és az alkalmak számához, amelyek 5x2 órás intervallumokban valósultak meg. A távoktatás miatt, az eszköz rádiófunkcióinak bemutatását teljes egészében kihagytuk az alkalmak során, hiszen ehhez több eszközre lett volna szüksége a diákoknak.

Az alábbiakban röviden kifejtjük az egyes alkalmak tartalmi részét. A teljes óravázlatokkal támogatott tananyagok a fenti segédanyagokban megtalálhatók. Az órákat igyekeztünk folyamatos önálló feladatokkal tűzdelni, amelyek beadása online történt. Ne felejtsük el, hogy ezeket a tananyagokat egyfajta vázlatnak használjuk és igyekezzünk mindig az adott kurzus résztvevőire szabni az óra menetét. Ahol külön nincs említve, hogy a blokkos tananyag került felhasználásra, ott alapértelmezettként a Python nyelven íródót használtuk fel.

**1. foglalkozás:**

A foglalkozás lényegi részét megelőzte egy úgynevezett installációs rész. Erre azért van szükség, mert a felület, amelyet a későbbiekben részletezünk, csak a legfrissebb firmware verzióval ellátott eszközöket fogja felismerni, illetve csak akkor lesznek elérhetők olyan funkciók, amelyek nagyban gyorsítják a munkafolyamatokat. Ezt könnyedén meg tudjuk valósítani, ha ellátogatunk a következő weboldalra: https://microbit.org/get-started/user-guide/firmware/.

Ezt követte az első programunk elkészítése, amely nem volt más, mint a „Hello World!" kiíratása. Az óra hátralévő részében a micro:bit kijelzőjére rajzoltunk ki alakzatokat és animációkat készítettünk. Önálló feladatként a saját képek készítése volt a gyerekekre bízva (1. ábra).



**1. ábra:** Az első napi önálló munka néhány eredménye

**2. foglalkozás:**

A második foglalkozást szintén a kijelző használatával kezdtük. Itt azonban már nem összefüggő képeket jelenítettünk meg, hanem a LED mátrix pontjait külön-külön címeztük meg. Itt előkerültek a véletlenszámok és azok informatikában betöltött szerepe. A gyermekek figyelmét na-

gyon jól meg lehet ragadni, ha kedvenc számítógépes játékaikat hozzuk példaként és ezen keresztül vezetjük be az új ismereteket. Az óra második felében, a blokkos tananyagból elérhető emeletes ház animációját implementáltuk Python nyelven. Emellett maradt időnk még egy manipulált dobókocka megvalósítására is. Megállapítható, hogy a játékos és érdekes megközelítése a véletlen számoknak, nagyban megkönnyíti, hogy a gyerekek elsajátíthassák az ismereteket és alkalmazni tudják azokat

Itt került sor az eseményvezérelt programozás alapjainak bemutatására a gombok használata által.

**3. foglalkozás:**

Ezen a foglalkozáson már a szenzoroké volt a főszerep. Az iránytű elkészítése után ugrásszámláló alkalmazást készítettünk el Python nyelven.

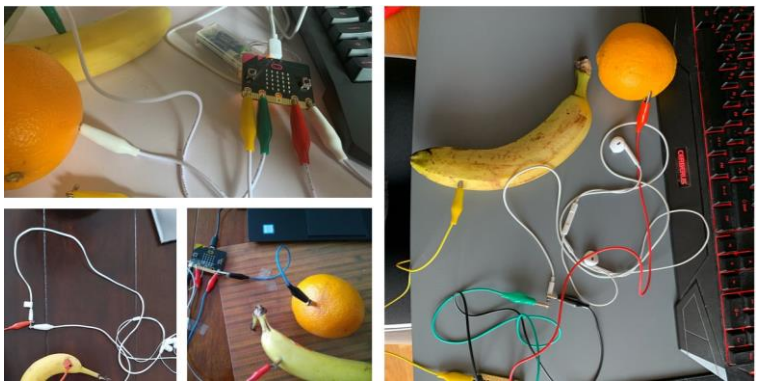
Áttekintettük, hogy a gombnyomásokon túl milyen egyéb módon adhatunk utasítást a micro:bitnek. Ennek keretei közt elkészítettük a saját 8-as golyót[1] megvalósító programunkat, ahol már szükség volt a lista adatszerkezetre és műveleteire. A megvalósítás lényegi része a diákok önálló feladata volt, miután megismerték az új műveletekhez tartozó parancsokat.

**4. foglalkozás:**

A negyedik foglalkozáson már barkácsolás jellegű feladatokat oldottunk meg. Nagy népszerűségnek örvendett ez az alkalom a gyermekek körében mind a három turnus alatt. Itt szükség volt az alábbi kiegészítőkre is:

- 4 db krokodilcsipesz kábellel/sok alufólia
- 2 db szög
- cserép és föld, ami lehetőleg száraz volt
- 1 banán
- 1 narancs

Az óra első felében egy gyümölcszongorát (2. ábra) készítettünk el, ahol saját magunk és a gyümölcsök segítségével kiterjesztettük a micro:bit áramkörét és ezáltal hangokat játszattunk le az eszközzel. Majd az önálló feladat egy sziréna elkészítése volt.

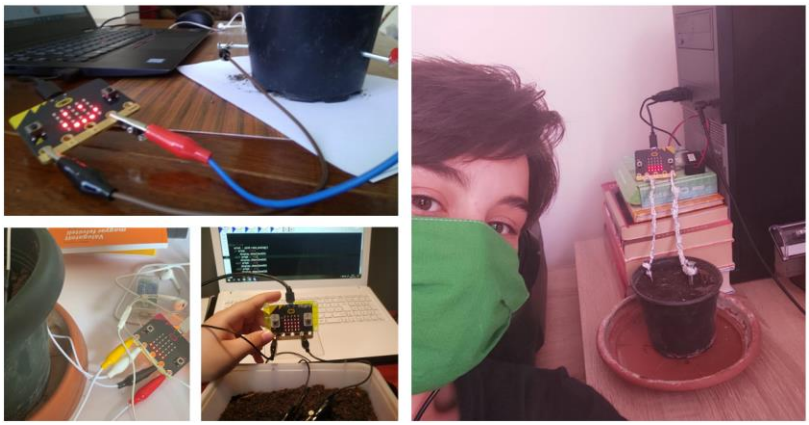


**2. ábra:** A gyerekek által készített gyümölcszongorák

[1] Mesékben és filmekben többször szerepelt a bűvös 8-as golyó, amely véletlenszerűen válaszol a feltett kérdéseinkre.

A foglalkozás második részében pedig egy okosotthon projektben is felhasználható nedvességmérőt (3. ábra) hoztunk létre, amely valós időben követi nyomon a talaj nedvességét azáltal, hogy figyeli annak ellenállását. Itt bemutatásra került egy kidolgozott, micro:bit segítségével vezérelt öntözőrendszer és kitértünk az okosotthonok felépítésére is.

Ez a feladat volt az, amely igazán nyomot hagyott a gyerekekben, valamint a fenntarthatóságot és energiatakarékosságot is figyelembe véve sokan továbbgondolták a feladatot és hozzáfogtak saját projektjük megvalósításához.



**3. ábra:** Az elkészült nedvességmérők

**5. foglalkozás:**

Az utolsó napon került sor a gyerekek által leginkább várt foglalkozás megtartására. Itt ugyanis a Flappy Bird játékot valósítottuk meg a micro:biteken. A játék elkészítése folyamán nagyban építettünk az eddig tanultakra, és szinte minden egyes tanult elem használatba került. A diákok a tanultakat most egy komplex program elkészítésére használhatták fel, és visszajelzést szerezhettek, hogy milyen szinten tudták elsajátítani ezeket az ismereteket.

A játék készítése során új ismeretként megjelent az objektumorientált programozás, valamint a diákok bepillantást nyertek a különböző fejlesztési fázisokba. A programot részekre bontva építettük fel, aminek egyes részeit önálló munkában kellett meghatározni.

# 3. A szakkör szoftverigénye, a virtuális környezet bemutatása

A szakkör folyamán több szoftvert is használtunk, annak érdekében, hogy a diákok számára megteremtsük az online oktatáshoz szükséges legideálisabb környezetet. Igyekeztünk olyan programokat választani ehhez, amelyek alkalmasak a szervezésre és a feladatkiadásra, az online órák megtartására, a szemléltetésre és természetesen a kódolásra és annak felügyeletére. Az alábbiakban ezeket a felületeket tekintjük át.

## 3.1. A virtuális tanterem és a foglalkozások

A lebonyolítás ezen részéhez a Google szolgáltatásait használtuk. A Classroom felületen lehetőség van csoportok szervezésére, a feladatok kiadására, valamint információkat tudunk megosztani a diákokkal. Tökéletesen használható hagyományos iskolai foglalkozások lebonyolítására is, mivel gyakorlatilag egy komplett LMS[2] rendszert kínál fel a Google számunkra.

[2] Learning Management Systems

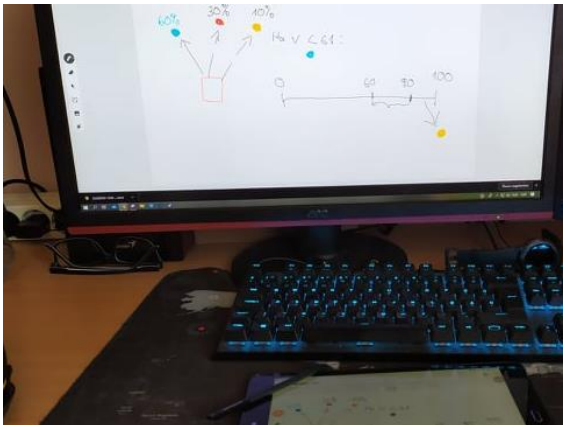
A Classromban, a csoport naptárjába beírhatjuk az órarendet, amelyhez egy konferenciabeszélgetést is társíthatunk a Meet alkalmazás segítségével. A Meet szintén rendelkezik az online találkozók lebonyolításához szükséges minden funkcióval. A gyerekek könnyedén indíthatnak képernyőmegosztást, így a tanár személyre szabott segítséget nyújthat számukra.

A kiválasztás szempontját több tényező befolyásolta. Olyan szoftvercsomagra volt szükségünk, amely nem csak az osztálytermi szerveződést, de az online órák tartását is lehetővé teszi. Fontos volt az is, hogy különböző fájltípusokat könnyedén integrálhassunk az oldalra, hiszen ki kellett adni a feladatokat, valamint a diákoknak fel kellett tölteniük képeket, videókat. Olyan rendszerre volt szükségünk, amelyet a diákok nagy része ismer és könnyen kezel és elérhető számukra. A Google szoftvercsomag ezeket a kritériumokat teljes mértékben abszolválta, mindemellett az eléréséhez mindösszesen egy e-mail címre van csak szükség. Végső soron pedig a létszámkorláttal sem volt probléma, hiszen 100 főig ingyenes a Meet szolgáltatás is.

Amennyiben olyan diákok szeretnék használni a Google fiókjait, akik nem múltak el 16 évesek, akkor a szülő email címével történő bejelentkezés a nem intézményi szintű esetekben megoldás lehet. Intézmények számára azonban lehetőség van a G-Suite rendszer igénylésére, amely által egységes email cím adható a diákok számára, valamint ezek a fiókok felügyelhetők is a rendszergazdai jogosultsággal rendelkezők által. Kiemelendő még a Google szoftvercsomagnál, hogy tökéletesen használható többféle platformon is, ezért a diákoknak nem feltétlenül van szükségük számítógépre.

## 3.2. A virtuális tábla

Külön kiemelnénk egy lehetőséget az online tartott órák esetében a táblai rajzok kiváltására. Úgy gondoljuk a programozás tanítása folyamán a táblarajz egy nélkülözhetetlen pedagógiai szemléltető eszköz, amelynek segítségével a diákok könnyebben értik meg az absztrakciós szinten végbemenő folyamatokat, ábrázolásokat.



**4. ábra:** A véletlenszámok szemléltetése a Jamboard segítségével

Esetünkben ehhez is a Google egyik alkalmazását használtuk, méghozzá a Jamboardot. Ezen kívül pedig egy tablet volt a segítségünkre, amelyhez tartozik toll is. Ez helyettesíthető egyszerűbb rajztáblával, ha pedig nincs kéznél tablet az eszköztárunkban, akkor az egérrel is rajzolhatunk. A Jamboard azért lehet jó választás, mivel a virtuális füzetünk megosztható másokkal és így a diákok egyrészt megkaphatják a jegyzeteket, másrészt közösen is hozhatnak létre produktumokat.

A szakkör folyamán megnyitottuk a Jamboard alkalmazást a tanári gépen és a résztvevők a megosztott képernyőn láthatták, hogy mit rajzolunk a táblagépen (4. ábra). Ez a megoldás, hogy nem ők nyitnak meg még egy alkalmazást, hanem csak kivetítve látják az eredményt, azért szerencsésebb megoldás, hogy ne terheljük le túlságosan az eszközüket, hiszen nem egységes erősségű hardverekkel vannak felszerelkezve a résztvevők.

## 3.3. Az online programozási felület

Nem beszéltünk még a programozást megvalósító felületről. Az online programozás és a közös dokumentum szerkesztés nem újkeletű dolog. A jól bevált oldalak, mint például a repl.it helyett

azonban választásunk egy olyan oldalra esett, amely igaz még csak béta fázisban van, de dedikáltan a micro:bithez készült. Ez nem más, mint a micro:bit classroom.[8]

Az oldalon nem csak Pythonban, hanem blokk alapú környezetben is programozhatunk. A nyelv kiválasztása után a weboldal generál egy ikonokból és egy pinkódból álló belépési jelszót. Ezzel biztosítva van, hogy illetéktelenek ne lépjenek be a csoportba.

A cikkben a weboldal kódolással foglalkozó részének (5. ábra) bemutatására fókuszálunk és körbejárjuk annak lehetőségeit. A felület tökéletesen használható az online oktatás területén. A kiemelkedő hasznosíthatósága abban rejlik, hogy olyan megoldásokat nyújt, amelyekkel a diákok számára egyszerűbbé teszi a kapcsolódást az eszköz és a gép között, a tanárok számára pedig jó lehetőségeket nyújt a távolság áthidalására. Ezeket a tulajdonságokat az alábbiakban részletezni fogjuk.


**5. ábra:** A micro:bit classrom kezelőfelülete

Magával a kódszerkesztővel különösebben nem kívánunk foglalkozni, hiszen az gyakorlatilag egy szokványos Python IDLE. Ez talán az egyetlen hátránya a környezetnek, hogy nem található benne semmiféle ellenőrző rendszer, mint például egy Visual Studio-ban, vagy a már korábban említett repl.it oldalon. Kapunk tehát egy szövegszerkesztőt, amely a kódot vizuálisan tagolja, de ebben ki is merül az oldal ezen funkciója.

A kész kódunkat le is tölthetjük, azonban, ha csatlakoztattunk micro:bitet akkor a fájlműveletek mindegyike kihagyható. Egy egyszerű kattintással rá tudjuk tölteni közvetlenül a csatlakoztatott eszközre a kódunkat. Tapasztalataink alapján ez nagyban gyorsította a feltöltés sebességét a klasszikus szakkörökhöz képest, ahol másolással és beillesztéssel kellett az eszközre juttatni a kódot. Az eszköz csatlakoztatását a korábban leírtak szerint kell megvalósítani.

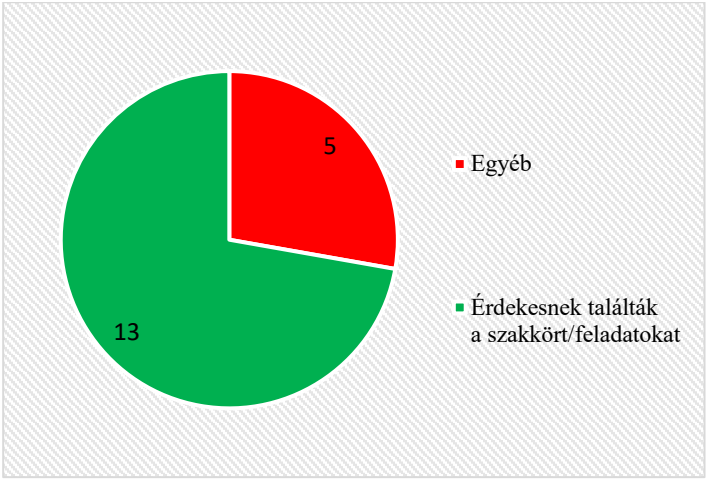
Tanárként lehetőségünk van a tanteremben lévő összes diák kódját megtekinteni. Ez az egyik nagy előnye a felületnek. Egy kattintással kiválaszthatjuk, hogy melyik diák kódját szeretnénk látni és könnyedén tudunk számára segítséget nyújtani. Nincs szükség képernyőmegosztásra és egyéb beállítás alkalmazására. Másik nagy előny tanári szempontból, hogy a mi általunk írt kódot egy kattintással át tudjuk küldeni az összes, vagy az általunk kiválasztott diák számára. Erről a diákok értesítést kapnak és elfogadhatják a kódunkat, vagy folytathatják a munkát a sajátjukkal. Keretprogramok kiadására rendkívül jól hasznosítható. Ezeken felül lehetőségünk van az összes kódról egy dokumentumot készíteni és lementeni azt. Ez a dokumentum minden diák kódját tartalmazni fogja és tagolja azokat számunkra.

A távolság és kódjavítási lehetőségek messze túlmutatnak az egyszerű szerkesztő hátrányain. A felület kifejezetten micro:bithez készült, valamint a fent említett pozitív tulajdonságok összességét figyelembe véve megállapítható, hogy egy szinte tökéletes koncepció megvalósulását láthatjuk a micro:bit classroom oldalán. A béta verzió szakaszhoz képest rendkívül kevés hiba volt tapasztalható. Legtöbb esetben ez kimerült abban, hogy a kódrészlet első átküldésre nem érkezett meg, de ezt egy újraküldés gyorsan orvosolta. Reméljük a jövőbeni fejlesztések erősítenek magán a szerkesztőn is és akkor ténylegesen egy hibátlan felületről beszélhetünk majd. Oktatási szempontból most is egy kiváló környezetről van szó, amely a jelen helyzetben lehetővé teszi a robotika online térbe való áthelyezését.

## 4. Visszajelzések és záró gondolatok

A tábor végeztével egy rövid kérdőív is kiküldésre került, amelyben a résztvevők leírhatták, hogy mi tetszett és mi nem tetszett részükről a szakkörben valamint, hogy ki venne még részt és milyen foglalkozáson. A kérdőíveket a 42 diákból 18 töltötte ki.

A kérdőívben megadott válaszokat tekintve a legfőbb pozitívum az érdekes feladatok bemutatására vonatkozott (6. ábra). Ide sorolhatók azok is, akiknek a konkrét játékkészítés tetszett legjobban. Az egyéb kategóriában 3 ember a „minden tetszett" választ adta, 1 azt emelte ki, hogy megtanult programozni, 1 pedig, hogy a vírus alatt is sikerült megvalósítani az interaktív szakkört.



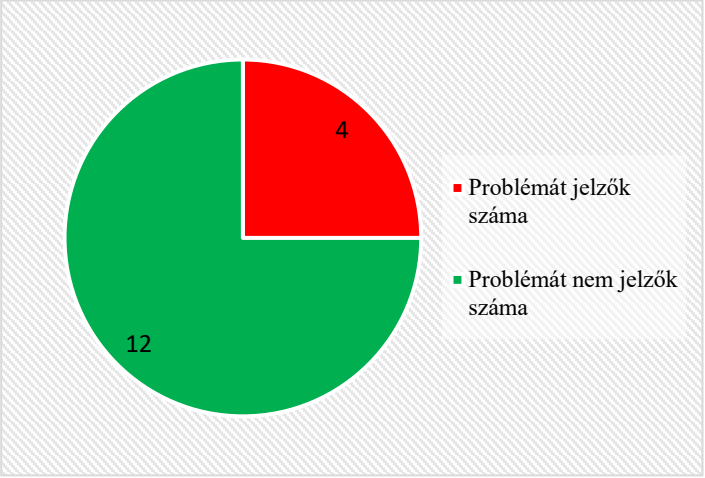

**6. ábra:** A pozitívumok megoszlása a szakkör kapcsán

Ahhoz, hogy teljes képet kapjunk, a válaszokat az alábbiakban közöljük.

- *Az új dolgok készítése. Legjobban a Flappy Bird és az egyszerű Tamagotchi csinálás, amit én tovább fejlesztek.*
- *Hiába programoznak a gyerekek már évek óta más nyelveken, még a nagyobbnak is sikerült újdonságokat mutatni. Az oktató kedves és türelmes volt és felkészült, sok más programozó kurzus oktatóival ellentétben. Ha lenne ilyen lehetőség a következő tanév során, szívesen venne részt rajta a nagyobbik fiam, aki programozó szeretne lenni. Jó, hogy a jelen helyzetben ez online zajlott, s ez máskor is egy időtakarékos megoldás (nem megy utazgatással az idő).*
- *Hogy a vírus alatt is meg tudtak csinálni ilyen interaktívan*
- *Az összes feladat tetszett, mert érdekesek voltak és egymásra épültek. Az oktató, Bence nagyon türelmesen és segítőkészen prezentálta a feladatokat. A közösség is nagyon jó volt.*
- *Minden tetszett*

- *Az, amikor leprogramoztuk a FLUPPY BIRD-et ,mert olyan fura ,hogy egy az egyben ugyan az, mint amit a playáruházból le lehet tölteni.*
- *Minden. Az oktató nagyon kedves és türelmes volt.*
- *Minden tetszett. Az oktatást szisztematikusan felépítették, átadták a szükséges ismereteket. Az oktató segített, hogy mindenki követni tudja a feladatokat. A feladatok, kísérletek érdekesek is voltak.*
- *Nagyon jók voltak a feladatok. Érdekes volt a gyümölcszongora, tetszett, hogy barkácsoltunk is.*
- *Nekem a Flappy Bird készítése tetszett a legjobban, mert jó volt elmélyülni a gondolatmenetében.*
- *hogy megtanultam programozni*
- *Minden tetszett mert érdekes feladatok voltak meg jó volt a közösség*
- *Sokat tanultam, megismerkedtem a Python programozási nyelvvel. Tetszettek a feladatok, jó hangulatban teltek a foglalkozások. Szimpatikus volt a tanfolyamvezető (Bence) hozzáállása. Szívesen vennék részt egy hosszabb (2 hetes) kurzuson is.*
- *Minden tetszett :)*
- *Az oktatás követhetősége és érdekessége.*
- *Érdekesek voltak a feladatok, és délután is foglalkoztam vele, mert érdekelt :)*
- *Igazából minden! Leginkább azok a feladatok, amikor már bonyolultabb programokat írtunk és a tanultakat ötvöztük. Valamint, hogy olyan programokat készítettünk, ami egy okos otthonhoz tartozhat és ez lenyűgözött, hogy ezeket mi megcsináltuk.*
- *Új dolgokat tanultam, érdekes témák voltak, Bence nagyon közvetlen volt, minden kérdésre válaszolt.*

A kitöltők közül mindösszesen 4 ember válaszolt a *mi nem tetszett* kérdésre (7. ábra). 1 esetben a tempó volt lassú, 1 esetben túl gyors. A maradék két eset közül az egyiknél a videók készítése kapcsán merültek fel problémák, míg a másik esetben a barkácsoláshoz nem sikerült időben felszerelést biztosítani.




**7. ábra:** A problémák felmerülésének megoszlása

A kérdőívből még kiderült, hogy 12 résztvevő foglalkozna újra programozással, ami nem feltétlenül csak micro:bithez köthető. 5 esetben nincs adatunk, 1 ember pedig nem foglalkozna az informatikával a továbbiakban.

A kevés kitöltés ellenére, a válaszokat látva úgy gondoljuk, hogy a robotika a diákok számára platformtól és minden mástól függetlenül is egy érdekes téma. Segítségével, még a digitális oktatás keretei között is megfelelően lehet motiválni a gyermekeket és felkelteni az érdeklődésüket a programozás iránt. A kevés kitöltés valószínűleg betudható a nyári időszaknak és az e-mailben való meg-

keresésnek, hiszen biztosak vagyunk abban, hogy ha az utolsó alkalom keretei között élőben kérdezzük meg a diákokat, akkor sokkal több kitöltés született volna. Legközelebb ezért inkább ezt a módszert választjuk.

A szakkör a lemorzsolódást tekintve is sikeresnek mondható, hiszen bárki számára adott volt a lehetőség, hogy ne jöjjön a következő alkalomra, de ezzel szerencsére nem élt senki. A résztvevők a szakkör alatt jól érezték magukat és talán a jelenlegi helyzet ellenére is sikerült egy kicsit összekovácsolni őket. Több példa volt arra, hogy a diákok egymást megkeresték a délutánok folyamán is, közös tanulás vagy játék kapcsán.

Az eszköz felől megközelítve a dolgot, jobban ki lehet aknázni az eszköz funkcióit, ha egy klaszszikus tantermi órán veszünk részt. Ennek ellenére nem kellene elvetni az online szakkörök tartását sem, hiszen ezáltal a távolságot leküzdve (8. ábra) olyan diákok is esélyt kaptak megismerkedni a robotikával, akik normál esetben szinte biztosan nem tudtak volna a szakkörökre járni. Úgy gondoljuk, hogy az online szakkörök segítségével több gyermekhez tudunk elérni, ezzel átadni nekik azt a tudást, amiből kiindulva már saját maga is fel tudja fedezni a programozás világát és az adott eszköz funkcióit.


**8. ábra:** A T@T Kuckó összes szakkörén részt vevők megoszlása a térképen [9]

## Irodalom


1. John M. Zelle [1999]: Python as a First Language
2. Linda Grandell, Mia Peltomäki, Ralph-Johan Back and Tapio Salakoski [2006]: Why Complicate Things? Introducing Programming in High School Using Python
3. Dr. Toby Donaldson [2003]: *Python as a First Programming Language for Everyone*
4. Dr. Abonyi-Tóth Andor[2018]: A micro:bitek felhasználási lehetőségei az oktatásban, InfoDidact 2018, https://people.inf.elte.hu/szlavi/InfoDidact18/Manuscripts/ATA.pdf (utoljára megtekintve: 2020. 10. 10.)
5. Gaál Bence[2019]: A robotika témakör integrálásának lehetőségei a természettudományos tantárgyak oktatásában, InfoDidact 2019 https://people.inf.elte.hu/szlavi/InfoDidact19/Manuscripts/GB.pdf (utoljára megtekintve: 2020. 10. 10.)

6. Szűcs Norbert: Programozzunk micro:biteket Python nyelven! (2019) http://microbit.inf.elte.hu/wp-content/uploads/2019/10/microbit_python_szakkor_szucsnorbert.zip (utoljára megtekintve: 2020.10.19.)
7. Abonyi-Tóth Andor: Programozzunk micro:biteket! (2018) http://microbit.inf.elte.hu/wp-content/uploads/2018/05/Programozzunk-microbiteket-2018.pdf (utoljára megtekintve: 2020.10.10.)
8. A micro:bit classroom- Beta hivatalos weboldala: https://classroom.microbit.org/ (utoljára megtekintve: 2020.10.19.)
9. T@T Kuckó- Nyári online alkotótábor(2020): https://sites.google.com/view/elte-kucko-alkoto/alkot%C3%B3t%C3%A1bor-2020?authuser=0 (utoljára megtekintve: 2020.10.19.)

# Szituációs gyakorlatok kidolgozása a Scrum oktatásához

Gulyás Gergő

gery0704@inf.elte.hu
ELTE IK

**Absztrakt**: A Scrum napjaink egyik közkedvelt agilis módszertana, amelyet főként a szoftverfejlesztés területén alkalmaznak, de egyre több iparágban terjed. A módszertannal való ismerkedés, tanulás folyamatát nagymértékben megkönnyíti a valós szituációk megismerése. Szituációs gyakorlatok során újabb képességeket sajátíthat el és fejlesztheti a résztvevő, akár kezdő még a módszertan ismereteivel kapcsolatban, akár haladó. Ebben a cikkben ipari környezetből származó, valós esetekre alapozó szituációs gyakorlatok kidolgozási módját mutatjuk be és példákon keresztül illusztráljuk.

**Kulcsszavak**: Scrum, Scrum Master, Agilis, Szituációs gyakorlat, Oktatás

## 1. Bevezetés

A mai ipari szférában, versenyhelyzetben kifejezetten fontos, hogy a cégek milyen reakcióidővel tudnak alkalmazkodni a gyorsan változó és fejlődő trendekhez, illetve az üzleti igényekhez. Kiemelten fontos életükben a különböző erőforrások minél optimálisabb kihasználása, a munkaerő gyors, precíz csoportosítása a hatékonyság megtartásával. A megrendelők ma már gyorsan szeretnének használható terméket kapni, későbbi új funkciók bevezetésével, a korábbi teljes funkcionalitást lefedő, de hosszú hónapok alatt szállított termékek helyett. Ezekre az igényekre volt válasz az agilis módszertanok kialakulása, amelyek segítségével a cégek hatékonyabban igazodhatnak a változó piaci feltételekhez. Napjainkban a cégek nagy százaléka használ valamilyen agilis módszertant a működése során.[2]

A Scrum napjaink talán legkedveltebb agilis módszertana, rengeteg alternatív módszertan is kifejlődött belőle, ami jól mutatja, mennyire meghatározó. A State of Agile 2020. évi statisztikái szerint a megkérdezett cégek 95%-a használ valamilyen szinten Scrumot. [2] Természetesen ez nem jelenti, hogy a cég összes csapata a Scrum szerint dolgozik, 44% esetében kevesebb, mint a csapataik fele alkalmazza a Scrumot. Kifejlesztői, Ken Schwaber és Jeff Sutherland alapelvként tűzték ki, hogy a keretrendszer legyen egyszerű, könnyen érthető [1]. A Scrum a fejlesztés időtartamát sprintekre, azaz munkaszakaszokra osztja, minden sprint célja egy működő kód, inkrementum elkészítése, így a sprint végén a megrendelő mindig új funkciókkal működő verziót kaphat a termékéből. Egy sprint hossza maximum 4 hét, így a szállító gyakori visszajelzést kap a megrendelőktől a sprint végi bemutatókon arra vonatkozóan, hogy mennyire elégedettek a termék aktuális állapotával.

A Scrum módszertan alkalmazása során a munkavállalók csapatokba rendeződnek a Scrum Guideban leírtaknak megfelelő szerepkörökben és létszámban. A csapat tagja a Scrum Master, aki a csapat szolgáló-vezetője, felelős a Scrum megértéséért, betartásáért, támogatja a csapat munkáját. Célja a csapata munkájának és a csapat által létrehozott értéknek a maximalizálása [1]. A Scrum Master mellett a csapat része még a Product Owner (termékgazda, röviden PO) aki a megrendelőt képviseli a projektben, akár a megrendelő is delegálhatja [a projektbe]. Mellettük a csapatot [még] a fejlesztők és tesztelők alkotják, ideálisan a csapat 6-8 fős. Kutatásunk központjában a Scrum Masterek szerepeltek, általuk igyekeztünk a Scrum csapatok működését és dinamikáját a lehető legjobban körbejárni.

Fontosnak tartjuk a Scrum oktatását, mivel a cégek 95%-a [2] alkalmaz valamilyen Scrumhoz köthető módszert. A legtöbben a Scrum technikáin belül a napi standup meetinget alkalmazzák (85%) és a retrospektív megbeszélések alkalmazását (81%) [2]. A legismertebb Scrummal foglalkozó szervezetek nemzetközileg a Scrum.org [3], illetve a Scrum Alliance [4]. Ezek a szervezetek saját kurzusokat is szerveznek, amelyek nemzetközileg elismert [5] igazolásokat adnak a képzést elvégző tanulóknak. Az egyetemet elvégzett hallgatók munkahelyeiken valószínűleg találkoznak a Scrum valamilyen formájával, így a Scrummal való ismerkedés és a különböző szituációk megismerése mindenképpen hasznukra lesz majd a munkaerőpiacon is.

A kutatás során célunk volt valós ipari szituációk megismerése és ezeknek az oktatásba történő beintegrálása. A kutatás első szakaszában kérdőív segítségével gyűjtöttünk adatokat. Ezeket a kérdőíveket célzottan készítettük, kitöltőik Scrum Masterek. A munka második szakaszában interjúk formájában valósult meg az adatgyűjtés. Ezt követően az adatok feldolgozása jutott sorra, majd ezek alapján olyan szituációs gyakorlatok kidolgozása, melyek Scrum oktatása során használhatóak.

Határozottan az egyik legnehezebb feladatot a kapcsolatfelvétel, Scrum témán belüli ismeretségek, kapcsolatok kiépítése jelentette. Kiemelten fontos volt, hogy jelenleg is aktívan Scrum Masterként dolgozó, vagy hosszú ideig ebben a szerepkörben tevékenykedő korábbi Scrum Masterek legyenek a kérdőív kitöltői, illetve interjúalanyok, ezáltal is biztosítva, hogy valós ipari szituációkról kapjunk információkat, tapasztalatokat. A kutatás során megismert Scrum Masterek 75%-a jelezte, hogy szívesen együttműködne a kutatás további részében is. Munkánk egyik pozitív következménye tehát, hogy elkezdett kialakulni egy kapcsolatbázis, amelyre számíthatunk a későbbiekben. A megkérdezettek 75%-a azt is jelezte, hogy szívesen hallana a kutatás eredményeiről, ami azt jelenti, hogy az aktívan dolgozó Scrum Masterek is fontos, érdekes ötletnek tartják a szituációs gyakorlatok kidolgozását.

A cikk további részében bemutatjuk, milyen módszerekkel igyekeztünk alapanyagot gyűjteni a szituációs gyakorlatokhoz. Először is szeretnénk néhány kapcsolódó területet bemutatni a 2. fejezetben. A 3. fejezetben először a Scrum Mastereknek készült kérdőívünk felépítését és eredményeit tárgyaljuk. Ezt követően a Scrum Masterekkel készült interjúk kerülnek bemutatásra, a fejezet végén pedig megvizsgáljuk egy jó szituációs gyakorlat jellemzőit. A 4. fejezetben bemutatjuk egy elkészült szituációs gyakorlatunkat és hogy az egyes elemei milyen célt szolgálnak. További terveinkről a 5. fejezetben számolunk be.

## 2. Kapcsolódó munkák

A hallgatók számára mindenképpen előnyt jelent, ha a munkaerőpiacra kerülve a lehető legtöbb tapasztalattal rendelkeznek. A [7] cikkben láthatjuk, hogy a lexikális tudáson alapuló oktatással szemben, a probléma alapú oktatásban (Problem Based Learning, PBL) résztvevők jobb eredményeket érnek el, amellett, hogy a tapasztalataikat is gyarapítják a problémamegoldások során. Ezért is igyekeztünk a szituációs gyakorlatokat valós ipari környezetben megtörtént szituációk alapján felépíteni. A módszer hatékonyságáról és alkalmazásának módszeréről olvashatunk Szögedi Ildikó doktori értekezésében is [13]

A probléma alapú tanítási módszer, amely fejleszti a résztvevők problémamegoldó képességét, előtérbe kerülése már korábban megfigyelhető volt és ennek a szemléletnek az egyre szélesebb körben való elterjedéséről tájékozódhatunk a [11] cikkben.

A probléma alapú oktatás eredete alapvetően egészségügyi területekről ered, de mára már sok finomításon és optimalizáláson esett át a módszer. A PBL fejlődését és kialakulását összefoglalta számomra David Boud és Grahame E Feletti – The Challenge of Problem-Based Learning c. könyve [12] A módszer lényegét és hatékonyságát érthetőbbé teszi fejlődésének és kialakulásának ismerete véleményem szerint, így új nézőpontokat nyújt számunkra, ha ezeket a tényezőket is megismerjük.

Fontos a szituációs gyakorlatok során, hogy a hallgatók közül ki tölti be az egyes szerepeket, esetünkben kiemelten fontos a Scrum Master szerepet betöltő személyisége, elhivatottsága és képességei. Ezeket a nehézségeket bemutatja a [8] cikk, továbbá megfigyelték, hogy a Scrum Master szerepét betöltő hallgató a félév során sok tapasztalatot szerzett és sokat fejlődött. Szeretnénk munkánk során olyan szituációs gyakorlatok létrehozásának menetét bemutatni, amelyek a Scrum Mastert helyezik a központi szerepbe, így a tapasztalat szerzése és a fejlődése még inkább hatékony lehet.

A Scrum oktatását egyetemi környezetben már több esetben is kutatták korábban. A [9] cikkben egy kurzus keretében került a Scrum módszertan alkalmazásra, de a Scrum Master szerepet javarészt az oktatók gyakorolták. Kutatás-fejlesztés labor keretein belül is alkalmazták a Scrum módszertant, ez a [10] cikkben részletesen tárgyalva van. Ez esetben a Scrum Master szerepkört is a résztvevők töltötték be.

Elmondhatjuk tehát, hogy az oktatás során alkalmazott Scrum módszertan esetében a Scrum Master szerep a korábbi munkákban is már egy sarkalatos pont volt. Egyes esetekben a résztvevők, máskor az oktatók töltötték be ezt a szerepkört, viszont a hallgatókra jó hatással volt a szerepkör és fejlődtek ennek hatására. Szeretnénk, ha szituációs gyakorlataink segítségével a hallgatók Scrum Master képességei hasonlóan fejlődnének a szituációk ipari környezetből merített alapjai segítségével.

# 3. Kutatási módszer

## 3.1. Kérdőív és általa gyűjtött információk

A Scrum Masterek motivációival, munkájával foglalkozó kérdőívünk év elején készült el. A kérdőív 32 kérdést tartalmazott, melyek közül 16 kérdés hosszú kifejtést igényelt, így a kitöltésének ideje az átlagos kérdőívekhez képest jóval hosszabb volt, körülbelül 50 perc. A hosszú, összetett felépítés sok kitöltőt elriaszt, ezért a kitöltők száma alacsony volt, mindössze 8. A kevés kitöltés ellenére, a sok kérdés okán hasznos új információkat szolgáltatott a kérdőív.

A 3 részre bontott kérdőívünk az alábbi felépítéssel rendelkezett:

- A Scrum Masterre és munkakörére vonatkozó kérdések

Az első szakaszban igyekeztünk megtudni, a Scrum Master hogyan látja feladatkörét és milyen tapasztalatokkal rendelkezik már ezt a szerepkört illetően. A kitöltő Scrumról alkotott nézeteit is ebben a szakaszban mértük fel. Néhány példa kérdés:

  - Hogyan írná le a saját feladatkörét pár szóban?
  - Miért lett Scrum Master?
  - Mely tulajdonságokban érezte a legtöbb hiányosságot a saját képességeiben pályája elején?

- A Scrum Master csapatára, csapata munkájára irányuló kérdések

A kérdőív második szakaszában a Scrum Master csapatára került a fókusz. Szerettük volna megtudni, hogy a csapat hogyan viselkedik egy-egy hétköznapi szituációban, illetve milyen képességekkel rendelkezik egy átlagos Scrum csapat. A szakasz néhány kérdése:

  - Mennyire hasznosak a csapata számára az egyes meeting típusok? (meeting típusok felsorolva)
  - Melyek azok a témakörök, amelyekben erősítené a csapata tudását?
  - Milyen eszközökkel fejlesztené a szükséges területeket?

- A Scrumról alkotott általános meglátásokkal kapcsolatos kérdések

A kérdőív utolsó szakaszában kicsit elvonatkoztatva a Scrum Master jelenlegi munkájától és csapatától, az agilitásról és a Scrumról alkotott általános gondolatait szerettük volna megismerni. Igyekeztünk a Scrum Master motivációit megvizsgálni. Kérdések a szakaszból:

- Hogyan fogalmazná meg az Ön filozófiáját a Scrumról?
- Magyarázza el az agilitás alapjait a saját szavaival!

Általánosságban elmondható a válaszokból, hogy minden Scrum Master többször is hivatkozott a Scrum Guidera [1], mint a Scrum alapjaira. Elmondható tehát, hogy a Scrum Guide a mai napig megfelelő alapot ad a módszertan működéséhez, a Scrum fejlődését követi és nem elavult – tehát bátran építhető rá tananyag is. Volt, aki a saját feladatkörének leírásában említette, kiemelve, hogy a Scrum Guideban leírtaknak kell megfelelnie teljesen: „Precíz megfogalmazás a Scrum Guide-ban van, az baj, ha valaki nem úgy írja le a szerepkörét, mint amit ott leírnak". A válaszok alapján A Scrum Masterek munkájuk során többször is visszanyúlnak a Scrum Guidehoz, például arra a kérdésünkre, hogy mit tesznek, ha nem tudják mi lenne a Scrumnak megfelelő döntés, ilyen válasz is érkezett: „Elolvasom a Scrum Guide-ot".

A kitöltők mindegyike vallotta, hogy törekszik a folyamatos megújulásra, aminek nem csak az új irányok és trendek követése a lényege. Kiemelten fontos a hatékonyság magasan tartásának, növelésének érdekében is, hogy a Scrum Master ne unalmas, ismétlődő módszereket alkalmazzon. Egy Scrum Master például így beszélt a retrospektív meetingjeiről: „Mint Scrum trainer folyamatosan figyelem a tool-ok és módszerek megjelenését, ezt próbálom a saját retroimba is belecsempészni...". Volt olyan kitöltő, aki személyes, négyszemközti beszélgetéseket is szervezett a csapattagjaival rendszeresen, hogy így újabb fórumon tudjanak jelezni neki az esetleges problémákról. Ez a kitöltő emellett kiemelte azt is, hogy természetesen a módszerek is csapatfüggőek, nincs minden helyzetben használható általános módszer.

A kérdőívek alapján a Scrum Masterektől megtudtuk, melyek azok a területek, amelyekre a legnagyobb figyelmet szükséges fordítani (a csapat hatékony működése érdekében) az oktatásban is: A Scrum Masterek csapatukban leginkább a szerepkörök (Product Owner, Scrum Master) konkretizálását, a backlogok (Product, Sprint) kezelésének módját és a csoportdinamikát fejlesztenék.

A válaszok nagy segítségünkre voltak később a szituációk szereplőinek kialakításánál, és a csapatok működési mechanizmusainak felvázolásában.

## 3.2. Interjúk és általuk szerzett információk

Az interjúk során különböző háttérrel rendelkező Scrum Masterekkel történt beszélgetés, akik különböző cégeknél dolgoznak jelenleg, így több cég működésébe és hétköznapjaiba nyerhettünk betekintést.

A beszélgetések fókuszában mindvégig a Scrum Masterek által megélt szituációk álltak, amelyek kiválóan szemléltették a Scrum Master szerepét, a Scrum működését csapaton belül és nagyobb kontextusban a cégen belül is. Az interjúk online beszélgetés formájában készültek, nagyjából 30-40 percig tartottak személyenként. Az interjú elején közöltük a Scrum Masterekkel, hogy olyan eseteket szeretnénk megismerni, amelyekből szituációs gyakorlatot készíthetünk, illetve szeretnénk, ha a Scrum Master és annak döntése lenne fókuszban a szituációkban. Az interjúalanyok ezután kifejtettek 2-3 szituációt eddigi tapasztalataik alapján. Természetesen előfordult, hogy egy szituáción hamarabb továbbugrottunk, mert céljainknak nem volt megfelelő, de olyan is, hogy egy ígéretes szituációt részletesen megbeszéltünk, így akár több szituációs gyakorlat is kialakítható volt egy-egy elmesélt esetből.

Az elmondások alapján sok ipari szereplő előnyben részesíti a személyközpontú területekről, emberi működéssel foglalkozó képzésekből (pl. pszichológia) érkező Scrum Mastereket. Ennek oka több interjúalany szerint is, hogy fontos, hogy a Scrum Master az „emberekhez értsen" és a szemé-

lyeket tudja megfelelően kezelni, akár a motiváltság, mentorálás és csapatmorál kérdésében. Egyes cégek például kifejezetten szeretnek pszichológus végzetséggel rendelkező Scrum Mastereket alkalmazni. Megfigyelhető tehát, hogy akár a folyamatok optimalizálásának egyik legfontosabb pontja is lehet, hogy az emberi tényezők kezelését erre szakosodott Scrum Masterekre bízza a cég. A technikai döntéseket, mentorálást és műszaki segítségnyújtást egy architect vagy senior is végezheti ebben az esetben.

A beszélgetések során az intejúalanyok szinte mindegyikével érintettük az első lépéseket a csapataiknál, legyen szó életük első saját csapatáról pályakezdőként, vagy tapasztalattal rendelkezve a sokadik csapatukról. A megkérdezettek egyetértettek abban, hogy fontos, hogy minél gyorsabban mutassanak sikereket, eredményeket az új csapatnak, amely így motiváltan és bizakodóan tud fordulni az új Scrum Masterrel megélt változások felé. Fontos szempont tehát az agilis működés optimalizálása szempontjából, hogy a Scrum Master és csapata között amilyen gyorsan csak lehet, kialakuljon a bizalmi kapcsolat.

Természetesen nincs két egyforma Scrum Master, két egyforma csapat vagy két egyforma szervezet, így nehéz lenne az ideális működéshez vezető biztos receptet leírni. A legjobban javaslatokkal, követendő példákkal tudnak segíteni a Scrum Masterek.

A beszélgetések során több szituációt is részletesen átbeszéltünk és elemeztünk a Scrum Masterekkel, végig fókuszban tartva a jó szituációs gyakorlat kialakításához szükséges szempontokat. A Scrum Masterek tapasztalatai alapján sikerült azonosítani megfelelően általános, több cégnél is tapasztalt hibákat, amelyekből tanulhatnak a hallgatók A jó Scrum Masterek tipikus személyiségjegyeiről is gyűjtöttünk információt, amely ugyancsak felhasználható szituációs gyakorlatok kifejlesztéséhez.

Az interjúk során a Scrum Masterek beszéltek nekünk: csapaton belüli konfliktusok kezeléséről és hogy ilyenkor mennyire fontos a Scrum Master konfliktuskezelési képessége; számít, hogy a Scrum Master meg tudja értetni miért fontos a Scrum (Ezt egy olyan megismert szituáció illusztrálja, mely során a nagy tapasztalattal rendelkező senior fejlesztő nem szeretett volna a Scrummal foglalkozni, csak egyedül-zavartalanul-megszokott módon tenni a dolgát); van amikor a Scrum Masternek meg kell védenie a csapatát és annak ideális működését (Példa megismert szituációra, amikor a product owner nem látta el megfelelően a dolgát, és így a csapat nem tudott megfelelően dolgozni.)

Látható tehát, hogy az interjúk során sikerült a Scrum Master tulajdonságait széles körben lefedő szituációkat megismernünk, amelyek a csapat és a Scrum Master munkájának több szakaszát, szempontját is érintik.

## 3.3. Kutatási módszerek elemzése

A szituációs gyakorlatok megalkotásának folyamatában egyértelműen kulcsszerepe volt a kérdőívnek és az interjúknak is. A kérdőív segítségével megfelelő alapinformációkat szerezhettünk egy Scrum Master munkájáról és egy Scrum csapat működéséről. Sikerült beazonosítani, hogy melyek azok a sarokpontok egy Scrum csapat mindennapi életében, amikor lényeges szituációk alakulhatnak ki és melyek a gyakran előforduló szituációk. Ezeknek az alapinformációknak a segítségével az interjúk készítésekor sokkal hatékonyabb és fókuszáltabb kérdéseket sikerült feltenni a Scrum Mastereknek.

A kérdőív során a kifejtős kérdéseknél természetesen fontos a kérdések pontos megfogalmazása, hogy biztosan arra kapjunk választ, amire szeretnénk. Ezzel szemben az interjú során feltehetünk általános kérdéseket, majd lehetőségünk van a megfelelő irányba terelni a beszélgetést a tervezett koncepciónak és éppen az egyedi esetben fennálló lehetőségeknek megfelelően. Nagy nehézség azonban az interjúkban a kérdőívekkel szemben az időzítés kritériuma. Egyenként egyeztetni az interjúalanyokkal időpontot, majd beszélgetésre időt szánni természetesen nagyobb terhelés, mint a kérdőív kiküldése egyszerre több embernek.

A további teendők a különböző módszerek tekintetében hasonlóak, majd a kérdőív és az interjúk adatai, információi alapján elkészíthetjük a szituációs gyakorlatokat. Az adatok feldolgozása természetesen különböző, viszont az információk alapján a szituációinkat könnyedén előállíthatjuk függetlenül attól, hogy az információinkat milyen módszerrel is szereztük.

Végig nézve a két eltérő módszert, természetesen nem jelenthető ki egyik sem jobb vagy roszszabb módszernek, az előbbiekben csupán igyekeztünk megvizsgálni előnyeiket és hátrányaikat egymással szemben.

# 4. Szituációs gyakorlatok kidolgozása

Az információk gyűjtését követően elkezdődött a szituációs gyakorlatok kialakítása. Ehhez elsősorban meghatároztuk a jó szituációs gyakorlatok ismérveit, majd ennek szem előtt tartásával fogalmaztuk meg a szituációs gyakorlatokat.

## 4.1. Egy jó szituációs gyakorlat jellemzői

1. Megtörtént eseményen alapuló: Talán a legfontosabb egy szituációs gyakorlattal kapcsolatban, hogy megtörtént események alapján készüljön. Ennek természetesen az az oka, hogy a lehető legéletszerűbben lehessen elemezni a történteket és a tanulságokat leszűrni. A szituációs gyakorlat így biztosan olyan eseményt próbál reprodukálni, ami már egy cég életében megtörtént probléma, így előfordulhat a résztvevőkkel is munkájuk során.

2. Kulcsszemélyek hangsúlyozás: Mivel kutatásunkat a Scrum mester szerepkör köré rendeztük, a szituációkban kulcsszerepet játszik a Scrum Master, kiemelten fontos kell legyen az ő döntése és szerepe az adott környezetben. Ezzel megfigyelhető milyen képességek szükségesek ahhoz, hogy egy Scrum Master az optimális működést elősegítő döntést hozza meg a munkája során.

3. Szereplők megfelelő bemutatása: A gyakorlat leírásában fontos a szereplők pontos leírása. Úgy definiáljuk a szereplőket, tulajdonságaikat, viselkedésüket, hogy ideálisan tudjuk reprodukálni a szituációt. A főszereplő személyek definiálása különösen fontos. Amennyiben nem elég pontosak a tulajdonságok, akkor a szituáció nem érheti el a célját, nem alakulnak ki a mérvadó események. Akár előfordulhat, hogy a gyakorlat így értelmét veszíti. Fontos azonban az is, hogy a szereplők személyiségjegyei viszonylag általánosak legyenek, amelyekkel a gyakorlat résztvevője jó eséllyel találkozhatott már korábban és amelyekkel azonosulni tud. A résztvevők könnyebben el tudnak játszani egyszerűbb, általános személyiségjegyet, minimális színjátszó tudással is.

4. Lehetséges megoldás felvázolása: Egy jó szituációs gyakorlat tartalmazzon egy lehetséges megoldást is a felvázolt problémára. A megoldás alapján a résztvevők következtetéseket vonhatnak le, elemezhetik saját döntéseiket és megtanulhatják, hogyan tudnak javítani a működésen.

5. Tanulságok levonása: A megtörtént szituációk a gyakorlatok elvégzése során kidomborítanak különböző tényezőket, amelyeket szükséges, hogy a Scrum Master megfelelően tudjon kezelni. Fontos ezek összefoglalása a szituációs gyakorlatokat követően. Természetesen nem minden tudás tanítható meg az ipari környezettől sterilebb környezetben, amikor a szereplők a szituációs gyakorlatot játszák le, de ebben az esetben is fontos a tudatosítás.

## 4.2 Példa szituációs gyakorlatra

Ebben az alfejezetben egy kidolgozott szituációs gyakorlat szövege olvasható. A <> jelek között olyan hivatkozások találhatóak, melyek magyarázatot adnak arra, hogy miért az adott módon alakítottuk ki a gyakorlatot. Az általunk készített gyakorlatok megoldásáról a Scrum Master utólag a véleményét is kifejtette. A továbbfejlesztésnél emiatt figyelhettünk azokra a szempontokra, hogy az adott Scrum Master hogyan cselekedne utólag.

**Gyakorlat**:

Adott egy Scrum csapat. A Scrum csapat tagjai különböző tapasztalatokkal rendelkeznek, emellett eltérő mentalitással és motivációval. <1> Ezek leírása az alábbi pár sorban olvasható. Fontos megjegyezni, hogy a szituációs gyakorlat szempontjából a fejlesztők, tesztelők, illetve a Scrum Master lényeges szereplője a történéseknek, így csak ők kerülnek részletezésre, de tetszőlegesen bővíthető a szereplők listája. <2> Kisebb létszám esetén Tesztelő1, Megfigyelő2, Fejlesztő2 hagyható ki, vagy a megfigyelők kezelhetőek egy megfigyelőként is.

**Alapinformációk**

Minden karakter csak saját leírását ismeri.

**Résztvevők**

**Scrum Master**: Viszonylag új vagyok a szakmában, 2-3 éve dolgozom Scrum Masterként, de ezt az időt végig az előző csapatomnál töltöttem. Most egy új cégnél dolgozom, <3> a betanulásom során végignéztem egy másik csapat működését, most pedig kezdődik a közös munka az új csapatommal. Scrum mesterként szeretném, hogy megbeszéléseinken a Scrum értékek mindig érvényesüljenek: tisztelet, nyitottság, fókusz, bátorság, elköteleződés.

**Fejlesztő 1**: Sokak szerint határozott személyiség vagyok. Szeretem átfogóan ismerni a projekteket, tudni a lehető legtöbb dologról és megismerni a döntések hátterét. Fiatal vagyok, de nagy ambíciókkal rendelkezem, ezért is szeretnék minden döntésbe belelátni, hogy ebből is tanulhassak. Határozottságom miatt előfordul, hogy vitába keveredem. <4>

**Fejlesztő 2:** Nem szeretem a feleslegesnek tűnő dolgokat, nagyon sok projektünk van jelenleg, így szeretném, ha hasznosan töltenénk a munkaidőnket, nem felesleges meetingekkel. Az agilis működés nincs ellenemre, de még csak mendemondákból ismerem a „jó" működést, a tapasztalat mindig mást mutat. <5>

**Fejlesztő 3**: 15 éve dolgozom a cégnél, szerintem jó meglátásaim vannak a dolgokról, ennek ellenére inkább csak akkor beszélek, ha kérdeznek, nem igazán vagyok kommunikatív. <6> Az évek alatt nagyon sok főnököt, struktúrát, szervezeti átalakulást átéltem, nem igazán vagyok lelkes egy-egy új rendszer elején, inkább megvárom hogyan bizonyít a gyakorlatban.

**Tesztelő 1**: Középkorú, de junior tesztelő vagyok. Majdnem 20 évig dolgoztam tanárként, így nem esik nehezemre az embereket kezelni, viszont alig két éve váltottam pályát és lettem tesztelő, emiatt természetesen nem sok tapasztalatom van még. Nem tartom magam sem sokat beszélő, sem csendes embernek, teljesen átlagosnak érzem magam.

**Megfigyelő 1**: Mennyire tud mindenki érvényesülni a beszélgetés folyamán? Ki az, aki jobban tud, ki az, aki kevésbé? Mennyire igyekszik a Scrum Master figyelni arra, hogy mindenkit meghallgassunk? <7> (bátorság, nyitottság, tisztelet értéke)

**Megfigyelő 2**: Mennyire próbálja motiválni a csapatát, akik kicsit kevésbé hisznek már a dolgokban? (elköteleződés értéke) Mennyire tudja tartani a fókuszt a megbeszélés céljáról? (fókusz értéke)

**A szituáció**

Új Scrum Master érkezett a csapathoz, ez az első csapata a cégnél, korábbi tapasztalatai nincsenek a céggel kapcsolatban. Először szeretné egymás megértése érdekében tisztázni a csapattal az alapokat, mit értünk az alapvető közös értékek alatt pl.: tisztelet, nyíltság stb. A csapatnak ez már sokadik Scrum Mastere és a sokadik ilyen meetingje. A korábbiakban is előfordult már, hogy nagy lelkesedéssel lefektették az alapszabályokat, meghatározták a fontos értékeket és a későbbiekben ez nem lett betartva.

**Egy lehetséges megoldás**

A szituációs gyakorlat alapjául szolgáló történetet egy tapasztalt, 5-6 éve a pályán lévő Scrum Master mesélte el nekünk. Szerinte a legjobb megoldás azonnal érvényesíteni a megbeszélt értékeket. Ha tahát az első ismerkedéssel és alapok tisztázásával foglalkozó megbeszélésen tiszteletről és nyíltságról beszéltünk, akkor már azon a megbeszélésen is hallgassunk meg mindenkit, beszéljünk nyíltan a problémákról stb.

A szituációs gyakorlat segítségével a résztvevők fejleszthetik a Scrum értékek közül a nyíltságot és a bátorságot is, mivel egy számukra ismeretlen embernek kell elmondaniuk esetleges problémáikat és hogy hogyan zajlana számukra az ideális munka. Emellett még a tiszteletet is fejleszthetik a résztvevők az esetlegesen felmerülő problémák megoldásával és a megbeszélt irányelvek megtartásával.

**Megjegyzések, kiegészítések**

<1> Fontos életszerű karaktereket felvázolni, akik rendelkeznek valamilyen domináns tulajdonsággal, de lehetőleg olyan egyszerű legyen ez a tulajdonság, amilyennel már bárki találkozhatott életében.

<2> Természetesen nem csak a részletezett emberekből állhat egy csapat, de a szituáció szempontjából a részletezett karakterek a fontosak, általuk alakulhat ki az tanulságokat biztosító szituáció.

<3> A szituáció szempontjából fontos, hogy a Scrum Master a cégnél új alkalmazott, mivel így nincsenek korábbi tapasztalatai az eddigi sikertelen működésről, csak a csapata elmondására alapozhat.

<4> A karakterben kiemelten fontos, hogy szeretne minél több részletet megtudni, heves vérmérsékletű és kíváncsi. Lényege az esetleges konfliktus kialakítása és hogy megtudjuk, a Scrum Masterünk ezt tudja-e jól kezelni. Általa jobban megvilágításba kerülhet a csapaton belüli nyitottság, bátorság, tisztelet működése.

<5> Szkeptikus karakter, aki fenntartásokkal kezeli a Scrum Master kijelentéseit és ellenállást tanúsíthat. Szintén azért fontos, hogy megfigyelhető legyen a Scrum Master viselkedése ebben a helyzetben. Általa jobban megvilágításba kerülhet a csapaton belüli fókusz és elköteleződés működése.

<6> Csendes karakter, a Scrum Masternek igyekeznie kell bevonni őt is a beszélgetésbe.

<7> Lényegében a csapat dinamikát fogja tudni visszatükrözni és a Scrum Master csapattal tanúsított viselkedését.

**Magyarázat**

Szituációs gyakorlatunk megalkotásánál természetesen figyelembe vettük a jó szituációs gyakorlat korábban bemutatott ismérveit. Az interjúknál a Scrum Masterek saját eddigi munkájukból idéztek fel valós, velük megtörtént szituációkat. Az interjúalanyokat emellett megkértük, hogy olyan szituációkat idézzenek fel nekünk, amelyeknek központjában az Ő szerepük, döntésük áll, hogy az Ő szerepüket tartsuk fókuszban. A szituációk bemutatásakor, megbeszélésekor a megkérdezett Scrum Masterek elmondták nekünk az adott szituáció megoldását, emellett megosztották velünk utólagos észrevételeiket is. Elmesélték, hogy a mai ismereteik és tudásuk szerint másképpen cselekednének-e vagy hogyan lehetne jobban megoldani az adott helyzetet. Igyekeztünk emellett a szituációban szereplő karaktereket megfelelően részletesen bemutatni, de fontos volt a karaktereket még a lehető legegyszerűbben jellemezni, hogy az őket eljátszó résztvevők könnyebben megformálhassák a karaktert.

## 4.3. Példák egyéb szituációs gyakorlatból

A korábbiakban láthattuk, mennyire fontos a szituációs gyakorlat kialakításakor, hogy a lehető legjobban garantáljuk, hogy a helyzetgyakorlat a megfelelő irányba haladjon. A történések központjában karaktereink állnak, így sokszor már a személyiségük leírásánál garantálhatjuk ezt.

Egy másik szituáció központjában például egy olyan fejlesztő csapat állt, ahol egy kérdés felmerülése miatt két táborra szakadtak. A Scrum Master fontos feladata a helyzetben a konfliktus kezelése és a vita kordában tartása. Annak érdekében, hogy garantáljuk a konfliktus kialakulását és ezzel ennek a problémának megoldására késztessük a Scrum Mastert, fontos, hogy mind a két csapatba konfrontálódásra hajlamos karaktereket tervezzünk.

Az alábbi személyektől várjuk a konfliktus kialakulását a szituációban:

**Fejlesztő 1:** Sokan gondolnak erős személyiségnek, szeretek mindig beszélni és én kimondani a végső szót. Sokszor azt hiszem, hogy biztosan az én meglátásaim a helyesek, ilyenkor teljesen beszűkül a látóterem sajnos és csak a saját javaslatom előnyeit látom mindennel szemben. Egy-egy vitában a nagy érvelés közepette akár a témától is eltérek és az igazamat a másik korábbi hibáinak felemlegetésével is igyekszem alátámasztani (Visszafogott személyeskedés).

**Fejlesztő 2**: Utálom, ha valaki konkrétumok nélkül próbál vitatkozni, szerintem csak tiszta, egyértelmű érvek mellett van értelme a nézőpontjainkat szembe állítani. Mindig ragaszkodom ehhez, és szóvá is teszem, hogy próbáljunk meg a konkrétumokra hagyatkozni. Talán néha túlzottan is ragaszkodom az érvekhez és a konkrétumokhoz. El tud fogyni a türelmem, ha valaki nem figyel más nézőpontot csak a sajátját, ilyenkor próbálom őt erre ráébreszteni minél előbb és minél határozottabban.

Látható, hogy a cél két vitában teljesen ellentétes személyiség különböző oldalra pozicionálása ezáltal vita generálása, ami így elősegítő az események bizonyos irányú menetét, például a konfliktus kialakítását.

A szituációs gyakorlatok során nagyon fontos, hogy a résztvevők képességeik legjavát nyújtva igyekezzenek a szerepeket eljátszani, hiszen így a lehető legvalósághűbb lehet a szituáció. Lehetőségünk van az események egy pontján akár a szituációk megállítására is, hogy tudjunk részkövetkeztetéseket megbeszélni, ami pontosabb eredményekhez segíthet minket.

# 5. Összefoglalás és további tervek

Kutatásunk során kérdőívet állítottunk össze és interjúkat vezettünk le, melyek segítségével adatokat gyűjtöttünk valós, ipari, Scrum módszertan alkalmazásával kapcsolatos szituációkról.

Az adatok elemzését követően tudtunk készíteni valós eseményeken alapuló szituációs gyakorlatokat. A szituációs gyakorlatok igyekeznek egyszerű, lehetőleg bármelyik cégnél előforduló problémákra alapozni, kerülve a túl speciális eseteket. Olyan karaktereket tartalmaznak, melyekkel a hallgatók könnyen azonosulhatnak, és melyek kódolják az általunk megjeleníteni kívánt szituációt (pl. konfliktus). A szituációk kielemzése során egyértelműen azonosíthatóak olyan tulajdonságok, melyekkel a Scrum Mastereknek rendelkezniük kell szerepkörük helyes betöltéséhez (pl. konfliktuskezelés). A gyakorlatok leírásai egy lehetséges megoldást is felvázolnak, hogy példát állítsanak a Scrum helyes működésére.

Természetesen a gyakorlatok sosem lesznek képesek tökéletesen reprodukálni egy valós ipari szituációt, a munkakörnyezet és a realitás hiányában. Törekszünk viszont ennek a lehető leghatékonyabb megközelítésére, hogy később valós ipari helyzetekbe a lehető legtöbb tapasztalattal, tudással és képességgel kerüljenek a hallgatóink.

A jelenlegi szakasz lezárulása után a tervek között szerepel további információk szerzése ipari környezetből és ezek vizsgálata. A Scrum alkalmazásának és működésének mélyebb tanulmányozása érdekében szükséges minél jobban belelátni a hétköznapi működésébe és annak elemzésére koncentrálni. Ez lehetővé tenné a gyakorlatok további optimalizálását, vagy akár egy új megközelítésből történő kidolgozását.

Legfontosabb közeljövőbeni tervünk a szituációs gyakorlatok tesztelése a lehető legtöbb és legkülönbözőbb csoportban. Az események lefolyásának megfigyelését és a tapasztalatok kiértékelését követően a jelenlegi szituációs gyakorlatok optimalizálását a következő időszakban kiemelt prioritással kezeljük.

## 6. Köszönetnyilvánítás

A kutatási projekt az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósult meg (EFOP-3.6.3-VEKOP-16-2017-00002).

## 7. Források

1. Ken Schwaber and Jeff Sutherland: *The Scrum Guide*
2. Digital.ai: *14th annual State of Agile report* [Online] [Hivatkozva 2020.október 21.] https://stateofagile.com/#ufh-i-615706098-14th-annual-state-of-agile-report/7027494
3. Scrum.org: *Scrum.org* [Online] [Hivatkozva: 2020. október 19.] https://www.scrum.org/
4. Scrum Alliance: *Scrum Alliance* [Online] [Hivatkozva: 2020. október 19]. https://www.scrumalliance.org/
5. CIO from IDG: *Scrum master certification: Top 9 certs for agile pros* [Online] [Hivatkozva: 2020 október 22] https://www.cio.com/article/3391591/scrum-master-certification-top-9-certs-for-agile-pros.html
6. Scrum.org: *Scrum Glossary* [Online] [Hivatkozva: 2020. október 19.] https://www.scrum.org/resources/scrum-glossary
7. Kovácsné Pusztai Kinga: *A probléma-alapú oktatás az informatika órán* (2017)
8. Ilyés Enikő: *Esettanulmány: Agilis szoftverfejlesztés egyetemi kurzuson* (2017)
9. Ilyés Enikő, Pintér Balázs, Szendrei Rudolf, Cserép Máté: *A Szoftvertechnológia tárgy agilisra vált* (2019)
10. Ilyés Enikő: *Agilis módszertan kutatás-fejlesztés laborban* (2019)
11. Molnár Gyöngyvér: *Problémamegoldás és probléma-alapú tanítás* (2004)
12. David Boud, Grahame E Feletti: *The Challenge of Problem-Based Learning* (1997)
13. Szögedi Ildikó: A probléma alapú tanulás, mint új gyakorlati készségfejlesztő módszer, az egészségügyi felsőoktatásban (2012) [Hivatkozva 2021. január 4.] http://ltsp.etk.pte.hu/portal/wp/File/Doktoriiskola/Tezisfuzetek/Szogedi_ertekezes2.pdf

# Sikeres oktatási eszközök és gyakorlatok – Egy *Munkaszervezés, projekt kommunikáció* kurzus tapasztalatai

Ilyés Enikő

ilyese@inf.elte.hu
ELTE IK

**Absztrakt.** Az oktatónak sok eszköz és gyakorlat típus áll rendelkezésére a tananyag átadására. 2019 őszén az ELTE Informatikai Kar felsőoktatási szakképzéses hallgatói 6 oktatási eszköz és 14 gyakorlat típus segítségével sajátíthatták el a „Munkaszervezés, projekt kommunikáció" tárgy tananyagát. Némely ezek közül gyakran használt az egyetemi oktatásban (például: *Vetítés, Projektmunka* stb.), mások ritkábban (például: *Szociometria, Szituációs gyakorlat, Számítási feladat csoportban, Projekt konzultáció* stb.). Ebben a cikkben részletesen tárgyaljuk az alkalmazott eszközöket és gyakorlat típusokat, és példákkal illusztráljuk. Ezt követően feltárjuk, hogy 60 hallgatói értékelés alapján melyek bizonyultak a legsikeresebbeknek.

**Kulcsszavak:** oktatási eszközök, oktatási gyakorlatok, projektmunka, szituációs gyakorlat

## 1. Bevezető

Az oktatónak sok eszköz és gyakorlat típus áll rendelkezésére a tananyag átadására. A négy oktatási szint - elmélet, gyakorlat, képességek, attitűd - fejlesztésére más-más eszközök és gyakorlatok alkalmasak, némelyek ugyanakkor több szintet is érinthetnek. Emellett a változatos eszközök és gyakorlatok használatának a hallgatók motiválásában is fontos szerepe van: A különböző személyiségtípusú hallgatókat más-más típusú eszközök és gyakorlatok kötik le, másként szeretnek és tudnak hatékonyan fejlődni [1].

Az informatikus hallgatók munkaszervezésre irányuló tanulmányai esetében is bizonyított, hogy több különböző módszer alkalmazása stimulálóan hat [2]. [3]-ban gyakorló projekt menedzsereket kérdeztek meg, hogy mely képességek elsajátítására milyen módszert használjanak, és a témakörök kapcsán több különböző módszert gyűjtöttek össze.

Napjainkban az IT világában az Agilis módszertanok a legdivatosabb munkaszervezési keretrendszer. Alkalmazásukra külön oktatni kell a hallgatókat, ugyanis az nem egyértelmű számukra, olykor kifejezetten nehezen vagy egyáltalán nem tudják adoptálni csupán ipari példa nyomán [12]. Az oktatásukat illetően több kísérlet van jelen a szakirodalomban. Oktatják szimulációk segítségével [4], de leggyakrabban szoftverfejlesztési csoportos projektmunka által [5], [6], [7]. Ehhez természetesen szükség van arra, hogy a hallgatók már tudjanak programozni. A szerző is több ízben szerzett tapasztalatokat agilis módszertanok szoftverfejlesztési projektek általi oktatásában [8], [9], [10]. A jelen kísérletben viszont nem alapozhatott a hallgatók szoftverfejlesztési tudására.

Ezekből az okokból kifolyólag egy féléven át, gyakorlati óra keretein belül számos eszközt és gyakorlat típust alkalmazott a tananyag átadására. Kérdőív segítségével felmérte, hogy az alkalmazott módszerek mennyire voltak hatékonyak és élvezetesek a hallgatók számára. E cikk célja, hogy visszaigazoltan sikeres ötletekkel lássa el az olvasót oktatási eszközök és gyakorlatok témakörben.

Az 1.1. alfejezetben tisztázzuk a kísérlet kereteit, a 2. fejezetben pedig az alkalmazott eszközöket és gyakorlatokat mutatjuk be, példákkal illusztrálva. A 2.2. alfejezetben kitérünk arra, hogyan lehet módosítani ezeket az eszközöket és gyakorlatokat, amennyiben láttásérült hallgató is részt vesz az

órán. A 3. fejezetben az eredmények és következtetések kerülnek tárgyalásra. A 4.fejezet összegzéssel zárja a cikket.

### 1.1. A kísérlet keretei

A „Munkaszervezés, projekt kommunikáció" tárgy keretei között valósult meg a kísérlet. A tárgy célja, hogy informatikai projekt működéssel, működtetéssel és üzleti kommunikációval kapcsolatos gyakorlatias ismereteket nyújtson. A tárgy címzettjei az ELTE Informatikai Kar felsőoktati szakképzéses, első féléves hallgatói (nagyrészük még nem tud, vagy csak nagyon alap szinten tud programozni).

Mivel a kísérlet eredményeit befolyásolhatta a hallgatók félév végi osztályozási módszere, ezt is megemlítjük a keretek tisztázásakor: A hallgatók a félév során maximum 100 pontot gyűjthettek. A pontokat a következő feladatok teljesítése révén lehetett összegyűjteni: maximum 60 pontot a projektmunkával, 15 pontot a projekt félév végi bemutatásának minőségével, 25 pontot félév végi csoportos interjú során. Az utóbbi 5-ös csoportonként 25 percben zajlott és az elméleti anyag ismeretét valamint alkalmazási minőségét mérte fel. Az érdemjegyek a következő leképezés szerint alakultak: kevesebb mint 55 pont esetén elégtelen (1), 55 - 64 pont között elégséges (2), 65 - 74 pont között közepes (3), 75 - 84 pont között jó (4), 85 – 100 pont között kiváló (5).

A kísérletben összesen 60 személy vett részt, akik négy különböző gyakorlati csoportba tartoztak (4 különböző időpontban volt órájuk, szemináriumi teremben).

## 2. Oktatási eszközök és gyakorlatok

A továbbiakban az alkalmazott eszközök és gyakorlatok bemutatására kerül sor. Az általános leírást konkrét, megvalósult esetek leírása is kiegészíti.

**Eszközök:**

*E-tankönyv:* Az első órán a hallgatók e-könyv formájában jutnak hozzá a tárgy teljes tananyagának 85%-át részletesen bemutató tankönyvhöz.

> Példa: Langer Tamás Projektmenedzsment a szoftverfejlesztésben – A hagyományostól az agilisig – 2. bővített, átdolgozott kiadása a Panem kiadó weboldaláról ingyenesen letölthető. Az első órán azt a házi feladatot kapták a hallgatók, hogy letöltsék ezt a könyvet.

*Jegyzet:* A félév utolsó negyedében a hallgatók megkapják a tanórákon leadott elméleti anyag oktató által készített kivonatát - minden témakör lényegretörő, tömör összefoglalását. A jegyzet félév végi megosztását az a feltételezés indokolja, hogy ennek birtokában a hallgatók kevésbé motiváltak a saját jegyzet készítésére, illetve a tankönyv használatra.

> Példa: A hallgatók a félév zárása előtt 3 héttel megkapták a tananyag 35 oldalas összefoglalóját (elektronikus formában).

*Tábla:* Az oktató a táblára rögzíti a címeket, alcímeket, kulcsszavakat és modelleket a tanóra során. Ezt rendszerint bővebb szóbeli magyarázat kíséri.

> Példa: A projekt témakört bemutató órán a projekt definíciója, a projekt négyszög ábrája, a minőség „képlete" került fel a táblára. Ezen fogalmak felvázolása előtt a hallgatók megvitatták aktuális meglátásaikat a projekt fogalmát illetően. Az itt felsorolt definíciók megismerése során ezekre is reflektáltunk.

*Vetítés*: Az oktató előre elkészített diasort vetít ki a tanóra során. A vetített anyagot további szóbeli magyarázat és a hallgatókkal közös tárgyalás, vélemény-kifejtés kíséri. A félév végén a hallgatók megkapják a vetített anyagot elektronikus formában.

Példa: A DISC személyiségtípusok különböző vetületeit (motiváció, csapat-szerep, konfliktuskezelési módok stb.) egy diasor foglalta össze. A vetítés során a hallgatók elmondhatták saját tapasztalataikat a témát illetően.

*Papír kártyák:* A hallgatók papír kártyákat kapnak a tanóra során, melyeket bizonyos kritériumok alapján kell csoportosítsanak, rendszerint csoportmunkában.

Példa: A Scrum módszertan szerepköreinek elmélyítéséhez a 3-4 fős hallgatói csoportok kártyákat kaptak. Minden kártyán egy aktivitás szerepelt, pl. „Priorizálja a termék kívánságlistát", „Minden futam végén visszatekintést vezet le", stb. A feladat az volt, hogy a hallgatók három csoportra osszák a kártyákat aszerint, hogy az egyes aktivitások a Termékgazda, a Scrum mester vagy a Fejlesztőcsapat felelősségi köréhez tartoznak.

*Internethasználat*: A hallgatóknak használniuk kell okoskészülékeiket a tanóra során ahhoz, hogy bizonyos témával kapcsolatos információknak helyben utánanézzenek.

Példa: Az agilis módszertanok bevezetése során a hallgatók felkérést kaptak, hogy okoskészülékeik segítségével keressék meg az interneten elérhető agilis kiáltványt.

**Gyakorlatok:**

*Szociometria:* A hallgatók és az oktató bizonyos kritériumoknak megfelelően kell elhelyezkedjen a térben, például egy képzeletbeli skálán, amely a tanterem egyik sarkától a másikig húzódik. A személyek térbeli elhelyezkedése tehát többletjelentéssel bír.

Példa: A félév elején (többek között a projektcsapatok létrehozása érdekében) fontos volt, hogy a hallgatók (és oktató) megismerjék egymást. Az ismerkedést szociometriás játék támogatta: a jelenlévőknek el kellett helyezkedniük egy képzeletbeli Magyarország térképen aszerint, hogy honnan jöttek; képzeletbeli skálán aszerint, hogy mennyi programozói tapasztalattal rendelkeznek; „hobbiszigetek"-et alkottak, stb.

*Szavazás:* A hallgatók egy bizonyos témát-kérdést illetően szavaznak, ezáltal véleményt formálnak és fejtenek ki egy ügyben.

Példa: Geert Hofstede szervezeti kultúrák modelljének elemzésével párhuzamosan a hallgatóknak szavazniuk kellett azt illetően, hogy a tulajdonság-párok közül (pl. folyamatorientáció-eredményorientáció, nyitott-zárt stb.) melyiket preferálnák egy potenciális munkahely esetén.

*Önismereti tesztek:* Olyan tesztek kitöltése és kielemzése, melyek révén a hallgatók jobban megismerhetik önmagukat (pl. működésüket, motivációjukat) a tananyag gyakorlati részeit illetően.

Példa: Egy DISC önismereti teszt kitöltése, kiértékelése révén elemeztük a hallgatók csapatmunkabeli működését, motivációját.

*Csapatmunka:* A hallgatók spontán csapatokat formálnak a tanórán és bizonyos kérdésekre együtt keresik a választ.

Példa: Az ügyfelekkel való kommunikáció gyakorlása érdekében a spontán kialakult 3-4 fős hallgatói csapatok egy-egy borítékot kaptak, mindegyikben egy problémás ügyfél leírása (pl.: Potyázó Panna, Egocentrikus Edgárd, Hisztérikus Henrik stb.) A csapatok feladata az volt, hogy kitalálják, hogyan érdemes profi ügyfélfogadóként az egyes személyekkel bánni. Miután közösen megbeszéltük ötleteiket, a borítékban lévő egyéb színű lapot is kihúzhatták, ahol szakmabeliek tippjeit olvashatták az ügyfél-típusok kezelését illetően.

*Számítási feladat csapatban:* A hallgatók számítási feladatokat oldanak meg 3-4 fős kiscsoportokban.

Példa: Spontán kialakult 3-4 fős csoportok egy-egy időelemzéses feladatot kaptak. Adott volt egy projekt tevékenységeinek becsült időtartama, illetve az őket megelőző tevékenységek listája (voltak párhuzamosítható tevékenységek is). A feladat a projekt teljes átfutási idejének megbecsülése volt.

*Modellezés:* Segédeszközök használatával a hallgatók (akár az oktatóval együtt) ábrázolnak egy elméleti modellt.

Példa: Az összetett szervezeti modellek elsajátítása érdekében a hallgatók az egyes modellek hierarchikus felépítését szemléltetően álltak fel a térben. Közvetlenül a tábla előtt állt a „vezérigazgató", egy sorral bennebb két „funkcionális vezető", további sorokkal bennebb az „alkalmazottak" stb. A személyek közötti közvetlen összetartozást és kommunikációs szálakat köztük kifeszített fonalak szimbolizálták. Ezek eltérő színűek voltak funkcionális, illetve projekt csapatok esetén.

*Projektmunka*: A hallgatók egy teljes projektet vezetnek le kiscsoportban, mely több héten átível. A projektek esetében hallgatók, oktatók vagy külső személyek is betölthetik a megrendelő szerepét. A különböző lehetőségek előnyeit és hátrányait tárgyalja [11].

Példa: A hallgatói projekt célja egy, az egyetemi élettel kapcsolatos termék vagy szolgáltatás létrehozásának részletes megtervezése volt. A termék-szolgáltatás ötletek is a hallgatóktól származtak. Két példa projektötletre: UniMap - egyetemi campus-térkép okostelefonos alkalmazás; Családbarát ELTE - gyerek- és kisállat megőrző szolgáltatás az egyetem épületében.

*Házi feladatok, mint a projektmunka részfeladatai*: A hallgatók hétről hétre kapnak házi feladatot, melyek egymásra épülnek és melyek együttesen a projekt megvalósítását képezik.

Példa: A termék-szolgáltatás létrehozásának megtervezésében egy részfeladat és egyben házi feladat volt a kivitelezéshez szükséges résztevékenységek listába gyűjtése és sorrendbe helyezése.

*Szituációs gyakorlat:* A hallgatók (akár az oktatóval együtt) egy bizonyos képzeletbeli szituációba belehelyezkednek és annak megfelelő szerepeket vesznek fel, annak megfelelően reagálnak.

Példa: Lift-út gyakorlat: Az oktató és projektcsapatonként egy hallgató egy befektető és egy támogatást kérő liftes találkozását szimulálták. A lift út egy percig tartott. A támogatást kereső (hallgató) ennyi idő alatt kellett felkeltse a befektető érdeklődését a projekt iránt. A szituáció eljátszása után közös kielemzésére került sor, majd a hallgatók újra kipróbálhatták magukat a helyzetben.

*Visszajelzés adása egymásnak*: A hallgatók visszajelzéseket adnak egymás munkájára vonatkozóan.

Példa: A hallgatók a szituációs gyakorlatot, illetve a projekt bemutatást követően szóbeli visszajelzést adtak a többi csapat teljesítményére vonatkozóan. Észrevételeket és továbbfejlesztési lehetőségeket fogalmaztak meg.

*Reflektálás:* A hallgatók együtt visszatekintenek egy munkaszakaszra és azt elemzik az optimalizálás és tanulás céljával.

Példa: A projekt elkezdése után 3 héttel a hallgatók a tanórán visszatekintettek az eddigi közös munkájukra, megbeszélték sikereiket és kudarcaikat, javító lépéseket fogalmaztak meg.

*Dokumentum írás:* A hallgatók szöveges dokumentumot szerkesztenek eredményeikből.

Példa: A hallgatók csapatonként projektdokumentumot készítettek a félév végére. Az 5-10 oldalas dokumentum óráról-órára történő bővítésére és szerkesztésére is kaptak meghívást a hallgatók.

*Projekt konzultáció:* Az oktató a tanórából elkülönített időtávot szán arra, hogy minden hallgatói projekt-csoporttal elbeszélgessen és visszajelzéseket, továbbfejlesztési lehetőségeket adjon nekik a projektükre vonatkozóan (még a projekt bemutatása előtt).

Példa: A projekt bemutató előtti órán az oktató projekt-csoportos konzultációt tartott, csoportonként 20 perc erejéig adott visszajelzéseket és javítási ötleteket a csapatoknak.

*Kiselőadás:* A hallgatók az általuk végzett munka eredményét kiselőadás formájában mutatják be.

Példa: A félév utolsó előtti óráján a hallgatók csapatonként 15 percben mutatták be projektük eredményét kiselőadás formájában.

Az 1. táblázat összefoglalja, hogy milyen témaköröket tartalmazott a „Munkaszervezés, projekt kommunikáció" tárgy és azok milyen eszközökkel és gyakorlatokkal lettek megközelítve.

| Témakör | Eszközök-gyakorlatok |
|---|---|
| Megismerkedés | Szociometria |
| Projekt definíciója, projekt négyszög | Tábla |
| A projekt környezete a vállalat - országok és vállalatok kultúrája | Csapatmunka<br>Szavazás |
| Szervezeti modellek | Modellezés |
| Projekt szerepek | Papír kártyák |
| Projekt életciklusa | Tábla |
| Időelemzés | Tábla<br>Számítási feladatok csoportban |
| Szoftverfejlesztési módszertanok | Tábla |
| Agilis módszertanok, Scrum módszertan | Tábla<br>Internet használat |
| Scrum szerepkörök | Papír kártyák |
| A hatékony csapatok közös tulajdonságai | Tábla |
| DISC személyiség modell | Vetítés<br>Önismereti teszt |
| Kommunikáció az ügyféllel | Papír kártyák<br>Feladványok |
| Projektmunka | Házi feladatok, mint a projektmunka részfeladatai<br>Szituációs gyakorlat<br>Visszajelzés adása egymásnak<br>Reflektálás<br>Dokumentum írás<br>Projekt konzultáció<br>Kiselőadás |
| Teljes tananyag összegzése | E-tankönyv<br>Jegyzet |

**1. táblázat**: Témakörök és használt eszközök-gyakorlatok

## 2.1. Eszközök és gyakorlatok újragondolása láttássérült hallgató esetében

A 2019-es „Munkaszervezés, projekt kommunikáció" egyik gyakorlati csoportjába egy látássérült hallgató is tartozott. Ebben az alfejezetben tárgyalásra kerül, hogy láttássérült hallgató jelenléte milyen kiegészítéseket/módosításokat igényel az eszközök-gyakorlatok alkalmazását illetően. Bemutatásra kerül e konkrét esetben szerzett néhány tapasztalat is.

Az *E-tankönyv* és az elektronikus formában szétküldött *Jegyzet* a látássérült hallgató számára is előnyösen használható a köreikben szokványos képernyő felolvasó szoftverek révén. Amire az oktatónak érdemes kiemelten odafigyelnie, hogy a képek-ábrák-diagramok minden esetben alapos, szöveges leírással legyenek kiegészítve. A *Vetítés* esetében el lehet küldeni a látássérült hallgatónak a vetített anyagot előre, aki szinkron vagy aszinkron módon felolvastathatja tartalmukat (tehát az óra folyamán, a vetítés alatt, például a saját gépéről, de későbbi, otthoni munka során is). A *Tábla* használat esetében figyelni kell arra, hogy hangosan és elképzelhetően mesélje az oktató, amit éppen a hallgatók a táblán láthatnak (például: „A tábla közepére egy rombuszt rajzolok. A felső csúcstól, jobbra haladva a következő négy címkét helyezem el a csúcsokra: idő, költség, cél, minőség. Ezt az ábrát hívjuk projekt négyszögnek. Összefoglalja azt a négy tényezőt, melyek a projekt megtervezése és kivitelezése során szem előtt kell tartanunk, és amelyek hatnak egymásra.") Az *Internethasználat*, mint eszköz valószínűleg megszokott a látássérült hallgató számára magánéleti és egyéb oktatási ügyekből kifolyólag, tehát ez bátran bevethető az óra során is. A *Papír kártyák* használata okozza a legnagyobb kihívást az eszközök közül – ebben az esetben érdemes egy jól látó hallgatót rendelni a látássérült hallgató mellé, aki felolvassa a kártyák szövegét és segít ezek elrendezésében. Bár hatékonyabb lehet, ha mindig ugyanaz a hallgató vállalja ezt a szolgálatat (jobb összehangoltságot eredményez), vigyázzunk arra, hogy a segítő hallgató ne terhelődjön túl. Mindig köszönjük meg a segítséget.

Az összes olyan gyakorlat, mely során csapatban dolgoznak a hallgatók (*Csapatmunka, Számítási feladat csapatban, Projektmunka, Projekt konzultáció*) előnyösen alkalmazható látássérült hallgatók bevonásával is, hiszen a csapattagok pótolni tudják egymás hiányzó képességeit. Ugyanakkor fontos figyelni rá, hogy a csapat valóban bevonja a látássérült hallgatót, ne kezelje kívülállónak.

E kísérlet során volt egy olyan hallgató, aki önkéntesen, minden óra folyamán a látássérült hallgató mellett ült és minden gyakorlat során segítségére volt, a csapatmunkákban is melléje szegődött. Ez általánosságban jó stratégiának bizonyulhat, de vannak szenzitívebb információt kezelő feladatok, mint például az *Önismereti tesztek*. Ebben az esetben tapintatosan kell engedélyt kérni a látássérült hallgatótól, hogy látó csoporttársa/oktatója töltse ki/értékelje ki helyette a papír alapú önismereti tesztet az általa szóban megadott válaszok alapján. A látássérült hallgatónak természetesen van joga ebbe nem beleegyezni és kivonni magát ebből a feladatból.

A *Szavazás, Visszajelzés adása egymásnak, Reflektálás* gyakorlatok, amennyiben szóban történnek, vagy egyszerű gesztusokkal (pl. szavazás kézfeltartással) legtöbbször gond nélkül abszolválhatóak a látássérült hallgatók számára is.

Az itt tárgyalt esetben a látássérült hallgató minden szavazásban részt vett, az egyik legaktívabb volt visszajelzések adásában, bevonódott a reflektálásba.

Hasonlóan jól működhet a *Szituációs gyakorlat* – például a Lift-út szimuláció kivitelezhető, hiszen a láttássérült hallgató számára sem idegen ez az élethelyzet. A *Kiselőadás* tartása is megvalósítható, bár amennyiben ezt diasor kíséri szükség lehet a diák megelőző korrektúrájára (pl. megjelenés szempontjából), majd vetítésének vezérlésére a csapattársak vagy oktató által. *Dokumentum írás* gyakorlat típusban segíthetjük korrektúrával a látássérült hallgatót, vagy az értékelésnél eltekinthetünk a dokumentum formai követelményeitől. Esetünkben ezek a gyakorlatok mind csoportban valósultak meg, így a csoporttársak biztosították a korrektúrát.

A *Szociometriás* feladatok nem előnyösek a látássérült hallgató számára, ezeket mindenképpen csak segítséggel tudja abszolválni. Az elrendezési szempont közlése után az oktató példaként hozhatja a láttássérült hallgató esetét, akit -szóban kifejezett tulajdonsága/meggyőződése alapján- a

megfelelő helyre kísér a térben. Amikor a teljes csoport elhelyezkedett, akkor szükséges a látványt szóban elnarrálni, hogy az általa megjelenített információk a látássérült hallgató számára is elérhetővé váljanak (pl. „Úgy látom, hogy a programozói tapasztalatokat felmérő skálán a csapat 80%-a a 3-as érték körül helyezkedik el, vagyis kevés programozói tapasztalattal rendelkezik. Van egy 9-es és egy 0-ás értékünk is.") A *Modellezés* esetében szükség lehet kreativitásra, hiszen olyan modellt kell alkotnunk, ami jól érzékelhető a látássérült hallgató számára.

Esetünkben a szervezeti működések modellezését papír poharak (alkalmazottak), teás dobozok (funkcionális vezetők) és bögre(igazgató) hierarchiát ábrázoló asztalra helyezésével valósítottuk meg külön a látássérült hallgató számára.

Összefoglalva, a legtöbb gyakorlat és eszköz jól alkalmazható a látássérült hallgatók esetében is. Természetesen fontos figyelni arra, hogy ne legyen olyan információ, mely szóban nem hangzik el, például csak egy ábrán látható, a táblán. Nagy segítség tud lenni, ha a hallgató mellé tudunk rendelni egy önkéntes látó hallgatót, aki segíti őt, amennyiben szükséges – de vigyázzunk, hogy ezt a hallgatót ne terheljük túl. A csoportos gyakorlatok előnyösek, hiszen itt kiegészítik egymást a hallgatók képességei és a látássérült hallgató is tapasztalhatja, hogy hozzájárul a munkához azzal, ami az ő erőssége, például ötletgazdagság, előadói képesség stb.

## 3. Eredmények és következtetések

Mivel a 60 hallgató, 4 gyakorlati csoportban, de ugyanazon menetrend, eszközök, gyakorlatok alkalmazásával vett részt a *Munkaszervezés, projekt kommunikáció* órán, csak összesítő mérések kivitelezésére volt lehetőség (nem például összehasonlító, A-B csoport jellegű mérésre). Az adatok kérdőív segítségével lettek begyűjtve, hiszen a 60 személy túl sok lett volna például interjú jellegű adatgyűjtéshez. A hallgatók a számonkérést követően töltötték ki a kérdőívet, mely felmérte az alkalmazott eszközök és gyakorlatok hatékonyságát és élvezhetőségét. Ezt a döntést az indokolta, hogy túl sok idő- és energiaveszteség lett volna minden héten, minden gyakorlati órát követően kitöltetni velük az adott órán alkalmazott eszközökre, gyakorlatokra vonatkozó értékelő kérdőívet, bár ez kétségkívül többlet információt szolgáltatott volna (pl. egy adott eszköz, egy témán belül mennyire bizonyult hatékonynak/kedveltnek).

A kérdőívek Google űrlap formában készültek és az utolsó órán az egyetem biztosította tabletek vagy saját okoskészülékeik segítségével töltötték ki őket a hallgatók. A kitöltés körülbelül 8 percet vett igénybe. A kérdőívek kitöltése anonim módon zajlott. A kérdőívet 60 hallgató töltötte ki.

A kérdőív három fő kérdést tartalmazott és ezen kívül lehetőséget biztosított a hallgatóknak arra, hogy szabadon, saját szavaikkal is visszajelzést adjanak a tárgyra vonatkozóan. A kérdőív váza:

- Jelöld az 1 .. 5 skálán, hogy a következő témakörök mennyire „jöttek át", mennyire élesen élnek benned! (1 - egyáltalán nem; 5 - teljes mértékben) (témakörök felsorolva)
- Jelöld az 1 .. 5 skálán, hogy a következő oktatási eszközök és gyakorlatok mennyire tetszettek neked! (1 - egyáltalán nem; 5 - teljes mértékben) (eszközök és gyakorlatok felsorolva)
- Jelöld az 1 .. 5 skálán, hogy a következő oktatási eszközök mennyire voltak hasznosak számodra abban, hogy elsajátítsd az anyagot? (1 - egyáltalán nem; 5 - teljes mértékben) (eszközök és gyakorlatok felsorolva)
- Ha van bármilyen visszajelzésed a tárgy kapcsán, ide írhatod le (pl. legjobb élmény, legnehezebb élménye, leghasznosabb tapasztalat a tárgy során stb).

### 3.1. Eszközök és gyakorlatok hasznossága és tetszése

Az 2. táblázat minden alkalmazott eszköz és gyakorlat kapcsán tartalmazza, hogy a hallgatói szavazatok alapján milyen átlag értékelést kapott a „hasznosság" és „tetszés" szempontok szerint. Ezen kívül leolvasható róla, hogy a hasznosság -, illetve tetszés ranglistán hányadik helyet tölt be az átla-

gok alapján (a táblázat hasznossági szempont szerinti csökkenő sorrendbe van rendezve). Az utolsó oszlopban a hasznosság és tetszés szerinti átlagértékek különbsége látható. Ha tehát ebben az oszlopban pozitív érték található, akkor az adott eszközt vagy gyakorlatot a hallgatók hasznosabbnak tartották, mint amennyire tetszett nekik. Ha negatív érték van feltüntetve, akkor jobban tetszett nekik, mint amennyire hasznosnak tartották. Az abszolút értelemben 0,15 feletti különbségek félkövér betűtípussal vannak kiemelve a táblázatban a jobb átláthatóság kedvéért.

| **Eszközök és gyakorlatok** | **Hasznosság rangsor** | **Hasznosság átlag** | **Tetszés rangsor** | **Tetszés átlag** | **Hasznosság és tetszés különbsége** |
|---|---|---|---|---|---|
| Jegyzet | 1 | 4,56 | 1 | 4,57 | -0,01 |
| Projekt konzultáció | 2 | 4,42 | 4 | 4,38 | 0,04 |
| Projektmunka | 3 | 4,40 | 8 | 4,20 | **0,20** |
| Önismereti teszt | 4 | 4,33 | 3 | 4,40 | -0,07 |
| Visszajelzés adása egymásnak | 5 | 4,30 | 7 | 4,23 | 0,06 |
| Vetítés | 6 | 4,23 | 2 | 4,42 | **-0,19** |
| Kiselőadás | 7 | 4,18 | 10 | 4,08 | 0,10 |
| Csapatmunka | 8 | 4,16 | 5 | 4,37 | **-0,21** |
| Szociometria | 9 | 4,14 | 6 | 4,25 | -0,11 |
| Feladványok | 9 | 4,14 | 13 | 3,95 | **0,19** |
| Modellezés | 10 | 4,09 | 9 | 4,18 | -0,9 |
| Papírkártyák párosítása | 10 | 4,09 | 11 | 4,03 | 0,04 |
| Szavazás | 11 | 4,00 | 8 | 4,20 | **-0,20** |
| Számítási feladat csoportban | 12 | 3,96 | 12 | 3,97 | 0,01 |
| Papírkártyák csoportosítása | 13 | 3,95 | 12 | 3,97 | -0,02 |
| Reflektálás | 14 | 3,93 | 14 | 3,88 | 0,05 |
| Tábla | 15 | 3,89 | 14 | 3,88 | 0,01 |
| Dokumentum írás | 16 | 3,88 | 15 | 3,77 | 0,11 |
| Házifeladatok | 17 | 3,81 | 18 | 3,58 | **0,23** |
| Szituációs gyakorlat | 18 | 3,70 | 17 | 3,60 | 0,10 |
| Internethasználat | 19 | 3,58 | 19 | 3,57 | 0,01 |
| E-Tankönyv | 20 | 3,51 | 16 | 3,68 | **-0,17** |

**2. táblázat**: Eszközök és gyakorlatok értékelése hasznosság és tetszés szerint

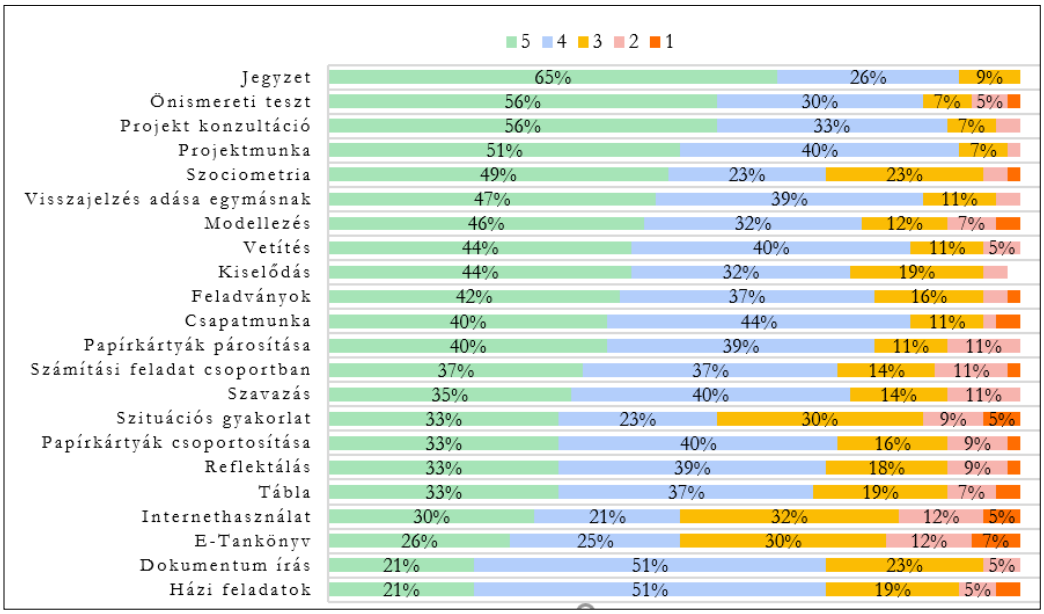
A *Jegyzet* (4,56), *Projekt konzultáció* (4,42), *Projektmunka* (4,40), *Önismereti teszt* (4,33), *Visszajelzés adása egymásnak* (4,30) eszközöket/gyakorlatokat értékelték az 5 leghasznosabbnak. A tetszés szerinti rangsorolásban az első 5 helyet a *Jegyzet* (4,57), *Vetítés* (4.42), *Önismereti teszt* (4,40), *Projekt konzultáció* (4,38), *Csapatmunka* (4,37) kapta. Láthatjuk, hogy a *Jegyzet* mindkét rangsor szerint első pozíciót ért el 4,55 feletti átlagokkal. Úgy tűnik, a hallgatók nagyon értékelik, hogyha oktató által nyújtott, tömör jegyzetet kaphatnak a tananyagról, még ha csak a félév végén is. Az *Önismereti teszt*-et is mindkét szempont szerint magasra értékelték a hallgatók, ami kifejezheti azt, hogy hasznos számukra és tetszik nekik, hogyha önmagukra is leképezhetik a tananyag valamely részét. A *Projekt konzultáció* volt még olyan gyakorlat, mely mindkét rangsor top 5-ösébe bekerült, ami arra enged következtetni, hogy nagy értéket jelent, ha az oktató személyre szabott javítási javaslatokat fogalmaz meg a hallgatók (osztályzatot is nagyban befolyásoló) feladataira. A *Vetítést* illetően azt láthatjuk, hogy a hallgatóknak jobban tetszik, mint amennyire hasznosnak tartják, ez utóbbi szempontól nem is került be az élen végzett első 5 közé. A *Visszajelzés adása egymásnak* és a *Projektmunka* a hallgatók szerint hasznos, de kevésbé tetszik nekik, mint a *Csapatmunka*. Az első öt helyre került eszközökre és gyakorlatokra mindenképpen gondolhatunk úgy, mint olyan amelyek bevetése az órán elégedettséget vált ki a hallgatókból.

A hasznosság szempont értékelését illetően szem előtt tartandó, hogy a hallgatói értékelésre hatással lehetettek a félév végi osztályozási kritériumok. Tehát fennáll a lehetősége annak, hogy amikor a hallgatók azt kellett értékeljék, hogy mennyire volt hasznos egy eszköz/gyakorlat számukra, akkor tulajdonképpen azt ítélték meg, hogy mennyire járult hozzá ahhoz, hogy a fennálló osztályozási rendszer szerint magas osztályzatot szerezzenek. Eltérő osztályozási rendszer tehát más átlagokat eredményezhetett volna.

Az abszolút értékben legnagyobb különbség egy adott eszköz/gyakorlat hasznosság és tetszés szerinti átlaga között a 0,23 volt és a *Házi feladatok, mint a projektmunka részfeladatai* illette. Hasonlóan 0,20 volt a különbség a *Projektmunka* esetében a hasznosság javára, valamint 0,19 a *Feladványok* gyakorlatot illetően. Ezen eszközök/gyakorlatok esetében tehát a hasznosság számottevően motiválóbb a hallgatók számára a munka élvezetességével szemben. Ellenkezőleg mutatkozik ez a *Csapatmunka* (-0,21), *Szavazás* (-0.20), *Vetítés* (-0,19), *E-tankönyv* (-0,17) esetében. Az abszolút értékben leginkább eltérő eszközök és gyakorlatok esetében az oktató élhet azzal a lehetőséggel, hogy azt a motivációs tényezőt hangsúlyozza alkalmazásukkor, mely magasabb átlagot mutatott.

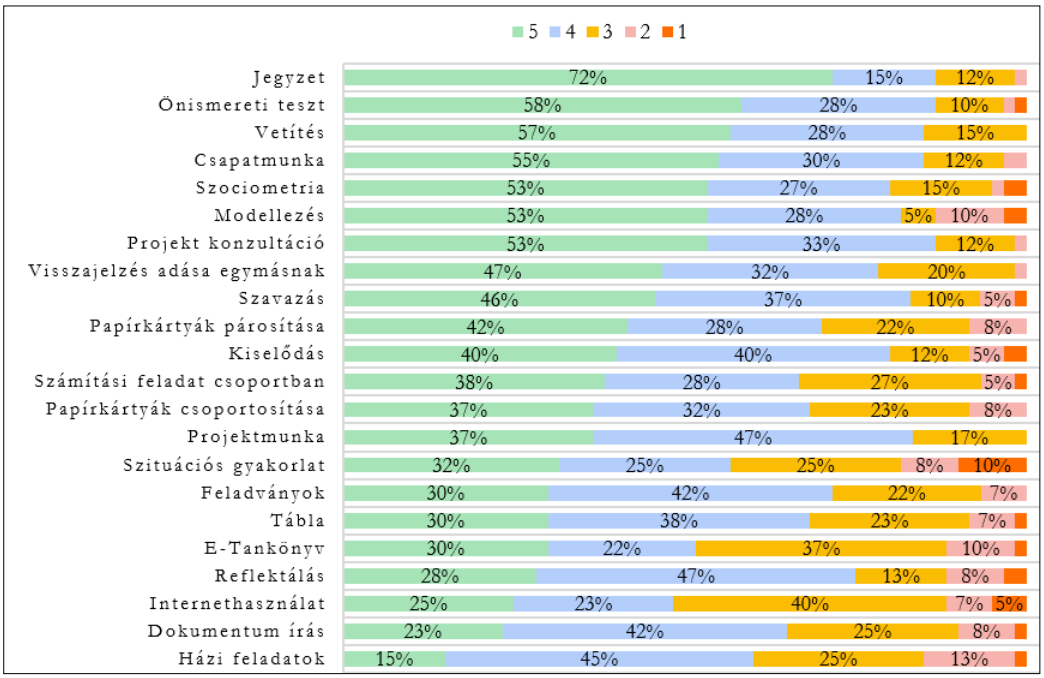
Vegyük észre, hogy a legkisebb hasznosság átlag (3,51) és a legkisebb tetszés átlag (3,68) is nagyobb, mint az értékelési skála középértéke. Ez arra enged következtetni, hogy az itt bemutatott eszközök és gyakorlatok a hallgatók számára többnyire hasznosak voltak és tetszettek nekik, tehát nincs olyan közöttük, melyet kifejezetten nem lenne ajánlott használni.

Az átlag értékeken kívül hasznos lehet látni az egyes szempontok szerint az eloszlást. Az 1. ábra a hasznosság szerinti eloszlást jeleníti meg. Van néhány olyan eszköz és gyakorlat, mely egyáltalán nem kapott 1-es értékelést (pl. *Jegyzet, Projekt konzultáció, Projektmunka, Visszajelzés adása egymásnak,*), vagyis ezek minden hallgató számára legalább valamennyire hasznosnak bizonyultak. Vannak olyan gyakorlatok, melyeket kevesebb mint a hallgatók 22%-a értékelt teljes mértékben hasznosnak, átlagukat tekintve mégis magas értéket kaptak, mert akár 51%-ban értékelték őket négyes értékkel (pl. *Dokumentum írás, Házi feladatok).* A legtöbb 1-es értékelést az *E-tankönyv* (7%), *Szituációs gyakorlat* (5%) és az *Internethasználat* (5%) kapta, ezek a gyakorlatok bizonyultak a legmegosztóbbaknak hasznosság szerint.




**1. ábra:** Eszközök és gyakorlatok hasznossága – Eloszlás diagram

A 2. ábra a tetszés szerinti eloszlást jeleníti meg. Található itt is néhány olyan eszköz és gyakorlat, mely egyáltalán nem kapott 1-es értékelést (pl. *Jegyzet, Vetítés, Csapatmunka, Visszajelzés adása egymásnak, Projektmunka*), vagyis ezek minden hallgatónak legalább valamennyire tetszettek. Vannak olyan gyakorlatok, melyeket kevesebb mint a hallgatók 40%-a értékelt teljes mértékben tetszőnek, átlagukat tekintve mégis magas értéket kaptak, mert akár 47%-ban értékelték őket négyes értékkel (pl. *Projektmunka, Reflektálás).* A legtöbb 1-es értékelést a *Szituációs gyakorlat* (10%) és az *Internethasználat* kapta, ezek a gyakorlatok bizonyultak a legmegosztóbbaknak is. Ezek használatakor fel kell készülnünk rá, hogy eltérő viszonyulásmódot, tetszési szintet válthatnak ki a hallgatókból.




**2. ábra:** Eszközök és gyakorlatok tetszés szerint – Eloszlás diagram

## 3.2. Témakörök elsajátításának mértéke és az alkalmazott eszközök és gyakorlatok közötti összefüggés

A félév végi visszajelző kérdőívben a hallgatók visszajeleztek arra vonatkozóan is, hogy mennyire tudták megérteni, elsajátítani az egyes témaköröket. A 3. táblázat az E. Átlag oszlopban rögzíti az elsajátítás becsült értékeinek átlagát. A táblázat ezen átlagok csökkenő sorrendje szerint van rendezve. Kíváncsiak voltunk arra, hogy van-e összefüggés aközött, hogy a hallgatók mennyire tudtak elsajátítani egy témakört és milyen eszközökkel és módszerekkel történt ennek átadása. A GY.Átlag oszlopban egy olyan súlyozott átlag van feltüntetve, melyben az alkalmazott eszközök és gyakorlatok hasznosságának átlaga a témakör leadásában alkalmazott aránya szerint érvényesül. Például: a *Kommunikáció az ügyféllel* témakör leadása során 60%-ban használtunk *Papírkártyák párosítása* (hasznossági átlag: 4.09) gyakorlatot és 40%-ban *Csapatmunkát* (hasznossági átlag: 4.16). Ebben az esetben GY.Átlag= 0.6*4.09+0.4*4.16 = 4.12. Az utolsó oszlop az E.Átlag és a GY.Átlag különbségét tartalmazza, vagyis összefüggést kíván feltárni a témakörök elsajátításának mértéke és a témakörök átadása során alkalmazott eszközök és gyakorlatok általános hasznosságát illetően. Ilyen összefüggés nem rajzolódik ki egyértelmű módon, a legnagyobb eltérések a „Projekt definíciója, projekt négyszög" (0.51) és a „Szoftverfejlesztési módszertanok" (-0.54) között van. Úgy tűnik, hogy a témakörök elsajátítását nem csak az alkalmazott eszközök és módszerek befolyásolják. A fent említett két végletet tekintve feltehető, hogy fontos lehet, hogy milyen előzetes tapasztalatokkal rendelkeznek a hallgatók a témakört illetően, vagy mennyire egyszerű egy fogalom, hiszen az E.Átlag-Gy.Átlag a közismertebb témakörök esetében a legnagyobb (pl. Projekt definíciója, projekt négyszög; Kommunikáció az ügyféllel) és a számukra teljesen új fogalmak esetében a legkisebb (pl. Szoftverfejlesztési módszertanok, Scrum szerepkörök).

Kiszámoltuk az E.Átlag és a Gy.Átlag között korrelációs együtthatót is, melynek értéke 0.26, mely ugyancsak inkább eltérésre, mint egyezésre utal a témakörök elsajátítása és az alkalmazott esz-

közök és gyakorlatok általános hasznosságát illetően. Egy bizonyos eszköz vagy gyakorlat alkalmazása tehát nem ugyanolyan mértékben hasznos bármely témakör átadása esetén, meg kell válogatnunk, hogy mikor és mire használjuk őket.

| Témakörök | E. Átlag | Alkalmazott eszközök és gyakorlatok | GY. Átlag | E.-Gy. |
|---|---|---|---|---|
| Projekt definíciója, projekt négyszög | 4.40 | Tábla | 3.89 | **0.51** |
| Kommunikáció az ügyféllel | 4.20 | Papírkártyák párosítása, Csapatmunka | 4.12 | 0.08 |
| DISC személyiség modell | 4.15 | Vetítés, Önismereti teszt | 4.28 | -0.13 |
| Időelemzés | 4.10 | Tábla, Számítási feladatok csoportban | 3.91 | 0.19 |
| Projekt életciklusa | 4.05 | Tábla | 3.89 | 0.16 |
| Projekt szerepek | 3.95 | Papírkártyák csoportosítása | 3.95 | 0.00 |
| A hatékony csapatok közös tulajdonságai | 3.95 | Tábla | 3.89 | 0.06 |
| Szervezeti modellek | 3.88 | Modellezés | 4.09 | -0.21 |
| A projekt környezete a vállalat - országok és vállalatok kultúrája | 3.83 | Csapatmunka, Szavazás | 4.03 | -0.20 |
| Agilis módszertanok, Scrum módszertan | 3.81 | Tábla, Internethasználat | 3.86 | -0.05 |
| Scrum szerepkörök | 3.72 | Papírkártyák csoportosítása | 3.95 | -0.23 |
| Szoftverfejlesztési módszertanok | 3.35 | Tábla | 3.89 | **-0.54** |

**3. táblázat:** Témakörök elsajátításának mértéke és az alkalmazott eszközök és gyakorlatok közötti összefüggés

## 3.3. Visszajelzések

A kérdőív utolsó pontjában a hallgatóknak volt lehetőségük szabadon, néhány mondatban visszajelezni bármilyen észrevételt a tárgy kapcsán. 33 hallgató élt ezzel a lehetőséggel. A visszajelzések többnyire pozitívek voltak. Néhány gyakrabban kiemelt témakört említünk meg.

Legtöbben (öten) az óra interaktív jellegét méltatták: „A legpozitívabb az volt, hogy voltak interaktív feladatok is és aktívan részt lehetett venni az órákon, véleményt nyilvánítani, megvitatni dolgokat." „Tetszett, hogy kifejezhettük a véleményünket.", „Az interakciók gyakorlása, beszéltetés fontos és jó, illetve a szemléltetések hasznosak voltak. Mindezek által a legtöbb óra élénken él bennem."

A másik kiemelt (öt említés alapján) a *Projektmunka* volt, melyre viszont vegyes visszajelzések érkeztek: „Az, hogy Csapatban kellett dolgozni, nagyon tetszett." „Meglepően nehéz volt jól együttműködni a csapattal." „Mindenképpen egy hasznos tapasztalat volt egy csapat részeként dolgozni, látni a működését, nehézségeit."

Hárman kiemelték, hogy fontos volt számukra, hogy az óra keretein belül jobban megismerhették egymást: „Tetszett, hogy ez a tárgy által megismerhettem pár olyan hallgatótársam, akikkel amúgy nem ismerkedtem volna."

Volt olyan visszajelzés, mely a jegyzetre vonatkozott: A jegyzeteket előbb is megkaphattuk volna, mert az sokat segített volna a félév közbeni anyag megértésében."

Néhány visszajelzés, mely az órák összességére vonatkozott: „Nagyon jól felépített óravázlatok, sok gyakorlati téma (pl. fonál, lift). Szívesen járok be órára." „Köszönjük az órákat! Tetszett, hogy bár egy elég száraz anyagról beszélünk, megpróbáltad minél színesebbé és változatosabbá tenni az órákat a különböző módszertani technikákkal és eszközökkel." „Csak így tovább!"

A fent idézettekből következtethetünk arra, hogy a hallgatók értékelik az eszközök és gyakorlatok sokszínűségét és azt, hogy aktívan részt vehetnek az órán. Az oktató tehát bátran teret engedhet ezeknek, mert motiváló hatással lesznek, akár a közösség épülésére tett hatásuk által is. A csapatos projektmunka már megosztóbb, de mindenképpen emlékezetes. A jegyzet hasznossága egyértelmű a statisztikák alapján is, viszont a kifejtős visszajelzések alapján felmerül a lehetőség, hogy ezt korábban megkapják a hallgatók, ne csak a félév végén. A korábbiakban említettük, hogy ehhez a megoldáshoz az a félelmünk társul, hogy ez esetben a hallgatók kevésbé lesznek figyelmesek az óra folyamán, nem használják egyáltalán a tankönyvet, a szükséges minimum tudásra játszva (ugyanis ez a jegyzet tartalma) próbálják megszerezni érdemjegyüket. Amennyiben ez megalapozott feltételezésnek bizonyulna is, akkor is az oktató kell eldöntse, hogy ezt támogatni tudja-e.

# 4. Összegzés

A „Munkaszervezés, projekt kommunikáció" tárgy az ELTE Informatikai Kar felsőoktatási szakképzéses hallgatóinak szól. A címzett hallgatók elenyésző szoftverfejlesztői tudással rendelkeznek. A tárgy célja, hogy projekt működéssel, működtetéssel kapcsolatos tudást nyújtson számukra.

A tárgy 2019-es megvalósulása során 6 oktatási eszköz és 14 gyakorlat típus állt az oktatás szolgálatában. A visszajelzések alapján a hallgatók értékelték, hogy ezáltal az órák változatosak és interaktívak voltak. Sokukban nyomot hagyott a csoportos *Projektmunka*, mely bár kihívásokat és nehézségeket is okozott, mindenképpen hasznosnak bizonyult: tanultak általa csapatban dolgozni. A klaszszikus *Projektmunka Projekt konzultációval* egészült ki, mely során a hallgatói csoportok projektjeikre javítási javaslatokat kaphattak egy héttel az értékelést megelőzően. A hallgatók egymásnak is visszajelzéseket adhattak az egyes gyakorlatok során, mely szokást nagyon hasznosnak ítéltek. Nagy mértékben tetszettek nekik a *Csapatmunka* és az *Önismereti tesztek kitöltése* is. A *Szituációs gyakorlatok* és az *Internethasználat* voltak a legmegosztóbb gyakorlatok, ezek kapták a legtöbb alacsony értékelést is. A statisztikákból kiderült, hogy az egyes témakörök elsajátításának sikere nem korrelál egyértelműen az átadásukra használt eszközökkel és gyakorlat típusokkal. Ez arra enged következtetni, hogy nem mindegy, hogy milyen témakör kapcsán alkalmazunk milyen eszközöket és gyakorlatokat, illetve az elsajátítás sikere más tényezőktől is függhet, mint például: milyen előzetes tapasztalatokkal rendelkeznek a hallgatók a témakört illetően. A tananyagot tömören összefoglaló, oktató által összeállított *Jegyzet* képezte a legnagyobb sikert a hallgatók körében, ehhez képest az *E-tankönyv* nagyon lemaradt a rangsorban. Eldöntendő kérdés marad, hogy a *Jegyzet* sikerét tovább növelné vagy csökkentené, ha nem csak a félév végén, hanem a félév elejétől rendelkezésre állna.

Ezek alapján tehát kifejezetten ajánljuk a *Jegyzet, Projektmunka, Projekt konzultáció, Önismereti tesztek, Csapatmunka* eszközök és gyakorlatok beépítését az órákba, annak meggondolását követően, hogy ezek mely témakör átadásában lehetnek a leghasznosabbak. A *Szituációs gyakorlatot* illetően, bár nagyon hasznosnak tartjuk őket, figyelmeztetjük az oktatókat, hogy megosztó fogadtatásra számítsanak. Összességében pedig indokoltnak és sikeresnek tartjuk sok különböző eszköz és gyakorlat alkalmazását egyazon tárgy keretein belül. A legtöbb eszközt és gyakorlatot a hallgatók hasznosnak értékelik és méltányolják a változatosságot, mely interaktivitást és motivációt szül.

## 5. Köszönetnyilvánítás

A kutatási projekt az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósult meg (EFOP-3.6.3-VEKOP-16-2017-00002). Köszönjük a támogatást!

## Irodalom

1. P. Sarmasági: *DISC assassment usage in school talent management*. In: dr. Andor Abonyi-Tóth, prof. Ing. Veronika Stoffová, doc. dr. hab. László Zsakó (ed.): DidMatTech 2020, Budapest (2020) 183–202
2. I. Ibrahim: *Teaching Project Management for IT Students: Methods and Approaches*. International Conference on Education and Management Technology, Singapore (2011) 185–191
3. H. Taylor, J. P. Woelfer: *Critical Skills for It Project Management and How They are Learned. `09* SIGMIS-CPR ACM, Limerick Ireland (2009)
4. M. Paasivaara, V. T. Heikkilä, C. Lassenius, T. Toivola: *Teaching students scrum using LEGO blocks*. ICSE 2014, Hyderabad India (2009) 382–391
5. J.G. Kuhl: *Incorporation of Agile Development Methodology into a Capstone Software Engineering Project Course*. In: University of Iowa - Iowa Research Online (ed.): 2014 ASEE North Midwest Section Conference, Iowa city (2014)
6. T. Smith, K.M.L. Cooper, C.S. Longstreet: *Software Engineering Senior Design Course: Experiences with Agile Game Development in a Capstone Project*. In: ACM New York (ed.): ICSE 2011, Waikiki, Honolulu (2011)
7. C. Anslow, F. Maurer: *An Experience Report at Teaching a Group Based Agile Software Development Project Course*. In: ACM (ed.): SIGCSE'15, Kansas (2015)
8. E. Ilyés: *Esettanulmány: Agilis szoftverfejlesztés egyetemi kurzuson*. In: Webdidaktika Alapítvány (ed.): INFODIDACT 2017, Zamárdi (2017)
9. E. Ilyés: *Iparorientált munkaszervezés egyetemi kurzuson*. SzámOkt 2018, Tusnádfürdő (2018)
10. E. Ilyés, B. Pintér, R. Szendrei, M. Cserép: *A Szoftvertechnológia tárgy agilisra vált*. In: Webdidaktika Alapítvány (ed.): INFODIDACT 2019, Zamárdi (2019)
11. Sz. Kolozsvári, A. Kiss, B. Molnár: *Agilis módszertan bevezetése az egyetemi projektlaborban*. In: Webdidaktika Alapítvány (ed.): INFODIDACT 2014, Zamárdi (2014)
12. E. Kovács, G. Kovásznai, G. Kusper: *Project Lab and Project Work Issues and Experiences at the Eszterházy Károly Collage*. XXI. DIDMATTECH 2008, Eger (2008)

# Az Algoritmusok és adatszerkezetek I. kurzus megújítása

Kovácsné Pusztai Kinga

kinga@inf.elte.hu
ELTE IK

**Absztrakt**. A felgyorsult világunkban a néhány éve még jól bevált pedagógiai módszerek egy része mára már elavulttá vált. Helyükre olyan új módszereket kell találnunk, mely a ma felnövő generáció szemléletéhez kapcsolódik. Ők már egy online világba születtek, ezért gondolkodásmódjuk, illetve életmódjuk gyökeresen megváltozott.
Egy megelőző kutatásomban azzal foglalkoztam, hogyan lehetne egy elméleti kurzust úgy megújítani, hogy a hallgatók könnyebben és élvezetesebben sajátíthassák el az ismereteket. Mivel a kutatás során jelentős eredményeket értem el, a megkezdett kutatást folytattam.
Ebben a cikkben a már sikerrel használt módszereket ültetem át az Algoritmusok és adatszerkezetek I. kurzusra, illetve további új módszereket mutatok be.

**Kulcsszavak:** gamification, informatika oktatás, számítógépes gondolkodás, edutainment, algoritmusok

## 1. Bevezetés

### 1.1. A kutatás előzménye

Egy megelőző kutatásban kísérletet tettem az Algoritmusok és adatszerkezetek II. kurzus innoválására. Ezt több szempont miatt is szükségesnek éreztem. Egyrészt a tárgy elméleti jellege, illetve korábbi megjelenése miatt (a régi tantervhez képest ez a tárgy fél évvel előrébb csúszott, azaz már a tanulmányai 2. illetve 3. félévében hallgatják,) a hallgatók számára egyre nehezebben teljesíthetővé vált. Másrészt pedig, a mai egyetemisták már a Z illetve *alfa generáció* tagjai, akik már úgy nőttek fel, hogy gyermekkoruktól elérhető volt az internet. Számukra magától értetődő a személyes kommunikációs eszközök használata, okostelefonnal kelnek és fekszenek, mindig elérhetőek és folyamatosan kapcsolatban vannak egymással az online térben. Könnyen kezelik az információk gyors áramlását, tevékenységeiket gyakran váltogatják „multitasking" során. Így a hagyományos, frontális eszközökkel nehéz lekötni a figyelmüket. A vizuális megjelenítést részesítik előnyben, szemben a hosszú, tagolatlan szövegekkel.

### 1.2. Az eredmény kiértékelése

Az eredmény a várakozásaimat is felülmúlta. A hallgatók véleményét online kérdőív formájában vizsgáltam, illetve a hallgatói részvétel mértéke is sokatmondó volt. Ezek alapján egyértelműen elmondható, hogy a hallgatóknak tetszettek az újításaim.

Azonban ennél sokkal fontosabbnak gondoltam, hogy a hallgatók jegyein is látszódjon az újítás pozitív hatása, ezért itt is végeztem kutatásokat. A kísérleti csoportjaim eredményét olyan csoportok eredményeivel , akik kurzusaiban nem jelentek meg az általam bevezetett innovatív elemek. A kurzus teljesítése kétlépcsős, a hallgatóknak először gyakorlati jegyet kell szerezniük, majd vizsgázniuk kell. Bár az általam tartott kurzus elsődleges célja a gyakorlati jegy megszerzése volt, az összehasonlítást mindkét jegy eredményeire elvégeztem. (Az eredmények összehasonlítását összefoglalóan az 1. ábra szemlélteti.)

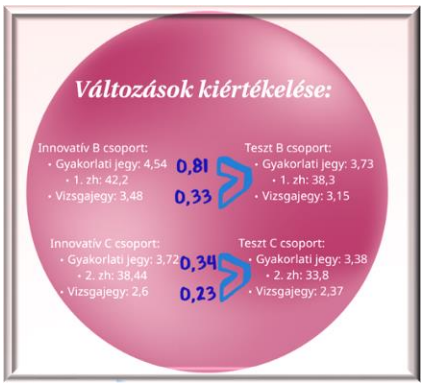
### 1.2.1. Gyakorlati jegyek összehasonlítása

A félév során két csoport algoritmus kurzusait innováltam. Kontroll csoportnak az előző év csoportjait használtam, akik még hagyományosan hallgatták a kurzust. A két „Innovatív" és a két „Teszt" csoportjaim hasonló típusúak voltak, az egyik B szakirányos fix csoportú, a másik C szakirányos normál csoportú. (A fix csoportokba csak olyan hallgatók jelentkezhetnek, akiknek minden vizsgájuk sikerült. Ha valakinek nem sikerül minden vizsga, akkor csak normál csoportba jelentkezhet. Általában a jobban teljesítő hallgatók választják a B szakirányt, a C pedig az általános szakirány.) Azaz elmondható, hogy mindkét évfolyamon volt egy átlagosnál jobban teljesítő és egy rosszabban teljesítő csoportom.

A félév során a hallgatóknak zh-t kell írniuk. A „Teszt" csoport zh-ja 5 feladatból állt, melyre elvileg 1,5 órát kaptak, de aki kért időhosszabbítást, annak engedélyeztem. Az „Innovatív" csoport zh-ja 6 feladatból állt, ugyanúgy 1,5 órát kaptak, de az időtartam hosszabbítása nem automatikusan járt, hanem feltételhez kötött volt. (Az órákon szerezhetők voltak úgynevezett „lehetőségek", amelyeket időhosszabbításra is felhasználhattak a hallgatók.) A zh 5 feladata hasonló volt a „Teszt csoporthoz, a 6. feladatban pedig egy tanult technika alkalmazásával kellett algoritmust írni. (A hallgatóknak ez a feladattípus megy a legnehezebben.) Elmondható tehát, hogy az „Innovatív" csoportnak ugyanannyi idő alatt nehezebb zh-t kellett megoldaniuk a „Teszt" csoporttal szemben. A B szakirányos fix csoporton a „Teszt" csoport hallgatói átlagosan 38,3 pontot értek el, míg az „Innovatív" csoport tagjainak átlageredménye 42,4 volt. A C szakirányos normál csoportos hallgatóknál a teszt csoport átlagosan 33,8 pontot ért el, míg az innovatív csoport 38,44 pontot szerzett. (1. ábra)

A gyakorlati jegyek a zh-k eredményeinél nagyobb eltérést mutattak: A B szakirányos fix csoportos hallgatóknál a teszt csoport átlaga 3,73 volt, az innovatív csoport átlaga pedig 4,54, azaz majdnem egy (0,81) jeggyel jobban teljesítettek az innovatív csoport diákjai. A C normál csoportnál is az innovatív csoport teljesített jobban: A teszt csoport átlaga 3,38, míg az innovatív csoport átlaga: 3,72. (Az eltérés itt csak 0,34.)

Az eredményeken az is látszik, hogy a hallgatók éltek az opcionális pontszerzési lehetőségekkel, ezáltal a félév során a kurzusba több munkát fektettek be.




**1. ábra:** A kurzus innoválásának eredményei

### 1.2.2. Vizsgajegyek összehasonlítása

A vizsgajegyek vizsgálatánál más kontroll csoportot választottam. Úgy gondoltam, hogy a felmérésem jobban tükrözi a valóságot, ha a vizsgált csoportok teljesítményeit egyező feladatokon mérem össze, így kontroll csoportnak a teljes évfolyamot vettem.

A B szakirányon az évfolyamátlag (, azaz a „Teszt" csoport eredménye) 3,15 volt, az „Innovatív" csoport átlaga pedig 3,48, azaz közel egyharmad (0,33) jeggyel kaptak jobb érdemjegyet. (1. ábra)

A C szakirányon az évfolyamátlag (, azaz a „Teszt" csoport eredménye) 2,37 volt, az „Innovatív" csoport átlaga pedig 2,6, ami közel egynegyed (0,23) jegynövekedést jelent. (Annak ellenére, hogy itt az innovatív csoport normál, az évfolyamban pedig több fix csoport szerepel.) (1. ábra)

### 1.3. A következmény

Az eredmény hatására eldöntöttem, hogy a tárgyat megelőző, Algoritmusok és adatszerkezetek I. tárgyat is innoválom, erről a kísérletről szól ez a cikk.

A múlt félévben sikeresen alkalmazott módszereken túl további új módszereket alkalmaztam a kurzus megújításakor. E félév során is folyamatosan monitorizáltam a hallgatók véleményét, de a COVID járvány miatt a hallgatói teljesítménynél nem tudtam olyan vizsgálatot végezni, mint az előző félévben.

## 2. A tanórán alkalmazott módszerek

### 2.1. A gamification

A gamification[1] szó a game (játék) és a fication (valamilyenné alakítás) szóból származik, magyarul játékosításnak, eljátékosításnak, vagy gamifikációnak is szokták nevezni.

Manapság jónéhány definíció létezik, de Deterding 2011-ben alkotott definíciója[2] vált a leggyakoribbá, mely szerint a gamification **„a játéktervezési elemek használata játékon kívüli kontextusban”**.

A gamification definíciójában két nagyon fontos fogalom jelenik meg, a *játékelemek* és a *játékmechanizmusok*, melyet gyakran összevonva játéktervezési technikának is neveznek. A játékelemeken a hagyományos és videojátékokból vett eszközöket értjük, a játékmechanizmusokon a játékok működési elvének alkalmazását. Az eszközök természetesen csak akkor működnek hatékonyan, ha a játék mechanizmusai adottak: a játék önkéntes, sikert ígérő, átlátható és kellően lehatárolt (ideje van).

A definícióban szereplő „játékon kívüli kontextus” pedig arra utal, hogy más a célja a játéknak és más a játékosításnak. Játék és játékosítás között a legnagyobb különbség, hogy valamit a játékban, a játék élvezetéért csinálunk, a játékosított alkalmazásban pedig a való életben egy előre meghatározott cél elérésének érdekében.

Az oktatásban is számos példát találunk a gamification alkalmazására. Számos ingyenesen elérhető szoftver létezik, melyek segítségünkre lehetnek az oktatás játékosításában. Ilyen app-ok például a *Socrative*, vagy az *Edmodo*. Ezekkel a szoftverekkel könnyen előállíthatunk gyorstesztet, feladatot, melyet a diákok az okostelefonjukon oldanak meg, azaz a kedvenc eszközüket órai aktivitásra használják. Az ilyen app-ok alkalmazásával a diákok motiváltabbakká válnak, nő az interaktivitásuk, és sokkal gyorsabban és pontosabban kapunk visszacsatolást a tananyag megértéséről. Ezen eszközök azonban nem csak a csoport munkáját támogatják, hanem nagyon fontosak lehetnek az egyéni tanulási élményben is.

Az értékelést is lehet gamifikálni. Prievara Tibor[18] nevéhez kötődik a pontrendszer bevezetése a hagyományos jegy alapú értékelés helyett. Számos irodalom[16,17,18] számol be a pontozásos módszer előnyeiről mind a diákok, mind a tanárok szemszögéből. A diákok az órán motiváltabbak, aktívabbak, ennek hatására jobban átlátták saját tevékenységeiket, illetve tudatosabban tudtak maguk elé állítani célokat. A tanárok sűrűbb és értékelhetőbb visszacsatolást kaptak, így jobban látták a tananyag elsajátításának mértékét, illetve a tanulói igényt.

A gamification alkalmazása során átveszünk olyan elemeket a játékok rendszeréből, amelyek segítségével motiváltabbá tehetjük diákjainkat, csökkenthetjük a rájuk nehezedő stresszt, valamint segíthetünk nekik, hogy önállóbbá váljanak, és valóban részesei legyenek a tanulás során meghozandó döntéseknek. Nádory Gergely[9] a következő elemeket nevezi meg:

- Önállóság
- Unaloműző

- Célok
- Siker és kudarc
- Azonnali visszajelzés

A gamification egyik alesete az edutainment. Az *Edutainment* [19] az „*oktatás*" és a „*szórakoztatás*" szóhasználat olyan technológiákra és szoftvertermékekre utal, amelyek valamilyen módon egyesítik az oktatást és a szórakoztatást. Az Edutainment technológia számos formában jön létre. Egy streaming video platform vagy előre csomagolt tanulási termék kategorizálható edutainmentként, ha szórakoztató és oktatási értékkel rendelkezik.

## 2.2. A digitális történetmesélés

A digitális történetmesélés[12] (digital storytelling, továbbiakban DST) egy olyan új tanulásszervezési eljárás, melyben **a hagyományos történetmesélés ötvöződik a digitális eszközhasználattal**. Lényege, hogy a tanulók nem öncélúan alkalmazzák a digitális eszközöket, hanem egyedi elbeszéléseket, sajátos multimédia alkotásokat hoznak létre, melyek felkeltik tanulótársaik figyelmét, lelkesedését és kommunikációt generálnak a feldolgozott témában a tanulóközösségen belül. Többszörösen bizonyított a DST tanulói motivációra[13, 14] és teljesítményre[15] gyakorolt pozitív hatása, fejleszti a tanulók problémamegoldó képességét, az önálló tanulás képességét, illetve a kritikai gondolkodás kialakulását is.

A digitális történetmesélés a diákok körében is nagyon népszerű, mivel olyan tevékenységeket használ, melyeket a diákok a kortárskapcsolataikban amúgy is csinálnak: képeket, videókat, történeteket osztanak meg egymással. A módszer alkalmazása során azonban ezen tevékenységek kiegészülnek egy önálló tanulási, gondolkodási fázissal, illetve egymás munkáinak kritikus, konstruktív értelmezésével.

A DST tanórai felhasználása esetében a módszer lépéseit a következő öt nagyobb szakaszban érdemes definiálni[5].

1. A diákok először megírják a digitális történetük magját adó szöveget. A feldolgozni kívánt témához kapcsolódó, meglévő ismereteiket, élményeiket kiegészíthetik különböző forrásokból válogatott információkkal. A források felkutatásában, az információk szelekciójában segítséget nyújthat a facilitátor pedagógus, a kinyert adatok szintetizálása, szöveggé formálása azonban már a tanuló feladata. A szövegalkotási folyamatot végigkíséri a konzultáció lehetősége.
2. A második szakaszban a diákok felmondják megírt szövegüket, azaz létrehoznak egy hangfájlt.
3. A harmadik szakaszban a diákok megtervezik és elkészítik a szöveghez szükséges képi elemeket. Ezek lehetnek saját készítésű fotók, illusztrációk, digitalizálhatnak papíron lévő, régi fényképeket, dokumentumokat is. A képanyagban megjelenhetnek interneten talált, jogtiszta felvételek is. A tanulók figyelmét fel kell hívni arra, hogy a képi és szöveges forrásokra hivatkozzanak digitális történetük végén. Ezen a ponton érdemes elgondolkodni azon, hogy a képek és a szövegek könnyebb összehangolása érdekében a tanulók storyboard-ot készítsenek, melyben pontosan megtervezhető a képek sorrendje.
4. A negyedik szakasz a vágás, amikor egy tetszőlegesen választott videószerkesztő szoftver segítségével (például: Microsoft Movie Maker, Sony Vegas, illetve az okoseszközökkel is használható online vágóprogram-alkalmazások: WeVideo és Power Director) a tanulók összeállítják digitális történetüket.
5. Az utolsó fázis pedig az elkészült alkotások levetítése, megvitatása és értékelése.

Az egyes szakaszok lehetőséget adnak a tanulóknak kreativitásuk kibontakoztatására, továbbá a kooperatív munkára, mely egyre jobban előtérbe kerül a valós életben.

Az eljárás alkalmazhatósága kézenfekvő minden olyan tantárgyi tartalom tematizálása esetében, melyben létjogosultsága van a személyes elbeszélések megjelenésének[3]. Kérdés azonban, hogy hogyan vonható be a DST a természettudományos tantárgyak módszertanába? Lanszki[4] szerint DST-vel nemcsak egyéni történetek artikulációja valósítható meg, hanem tematikus tartalomfeldolgozás is. Természettudományos tantárgyak esetében feltételezhető, hogy – a diákok életkorából fakadóan – kevés egyéni élethelyzetet feltáró digitális történet születik. Szükségszerű tehát a digitális történetmesélés definícióját tágan értelmeznünk: nemcsak az egyéni élettörténeteket soroljuk a digitális történet osztályába, hanem **a módszer lépéseinek segítségével létrehozott, narrált audiovizuális prezentációkat is**. Így értelmezték a DST-t a Houston-i Egyetem tanárai is és ezt az értelmezést alkalmaztam én is az óráimon.

# 3. Változtatások az algoritmusok és adatszerkezetek I. egyetemi kurzuson

A megelőző kísérletemben elsősorban a gamification lehetőségeit használtam a félév során. A gamification elemeit ebben a félévben is bevezettem, mind a számonkérés, mind az órák menete terén. Ezen felül a digitális történetmesélés irányába is elkezdtem kutatásokat végezni, melyeket megpróbáltam a félév során alkalmazni.

Kezdeti lépésként ebben a félévben is behoztam a **pontrendszert**, azaz értékelésnél nem csak a zárthelyi eredmények számítanak. A hallgatóknak két zárthelyi dolgozatot kell írniuk, mindegyik 60 pontos. Mindkét zh-n a kötelezően elérendő minimum a 20 pont. Továbbá minden órán kapnak házi feladatot, mely megoldása nem kötelező, azonban plusz pontszerzési lehetőség. Összesen 20 kiegészítő pontot szerezhetnek a házi feladatokból. Ezen felül kiadtam egy komplex projektmunkát, melyben több hallgatónak több rendezési feladatot, többféleképpen kellett kidolgozni. Erre is max. 20 pontot lehetett szerezni, de itt egy csoport összesen kapott pontot, és nekik kellett dönteni, hogy a kapott pontot milyen arányban osztják szét maguk között. Továbbá pontokkal jutalmazom azokat is, akik a tananyaghoz kapcsolódó témában komolyabb kutatásokat végeznek, és ezt velünk valamilyen formában (pl. videók készítése) megosztják. A félév végi jegyüket ez alapján a pontjuk összesítéséből számolom. Ha valaki nem éri el a zh-kon a kötelező minimumot, akkor pótzh-t kell írnia, ezt nem lehet egyéb pontozással kiváltani.

Az órák menetén is próbáltam változtatni, a COVID járvány előtt a Kahoot használatával színesítettem az órát, a járvány alatt pedig animált ppt-ket alkalmaztam.

A magyarázataim mellé az egyes témakörökhöz rövid youtube videókat kerestem, melyek segítségével később, a zh-ra, illetve vizsgára készülésnél is fel tudták eleveníteni ismereteiket. A videók egy részét a hallgatók készítették.

Ezen felül – az előző félévhez hasonlóan - minden órához készítettem egy – általában játékos – edutainment alkalmazást, melyeket szorgalmi feladatként adtam fel, a honlapomról elérhető.[7] Az előző féléves kutatásomban több keretrendszert alkalmaztam, azonban a naplózás miatt úgy döntöttem, hogy lehetőség szerint lecsökkentem ezeknek számát. Így a szorgalmi feladatok nagy része LearningApps-ben készült, ahol lehetőség van osztály létrehozására. Ezáltal a hallgatónak egyszer kell csak regisztrálnia és automatikusan eléri azokat a tankockákat, amiket megosztok vele, illetve automatikusan naplózza is a teljesítményét.

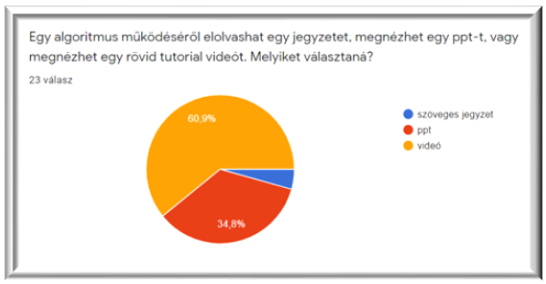
## 3.1. Tutorial videók, illetve animációk

A www.algoanim.ide.sk honlapon számos algoritmusnak illetve adatszerkezetnek található animációja illetve vizualizációja. Innen kölcsönöztem a buborékrendezés, beszúrórendezés, összefésülő rendezés, gyorsrendezés, lista, bináris keresőfák műveletei, lineáris idejű rendezések, kupacrendezés, illetve a hashelés algoritmusok animációját.

Ezen felül a rendezésekhez youtube videókat is kerestem és osztottam meg. Célom volt, hogy a rendezések témakört a hallgatók a megosztott segédanyagok mintájára önálló videót is készítsenek. A kezdeti lelkesedés nagy volt, több csoport is alakult, akik kiválasztották a bemutatandó rendezést, illetve néhányan el is jöttek konzultációra, ahol beszéltünk a megvalósítás módjáról. Sajnos azonban a karantén miatt a hallgatók leterheltsége nagyon megnőtt, illetve a projektmunka megnehezedett, így végül csak egyetlen színvonalas videó született meg, az is önálló munkában. (Itt megtekinthető ) Sajnos ez a munka csak a vizsgaidőszak elején született meg, így a hallgatók csak a vizsgára készülésnél tudták használni.

A hallgatók véleményét online kérdőív segítségével kutattam, melyet 23 hallgató töltött ki a következőképpen:

- A „Mennyire találja jó ötletnek, hogy az egyes algoritmusok magyarázatához a honlapomon elérhetővé tegyek tutorial videókat?” kérdést átlagosan 4,7-re értékelték (19 ötös, 1 négyes, és 3 hármas).
- „Egy algoritmus működéséről elolvashat egy jegyzetet, megnézhet egy ppt-t, vagy megnézhet egy rövid tutorial videót. Melyiket választaná?” kérdésre a hallgatók 61 % választotta a videót (2. ábra) és 56%-kuk (13 hallgató) meg is nézett legalább egy, általam megosztott videót. A kérdőív alapján átlagosan két videót néztek meg.



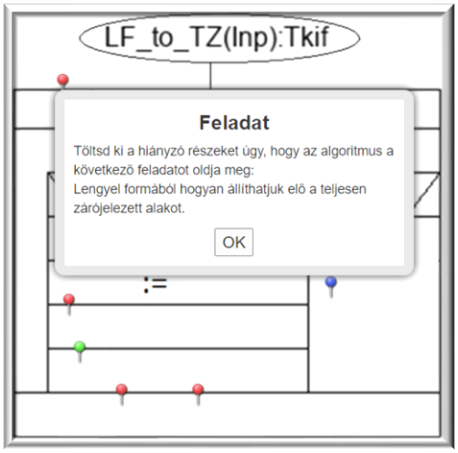

**2. ábra:** Hallgatói vélemény a videó alkalmazásáról

Kíváncsi voltam arra is, hogy a hallgatók hogyan értékelik a hallgatótársuk videóját. Sajnos csak kilencen (39,1%) nézték meg a videót, és ők átlagosan 4,78-ra (7 ötös 2 négyes) értékelték.

A kérdőív alapján látható, hogy fontos ebbe az új irányba nyitni az oktatás során.

## 3.2. Hozzárendelés a képeken a verem, illetve a sor adatszerkezet műveleteinek gyakoroltatására

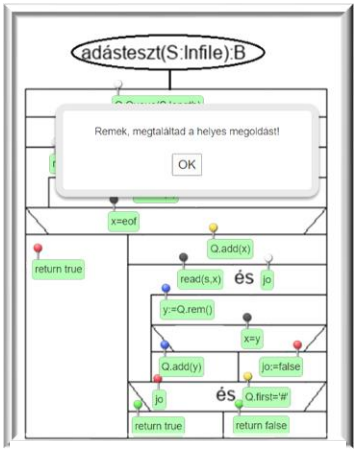
Egy algoritmus elkészítése mindig nehéz feladatnak számít, mivel ezt a feladattípust nem lehet gondolkodás nélkül megoldani. Általános tapasztalat, hogy a hallgatók egy része az ilyen gondolkodtató, algoritmuskészítő feladathoz a zh-n, vagy a vizsgán neki se kezd. Egy programozó szakember a munkája során azonban ezt a fajta ismeretet használja a legtöbbet, így fontosnak tartom, hogy az ilyen típusú feladatokkal megbarátkoztassuk a hallgatókat. Éppen ezért két „hozzárendelés a képeken” típusú feladatot is létrehoztam a LearningApps-ben, az egyik a verem, a másik pedig a sor adatszerkezet műveleteinek gyakorlásához kapcsolódik.




**3. ábra:** Verem adatszerkezet, kezdő kép

A verem adatszerkezet egyik legelterjedtebb alkalmazása a Lengyel forma, ezért ez a feladat (3. ábra) is a Lengyel formához kötődik, mely így hangzik: „*Töltsd ki a hiányzó részeket úgy, hogy az algoritmus a következő feladatot oldja meg: Lengyel formából hogyan állíthatjuk elő a teljesen zárójelezett alakot?*"

A sor adatszerkezet gyakoroltatására létrehozott feladat (4. ábra) szövege: *„Egy rádióadónak az a feladata, hogy újra és újra leadja ugyanazt a jelsorozatot. A jelsorozat végén egy szünet-jelet ad, a példákban ezt # jellel jelöltük. Az adást egy szekvenciális fájlban rögzítettük, készítsen algoritmust, mely sor segítségével eldönti, hogy volt-e hiba az adás során. Az input fájlt csak egyszer olvashatjuk végig.*




**4. ábra**: Sor adatszerkezet, a megoldás ellenőrzése

*Példák:*

*a a 0 b # a a 0 b # a a 0 b hibátlan*

*c 0 b # c 0 b # 0 b hibás*

*a a a b c hibátlan (nem volt ismétlés)*

*a b c d # a b c d # a b c d a hibás"*

A játék során a kezdőképen (2. ábra) megjelenik a feladat és egy üres stuktogram, különböző színű pálcikákkal. A hallgatók feladata, hogy a pálcikákra kattintva a megfelelő sort kiválasszák és beszúrják. Az összes pálcikán végighaladva, (vagy ha abba akarjuk hagyni a feladatot,) a képernyő jobb alsó sarkában található kék körben lévő pipára kattintva ellenőrizhetjük a megoldásunkat. A helyes megoldások zöld hátteret kapnak, a helytelenek pirosat. (4. ábra)

A hallgatók részvétele a feladatban, illetve a feladatok megtekintése nagyon érdekes. A vermes feladatnál az egyik csoportban 20-ból 18, a másik csoportból 23-ból 2, azaz összesen 43 hallgatóból 28 foglalkozott ezzel a feladattal, míg a feladat megtekintése 71 volt. A soros feladatnál a foglalkozási arány 16/20, illetve 3/23, azaz összesen a 43 hallgatóból 19 foglalkozott, ebből 18 meg is oldotta, és a feladat megtekintése 23 volt. A megtekintések közötti nagy különbség oka talán az, hogy a 2. feladat hetének közepén jött a karantén. Az első feladat adatainak alapján két dologra lehet következtetni, biztosan volt olyan hallgató, aki nem egyszer futott neki a feladatnak, illetve valószínűleg volt olyan hallgató is, aki megnézte a feladatot, de nehéznek találta, ezért nem foglalkozott vele.

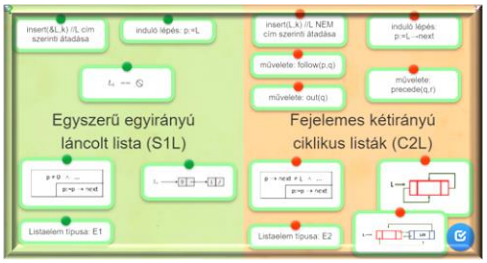
Az online kérdőívet az első feladatnál 17 hallgató töltötte ki, átlagosan 4,5-re értékelték, és egy hallgató kivételével mindenki hasznosnak találta. Ez a feladat több szöveges értékelést is kapott, ezek a következők voltak: *„Nagyon hasznos volt az algoritmus megértéséhez és megtanulásához", „Ezek a fajta feladatok a legjobbak", „Sokkal jobb a LearningApps-ben stukit kiegészíteni mint a canvasben, sokkal! Szerintem ilyen jellegű feladatok jók, mivel látom a stukiban amit kiválaszottam és gondolkodtató, de ha elveszve érzem magam, hogy mi kellhet oda akkor is van 4 válasz, amiből azért könyebb kikövetkeztetni, hogy melyik lehet a jó."*

A második feladatot csak 12 hallgató értékelt átlagosan 5-re, és mindenki hasznosnak találta. Erre a feladatra szöveges értékelés nem érkezett.

## 3.3. Csoportba rendezős játék listák témakör gyakoroltatására

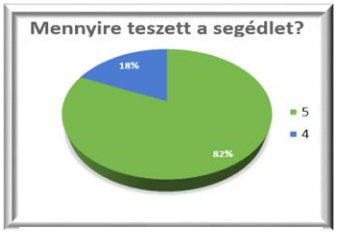
Mivel a kurzus egyik fő témaköre a lista adatszerkezet használata, és többféle listát tanulunk, fontosnak tartottam egy olyan app létrehozását, amelyik segít rendszerezni az egyes listák tulajdonságait, ezért a LearningApps-ben létrehoztam egy csoportba rendezős játékot. A játék kezdetén (5. ábra) két csoport különböző színben jelenik meg, az egyik csoport az egyszerű egyirányú láncolt lista (S1L), a másik csoport pedig a fejelemes kétirányú ciklikus lista (C2L). Középen fogjuk kapni az egyes meghatározásokat, amelyeket a megfelelő csoportba kell húznunk. Ha elfogytak a kártyák, (vagy ha abba akarjuk hagyni a játékot,) akkor a képernyő jobb alsó sarkában található kék körben lévő pipára kattintva ellenőrizhetjük a megoldásunkat. A helyes megoldások zöld keretet kapnak, a helytelenek pirosat. (6. ábra)


**5. ábra:** Játék kezdete




**6. ábra:** Játék vége

A hallgatók részvétele a feladatban átlagos volt, a 43 hallgatóból 16 oldotta meg (14/20, illetve 2/23), és 11 hallgató véleményezte is a feladatot. A hallgatóknak, átlagosan 4,81-re értékelték (7. ábra), mindenkit segített a tanulásban, valamint több pozitív szöveges véleményt is kapott. Ezekből néhány: *„Ez nagyon tetszett :D”*, *„Ez is nagyon jó, mindegyik témakörben lehetne egy ilyen”*.




**7. ábra:** Hallgatói vélemény

## 3.4. Kahoot alkalmazása összefoglalásra

Az I. zh előtti héten az óra elején egy Kahoot-tal foglaltunk össze. Ezt nagyon hasznosnak tartom, mivel egyszerű az alkalmazása, azonban a hallgatók nagyon szeretik, feldobja az óra menetét, szembesíti a hallgatókat a hiányosságaikkal, illetve használata után a hallgatók jobban szoktak figyelni. Ezen felül számomra is küld visszajelzést az egyes kérdések megoldásainak, illetve a hallgatók teljesítményeinek sikerességéről.

Egy Kahoot kvíz órai felhasználásához nem elegendő a tanári gép és projektor, minden hallgatónak szükség van egy telefon vagy laptop használatára. Kezdetben a tanári gépen a Kahoot.com oldalról elindítom az alkalmazást, amely kivetít egy game pin-t. A hallgatók a Kahoot.it oldalon a kivetített pin, illetve egy nickname beírásával csatlakoznak a játékhoz. Ezek után indul a játék. Minden feladatot kivetítünk a hallgatóságnak (8. ábra), ők pedig válaszolnak a telefonjuk segítségével, ahol csak a válaszok színeit és ábráit látják (9. ábra). Egy-egy feladatra adott mennyiségű idő (általában 20 másodperc) áll rendelkezésre. A kérdésre maximum 4 választ lehet beállítani, ezek közül több helyes megoldás is lehet. A helyesen megoldott feladatokért a hallgatók pontot kapnak, amely a gyorsaságot is figyelembe veszi. Minden egyes feladat után mutatja a szerzett pontok alapján felállított dobogó helyezettjeit (1-3), illetve lehetőségünk van a kép újbóli kivetítésére és a feladat válaszának megbeszélésére.

Az Algoritmusok I. Kahoot alkalmazás 4 kvízkérdést tartalmazott és mindegyiknek 4 lehetséges válasza volt. A kérdések sorrendje és a válaszok sorrendje is véletlenszerű. A 4 kérdésből 3 a következő heti zh egy-egy feladatának témáját dolgozta fel, a 4. kérdés pedig két feladathoz kapcsolódott.


**8. ábra:** Kahoot feladat kivetítve

Mindkét csoportban 17-en oldották meg az órán ezt a Kahoot alkalmazást, azonban más eredménnyel. A feladatok sikeressége az aktívabb csoportban: 18%, 41%, 71%, illetve 35% volt, míg a másik csoportban: 24%, 24%, 35%, 41%., ami arra enged következtetni, hogy az 1. kérdés lehetett a legnehezebb és a 3. kérdés a legkönnyebb. Az aktívabb csoportban két hallgató 100%-an oldotta meg, és két hallgató 0%-an, míg a kevésbé aktív csoportban nem született teljesen jó megoldás, és sajnos 5 hallgatónak nem sikerült egy jó választ sem adnia.

**9. ábra:** Kahoot feladat a játékos telefonján

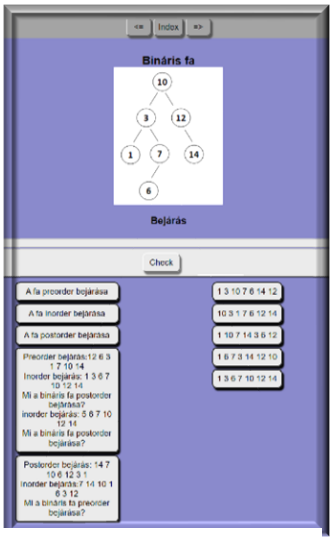
Az órai Kahoot-ból (live változat) aztán létrehoztam egy olyan önállóan használható kvízt, amelyet a hallgatók otthon bármennyiszer egyedül is meg tudnak oldani egy előre megadott időintervallum között. Ez a kvíz „challenge" változata, melyet otthoni gyakorlásra, zh-ra készülésre szántam. Sajnos a COVID járvány miatt a zh elmaradt, később más formában lett megtartva, így a challenge játék jelentősége érvényét vesztette.

Sajnos a hallgatói kérdőívet csak 5 hallgató töltötte ki, ők átlagosan 4.8-ra értékelték a Kahoot órán történő használatát (live változat), 4.4-re pedig a Kahoot otthoni használatát (challenge változatát). Mindenki úgy nyilatkozott, hogy segítette őt a tanulásban, és egy szöveges véleményt is kapott a feladat, amely így hangzik: *„Szerintem tök jó egy versenyszellemet bevezetni az órára, akár személyesen az egyetemen (mint ahogy tettük is) akár így online is"*

## 3.5. Párosítós feladat a bináris fák gyakoroltatására

A binárisfák témakörének elmélyítésére létrehoztam egy „drag & drop" párosítós feladatot a HotPotatoes-ban. A feladat két oldalból áll, az első oldalon (10. ábra) 5 feladat a bináris fák rekurzív bejárásait gyakoroltatja, a második oldalon (11. ábra) pedig 7 fontos bináris fával kapcsolatos tulajdonság algoritmikus leírását tartalmazza. Mindkét oldal ismeretei elsősorban az olyan órán tanult információk elmélyítését szolgálja, amely ismeretek a zh-n és a vizsgán is számon lettek kérve.


**11. ábra:** Bináris fák, 2. feladat, ellenőrzés

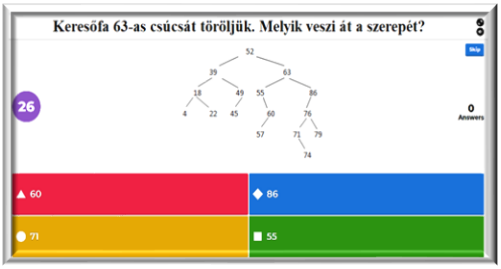



**10. ábra**: Bináris fák, 1. feladat, kezdő kép

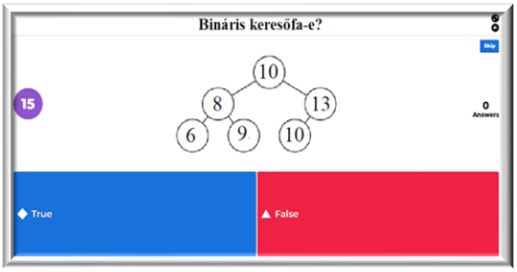
A feladatot 16-an oldották meg a 43-ból (14/20, illetve 2/23), és 6 hallgató véleményezte. Ők mindannyian 5-öst adtak a feladatra, illetve mindenki úgy nyilatkozott, hogy segítette őket a tanulásban. Erre a feladata is érkezett szöveges vélemény, ezek közül az egyik így szól: *„ez nagyon hasznos volt!!! iszonyatosan tetszett”*

## 3.6. Kahoot challenge bináris keresőfák gyakoroltatására

A bináris keresőfák témakörhöz létrehoztam egy Kahoot challenge appot, mely 7 kérdésből állt (4 kvíz és 3 igaz/hamis). Az igaz/hamis kérdések (13. ábra) arról szóltak, hogy egy-egy bináris fáról el kellett dönteni, hogy keresőfa-e, vagy sem. A négy kvízkérdés (12. ábra) a keresőfa bejárásairól, tulajdonságairól, műveleteiről, illetve a szintfolytonos bejárás algoritmusáról szól.



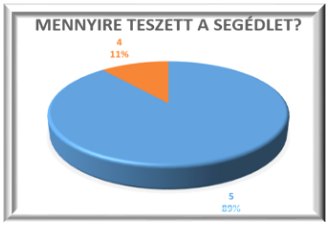

**12. ábra:** Kahoot kvíz feladat




**13. ábra:** Kahoot igaz/hamis feladat

A hallgatók megoldásai jól sikerültek, az egyes feladatok helyes megoldása 74% és 52% között mozogtak. Összesen 31 hallgató töltötte ki a kvízt, a megoldásaik 100% és két hallgató kivételével 43% között mozogtak.

A hallgatói kérdőívet 9-en töltötték ki, ők átlagosan 4,89-re értékelték (16. ábra) és mindenkit segített a tanulásban. A feladat néhány szöveges értékelést is kapott, ezek a következők: *„Tetszett, hogy volt időkorlát.”, „jó ez a Kahoot, csak lenne egy picivel több idő :D”, „nagyon jó a Kahoot versenyszellem miatt :D*”




**14. ábra:** Hallgatói vélemény a Kahootról

# 4. Összegzés

Rohamosan változó világunkban a tanárok sem tudnak megmaradni a hagyományos módszereknél, ha sikeresen szeretnének tanítani. Mivel a ma felnövekvő nemzedék már egy online világba született, az oktatásnak is nyitnia kell az okos eszközök felé. A cikkemben egy egyetemi tárgy innoválásával és annak hatásával foglalkoztam. A kurzust kiegészítettem online videókkal, melyeknek egy részét a hallgatók készítették el. Ezen felül „gamifikáltam” a kurzust, azaz pontrendszert vezettem be, kibővítettem a jutalmazási rendszert, illetve minden órához készítettem egy edutainment alkalmazást, melynek célja a tanult ismeretek elmélyítése, egy otthoni játékos környezetben. Mindezen változtatásokat a hallgatókkal névtelen online kérdőív formájában véleményeztettem.

A hallgatók magas részvétele, illetve válaszuk egyértelműen igazolták, hogy még a felsőoktatásban is helye van az oktatás gamifikálásának, illetve a tananyag vizualizálásának.

Az **EFOP-3.6.3-VEKOP-16-2017-00002**. számú projektben elvégzett szakmai feladat az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósult meg.

## Irodalomjegyzék


1. *wikipedia*
   https://hu.wikipedia.org/wiki/Gamification (utoljára megtekintve: 2020. 08. 19.)
2. Deterding, S., Sicart, M., Nacke, L., O'Hara, K., & Dixon, D: *Gamification. Using Game-Design Elements in Non-Gaming Contexts.* In CHI'11 Extended Abstracts on Human Factors in Computing Systems (2011)(pp. 2425-2428). ACM.
   http://dl.acm.org/citation.cfm?id=1979575 (utoljára megtekintve: 2020. 08. 19.)
3. Lanszki Anita (2016/b): *Digitális történetmesélés és tanulói tartalom(re)konstrukció*. In Új Pedagógiai Szemle. 66. 3/-4. 82−88.
4. Lanszki Anita (2015): *A tanulói aktivitás szerepe a digitális történetmesélésben.* In: Lévai Dóra és PappDanka Adrienn (szerk.): Interaktív oktatásinformatika. Eötvös Kiadó, Budapest. 79−92. 23. Barrett, H. C. (2009): *How to Create Simple Digital Stories.*
   http://electronicportfolios.com/digistory/howto.html (utoljára megtekintve: 2020. 08. 19.)
5. Barrett, H. C. (2009): *How to Create Simple Digital Stories.*
   http://electronicportfolios.com/digistory/howto.html (utoljára megtekintve: 2020. 08. 19.)
6. https://people.inf.elte.hu/kinga/algoritmusok2/seged.htm (utoljára megtekintve: 2020. 08. 19.)
7. https://people.inf.elte.hu/kinga/algoritmusok1/seged.htm (utoljára megtekintve: 2020. 08. 19.)
8. Nádori Gergyely és Prievara Tibor: *IKT módszertan: Kézikönyv az info-kommunikációs eszközök tanórai használatához* (2012)
   http://mek.oszk.hu/15900/15959/15959.pdf (utoljára megtekintve: 2020. 08. 19.)
9. Nádori Gergely: *Gamification* (2012) in PIL Akadémia 7
   http://tanarblog.hu/attachments/3010_7_gamification.pdf (utoljára megtekintve: 2020. 08. 19.)
10. Kovácsné Pusztai Kinga: *Játékosítás (gamification) az* oktatásban (2018) In Infodidact 2018.
11. Kovácsné Pusztai Kinga: *Edutainment is Education* (2019) In XXXII. DidMatTech 2019.
12. Lanszki Anita és Papp-Danka Adrienn (2017):*Digitális történetmesélés alkalmazása természetudományos témájú tantárgyi tartalmak feldolgozásában.* In Neveléstudomány 2017/2 (DOI: 10.21549/NTNY.18.2017.2.3)
13. Abdolmanafi-Rokni, S. J. & Qarajeh, M. (2014): *Digital Storytelling in EFL classrooms: The effect on the oral performance*. In International Journal of Language and Linguistics. 4. 252−257.
14. Ya-Ting, C. Y. & Wan-Chi, I. W. (2012): *Digital storytelling for enhancing student academic achievement, critical thinking, and learning motivation: A year-long experimental study*. In Computers & Education. 2. 339–352
15. Smeda, N., Dakich, E. & Sharda, N. (2014): *The effectiveness of digital storytelling in the classrooms: a comprehensive study*. In Smart Learning Environments. 6.
   http://www.slejournal.com/content/1/1/6 (utoljára megtekintve: 2020. 08. 19.)
16. Froman Richárd, Damsa Andrej: *A gamifikáció (játékosítás) motivációs eszköztára az oktatásban* (2016)
   http://folyoiratok.ofi.hu/uj-pedagogiai-szemle/a-gamifikacio-jatekositas-motivacios-eszkoztara-az-oktatasban (utoljára megtekintve: 2018. 10. 19.)
17. Kenéz András: *A játékosítás (gamification) a felsőoktatásban.* (2016) in Fehér András, Kiss Virág Ágnes, Dr. Soós Mihály, Dr. Szakály Zoltán (szerk.): Hitelesség és Értékorientáció a Marketingben. Debreceni Egyetem Gazdaságtudományi Kar: Debrecen. ISBN: 978 963 472 8 pp. 276–288.
18. Prievara Tibor: *Pontrendszer mint értékelési forma az angolórákon - Virágh Szabolcs írása* (2016) in http://tanarblog.hu/cikk/pontrendszer-mint-ertekelesi-forma-az-angolorakon-viragh-szabolcs-irasa (utoljára megtekintve: 2020. 10. 31.)

19. *Edutainment*. In techopedia
https://www.techopedia.com/definition/5506/edutainment (utoljára megtekintve: 2019. 04. 26)

# A módszeres programozás absztrakciós szintjei, a programozási paradigmák és a programozási nyelvek támogatása

Menyhárt László Gábor
(ORCID: 0000-0002-1574-4454)

menyhart@inf.elte.hu
ELTE Eötvös Loránd Tudományegyetem
Informatikai Kar

**Absztrakt.** Egy hétköznapi probléma számítógépes megoldása során több absztrakciós lépésre is sor kerül. A gyakorlott programozók számára ezek a lépések magától értetődőek, míg a kezdő programozók számára gyakran nehézséget okoznak. Ebben a cikkben összeszedem, hogy a módszeres programozás hogyan segíti az absztrakciót, hogyan jelennek meg a programozási paradigmák, mit segítenek és miben nehezítenek a programozási nyelvekben megjelenő paradigmákat támogató szintaktikai lehetőségek. Egy konkrét feladat megoldásával illusztrálom is a lépéseket és felmerülő problémákat.

**Kulcsszavak:** módszeres programozás, programozási paradigma, programozási nyelvek, oktatás, mérés

## 1. Bevezetés

Egy hétköznapi probléma megoldásához nem feltétlenül van szükség számítógépre. Amikor azonban sokszor kell elvégezni valamit, nagy mennyiségen kell ugyanazt megtenni, egyre gyakrabban halljuk a robotizáció bevezetését. Ilyenkor a gépeket programozni kell és a probléma megoldásához szükséges az adatokat, bemeneti paramétereket digitalizálni. Azaz valamilyen módon mérni kell és számítógéppel feldolgozható formába kell hozni. Ez a lépés majdnem mindig kimarad a programozók képzésének elején, rögtön elvárjuk, hogy feladat szövegének elolvasása után képesek legyenek az adatokat kigyűjteni. Ebben a cikkben egy konkrét példán keresztül igyekszem bemutatni a számítógépes problémamegoldás különböző lépései.

## 2. A probléma felvetése

### A feladat

A feladatunk legyen az, hogy egy születésnapi ünnepségen sokféle üdítőital került az asztalra. Valaki kitalálta, hogy a legcukrosabb italokat szedjük le az asztalról. Gyűjtsük ki a legcukrosabb italokat!

A módszeres programozást [5] követve megsejtjük a felhasználandó algoritmusmintákat, ami a jelen esetben a maximumkiválasztás és a kiválogatás. Ezek absztrakt specifikációját és algoritmusát ismerjük, így könnyebben elkészítjük a konkrét specifikációt és az algoritmusokat, amit lekódolunk, majd felismerjük, hogy hatékonyabban is meg tudjuk oldani a feladatot az algoritmusminták összeépítésével.

### Az absztrakciós lépések

Az első absztrakciós lépés annak a kitalálása, hogy az adatokat hogyan reprezentáljuk, illetve az adatok megszerzése. Ilyenkor már a feladatot is átfogalmazzuk erre a szintre. A *miből?*, *mit?* és *mi lesz igaz?* kérdésekre válaszolunk. Néha előfordul, hogy az adatokat transzformálni is kell, hogy köny-

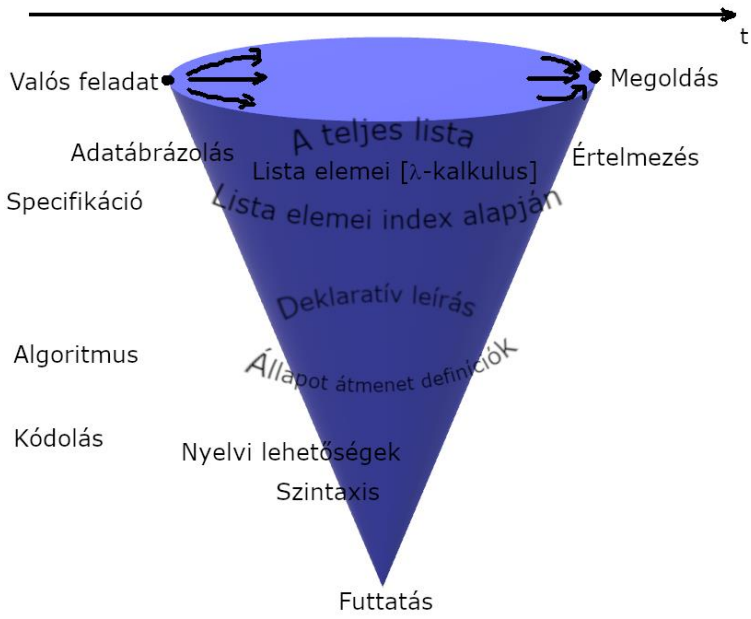
nyebben megoldjuk a feladatot. Ezeket az absztrakciós lépéseket segítik a módszeres programozás specifikációs elvárásai, ahol az adatok, típusok és azok megszorításai, azaz igaz matematikai kifejezések felírásával kompakt módon definiáljuk a problémát.

A második absztrakciós lépés sikeres teljesítésével eljutunk arra a szintre, hogy tudunk válaszolni a *hogyan?* kérdésre. A módszeres programozás sémákkal, programozási tételekkel, azaz algoritmusmintákkal segít. Ugyanakkor elképzelhető itt az imperatív állapotátmenetek leírása helyett egy deklaratív leírás is.

A harmadik absztrakciós lépéssel eljutunk az implementációhoz, amit a korábban megfogalmazott algoritmus vagy deklaratív leírás alapján könnyebben készítünk el. Itt arra a kérdésre válaszolunk, hogy *a gép hogyan?* végezze el a feladatát.

Másik oldalról közelítve fontos, hogy hogyan tervezünk, mi a kódolás módszere, milyen a megoldás stílusa, azaz milyen programozási paradigmát követünk. Én most a két fő, imperatív illetve deklaratív paradigmára koncentrálok. Egyre több programozási nyelvben jelenik meg egymás mellett több paradigma, amiket ezért multiparadigma nyelveknek hívhatunk. Konkrétan a λ-kalkulus támogatása jelenik meg, amivel egy nem teljesen tiszta funkcionális programozás hajtható végre, ahhoz hasonló gondolkodást követ.

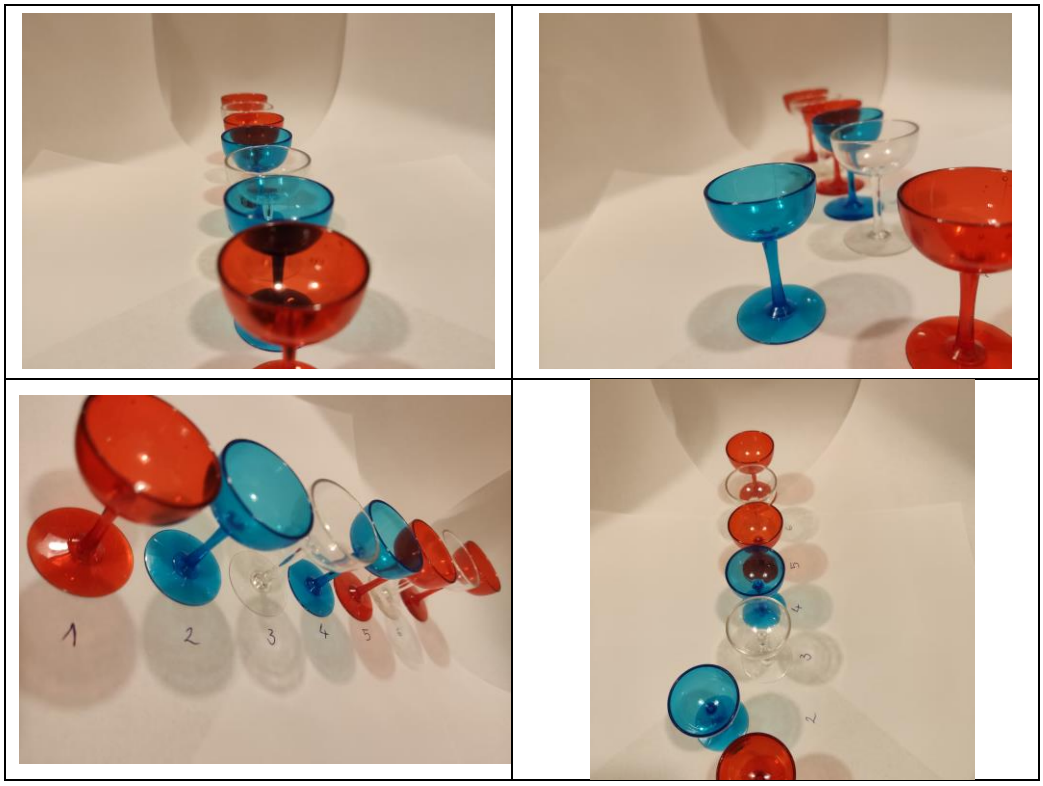
Az érdekel, hogy különböző absztrakciós szinteken hogyan jelenhetnek meg a paradigmák. A szoftvertesztelési V-modellből kiindulva felrajzoltam egy 3D-s alakzatot, mely azt próbálja érzékeltetni, hogy az egyes absztrakciós szinteken hogyan jelennek meg a paradigmák.




**1. ábra:** Hogyan jutunk el a feladattól a megoldásig?

Egy problémát sokféleképpen elkezdhetünk és a végén – reményeink szerint – ugyanazon megoldáshoz, vagy megoldásokhoz jutunk. Már a valós életben is megjelenhetnek a gondolkozási módok, szempontok. Lehet, hogy a poharak sorozata vagy listája, amire gondolunk, de előfordulhat,

hogy az egyes poharakkal szeretnénk foglalkozni, vagy ismerjük a poharak számát és a sorszámukkal hivatkozunk rájuk. De ez a lényegen, végeredményen nem változtat, csak más nézőpontból figyeljük.



**2. ábra:** Hogy tekintünk a poharakra?

## 3. Tervezés

**Specifikáció**

Ebben a konkrét feladatban is többféleképpen lehet az adatokat ábrázolni. Elképzelhető a nevek listája és külön a cukortartalmak listája azzal a kiegészítő információval, hogy azonos elemszámúak a listák és az azonos pozíciókon az összetartozó adatok vannak; vagy a névből és cukortartalomból álló összetartozó rekordok listáját vesszük. Már ettől a választástól is függ, hogy hogyan tudunk továbbhaladni.

Bemenet:

$N \in \mathbb{N}$

$it \in (n\ x\ c)^N$ , italok adatai

$n \in \mathbb{S},\ c \in \mathbb{N}$, név illetve cukortartalom

Kimenet:

$db \in \mathbb{N}$

$lc \in \mathbb{S}^{db}$ , legcukrosabbak

Előfeltétel:

nincs

Utófeltétel:

$$\exists j(1 \leq j \leq N): maxC = it_j.c \wedge \forall i(1 \leq i \leq N): it_i.c \leq maxC$$

$$db = \sum_{\substack{i=1 \\ it_i.c = maxC}}^{N} 1$$

$$\forall i\ (1 \leq i \leq db)\ \exists j(1 \leq j \leq N): lc_i = it_j.n \wedge it_j.c = maxC$$

$$halmazfelsorolás\ (lc)$$

Játszunk el kicsit a jelölésekkel és nézzük meg, hogy a paradigmák hogyan jelenhetnek meg a specifikáció szintjén.

Írjuk át az első sort kompaktabb módra:

$$maxC = \underset{i=1}{\overset{N}{MAX}}\ it_i.c$$

Vagy használhatunk halmazt is:

$$maxC = \underset{ital \in \boldsymbol{it}}{MAX}\ ital.c$$

Halmazjelölésekkel is felírhatjuk az első hosszabb kifejezést:

$$\exists ital \in \boldsymbol{it}: maxC = ital.c \wedge \forall masital \in \boldsymbol{it}: masital.c \leq maxC$$

Vagy itt a legkompaktabb lehetőség:

$$maxC = MAX\ \boldsymbol{it}.c$$

Az előzőek mintájára a szumma jelnél használjunk halmazt:

$$db = \sum_{\substack{ital \in \boldsymbol{it} \\ ital.c = maxC}} 1$$

Kihagyhatunk indexet az egyikből:

$$\forall i\ (1 \leq i \leq db)\ \exists ital \in \boldsymbol{it}: lc_i = ital.n \wedge ital.c = maxC$$

Vagy akár mindből:

$$\forall egyiklc \in lc: \exists ital \in \boldsymbol{it}: egyiklc = ital.n \wedge ital.c = maxC$$

Így itt is kaphatunk kompakt formát:

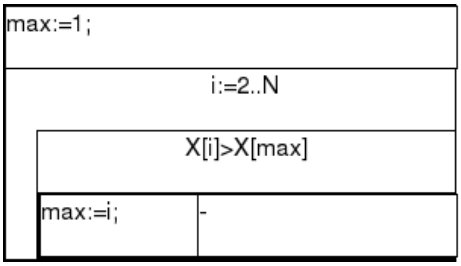
$$\forall egyiklc \in lc: \exists ital \in \boldsymbol{it}: egyiklc = ital.n \wedge ital.c = MAX\ \boldsymbol{it}.c$$

**Algoritmus**

Használjuk az algoritmusmintákhoz tartozó absztrakt algoritmusokat:

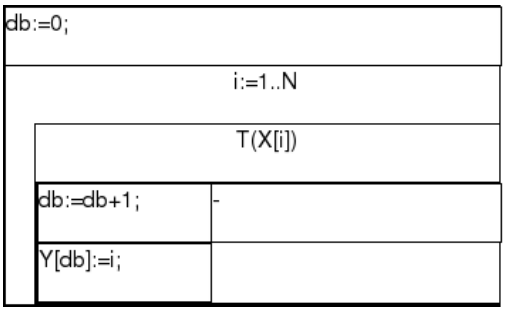
Maximumkiválasztás

```
max:=1
Ciklus i:=2-től N-ig
  Ha X[i]>X[max] akkor
    max:=i
  elágazás vége
Ciklus vége
```




Kiválogatás

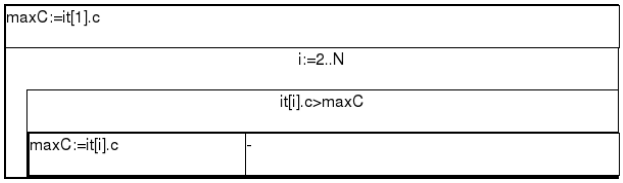
```
db:=0
Ciklus i:=1-től N-ig
  Ha T(X[i]) akkor
    db:=db+1;
    Y[db]:=i;
  elágazás vége
Ciklus vége
```




Írjuk át a konkrét feladathoz tartozó változók és feltételek használatával:

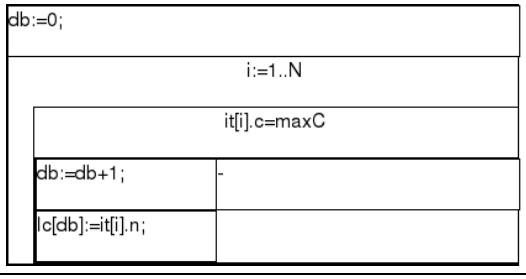
Maximumkiválasztás

```
maxC:=it[1].c
Ciklus i:=2-től N-ig
  Ha it[i].c>maxC akkor
    maxC:=it[i].c
  elágazás vége
Ciklus vége
```




Kiválogatás

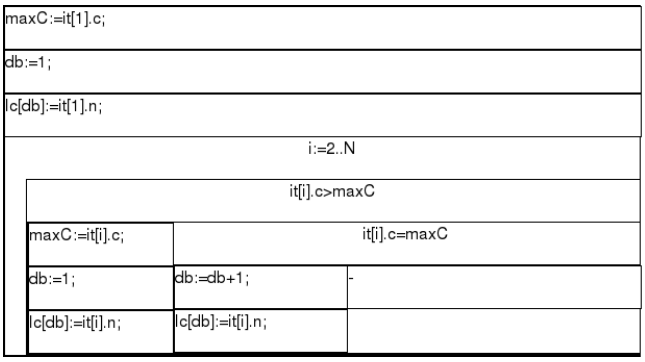
```
db:=0
Ciklus i:=1-től N-ig
  Ha it[i].c=maxC akkor
    db:=db+1;
    lc[db]:=it[i].n;
  elágazás vége
Ciklus vége
```



A tételek összeépítésével hatékonyabb kódot kapunk, így nem kell kétszer megvizsgálni a lista összes elemét:

Maximumkiválasztás és kiválogatás összeépítve

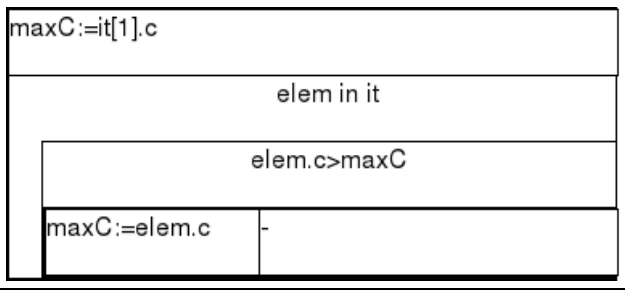
```
maxC:=it[1].c
db:=1;
lc[db]:=it[1].n;
Ciklus i:=2-től N-ig
  Ha it[i].c>maxC akkor
    maxC:=it[i].c
    db:=1;
    lc[db]:=it[i].n;
  különben Ha
it[i].c=maxC akkor
    db:=db+1;
    lc[db]:=it[i].n;
  elágazás vége
Ciklus vége
```




Ugyanezeket átírhatjuk olyan formába, hogy index helyett az egyes elemeket használjuk a ciklus változójaként.

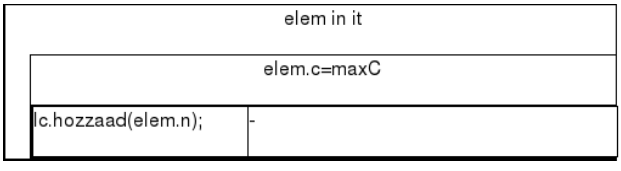
Maximumkiválasztás

```
maxC:=it[1].c
Ciklus elem az it-ben
  Ha elem.c>maxC akkor
    maxC:=elem.c
  elágazás vége
Ciklus vége
```




Kiválogatás

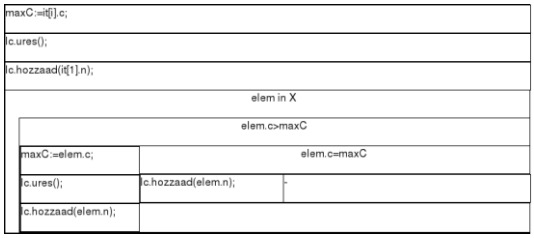
```
lc.üres()
Ciklus elem az it-ben
  Ha elem.c=maxC akkor
    lc.hozzáAd(elem.n);
  elágazás vége
Ciklus vége
```



A tételek összeépítésével itt is hatékonyabb kódot kapunk, amiben nem kell kétszer végighaladunk a lista elemein:

Maximumkiválasztás és kiválogatás összeépítve

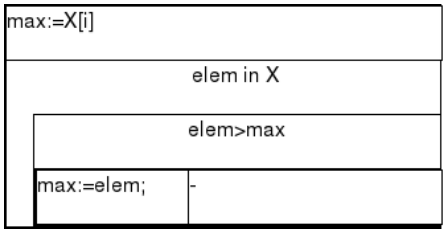
```
maxC:=it[1].c;
lc.üres();
lc.hozzáad(it[1].n);
Ciklus elem az it-ben
  Ha elem.c>maxC akkor
    maxC:=elem.c
    lc.üres();
    lc.hozzáad(elem.n);
  különben Ha elem.c=maxC akkor
    lc.hozzáad(elem.n);
  elágazás vége
Ciklus vége
```




Természetesen az eredeti absztrakt algoritmusokat is átírhatjuk hasonlóképpen:

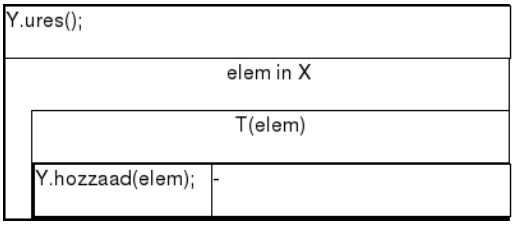
Maximumkiválasztás

```
max:=X[1];
Ciklus elem az X-ben
  Ha elem>max akkor
    max:=elem
  elágazás vége
Ciklus vége
```



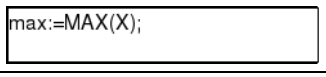

Kiválogatás

```
Y.üres();
Ciklus elem az X-ben
  Ha T(elem) akkor
    Y.hozzáad(elem);
  elágazás vége
Ciklus vége
```




Ez pedig a legkompaktabb algoritmus, amit néhány nyelv támogat is:

```
max:=MAX(X)
```




Ne menjünk el szó nélkül amellett a lehetőség mellett, hogy próbáljuk leírni a sorozatot „egyben". Azaz „látjuk" a sorozat egy-két elemét és „tudjuk", hogy van még a többi elem. Ekkor leírhatjuk, hogy hogyan viselkedik egy rövid sorozat (pl. egy elem) és azt is, hogy hogyan tudjuk visszavezetni rövidebb sorozatra.

```
maximum [e] = e
maximum (e:m:többi) = maximum ( (ha e>m akkor e különben m) : többi )
```

## 4. Implementáció

Az előző, az ismétlésről szóló cikkem [6] alapján a JavaScriptet [2] választottam az egyik implementációs nyelvnek, mert az támogatja az összes paradigma szerinti kódolási lehetőséget és még kiegészítő függvénykönyvtárak is léteznek hozzá, mint például az Underscore vagy a Lodash. Ezen kívül még Haskell-ben [3, 4], egy tisztán funkcionális nyelven is megírtam a feladathoz tartozó kódokat.

A következő öt kód mindegyike a maximumkiválasztás és kiválogatás egymás utáni alkalmazását implementálja:

```
maxC=it[0].c;
for (let i=1; i<it.length; i++)
{
  if (it[i].c>maxC) {
    maxC=it[i].c;
  }
}
names=[];
for (let i=0; i<it.length; i++)
{
  if (it[i].c==maxC) {
    names.push(it[i].n);
  }
}
```

```
maxC=it[0].c;
it.forEach(elem => {
  if (elem.c>maxC) {
    maxC=elem.c;
  }
});
names=[];
it.forEach(elem => {
  if (elem.c==maxC) {
    names.push(elem.n);
  }
});
```

```
maxC=_.max(_.map(it,function(elem){return elem.c;}));
names=_.filter(it,function(elem){return elem.c==maxC;});
```

```
maxC=_.max(_.map(it,function(elem){return elem.c;}));
names=_.where(it,{c:maxC});
```

```
maxC=lo.max(lo.map(it,function(elem){return elem.c;}));
names=lo.filter(it,function(elem){return elem.c==maxC;});
```

A kódot átírhatjuk úgy, hogy a tételeket összeépítjük így a kód komplexitása nő, megértése nehezebbé válhat, amikor elhagyjuk az algoritmusminta követését, viszont a futása gyorsabb lesz, vagyis nő a hatékonyság. A következő 5 kód ezt mutatja be:

```
maxC=it[0].c;
names=[];
names.push(it[0].n);
for (let i=1; i<it.length; i++)
{
  if (it[i].c>maxC) {
    maxC=it[i].c;
    names=[];
    names.push(it[i].n);
  } else if (it[i].c==maxC) {
    names.push(it[i].n);
  }
}
```

```
maxC=it[0].c;
names=[];
//names.push(it[0].n);
it.forEach(elem => {
  if (elem.c>maxC) {
    maxC=elem.c;
    names=[];
    names.push(elem.n);
  } else if (elem.c==maxC) {
    names.push(elem.n);
  }
});
```

```
maxO=it.reduce((tmpMax,elem) => {
  if (elem.c>tmpMax.maxC) return {maxC:elem.c,names:[elem.n]}
  else if (elem.c==tmpMax.maxC) { newnames=tmpMax.names; new-
names.push(elem.n); return {maxC:tmpMax.maxC,names:newnames};}
  else return tmpMax;
  },
  {maxC:it[0].c,names:[]}
);
```

```
maxO=_.reduce(it,function(tmpMax,elem){
  if (elem.c>tmpMax.maxC) return {maxC:elem.c,names:[elem.n]}
  else if (elem.c==tmpMax.maxC) { newnames=tmpMax.names; new-
names.push(elem.n); return {maxC:tmpMax.maxC,names:newnames};}
  else return tmpMax;
  },{maxC:it[0].c,names:[]}
);
```

```
maxO=lo.reduce(it,function(tmpMax,elem){
  if (elem.c>tmpMax.maxC) return {maxC:elem.c,names:[elem.n]}
  else if (elem.c==tmpMax.maxC) { newnames=tmpMax.names; new-
names.push(elem.n); return {maxC:tmpMax.maxC,names:newnames};}
  else return tmpMax;
  },{maxC:it[0].c,names:[]}
);
```

Haskell-ben csak a két lépéses megoldásra mutatok több példát. Az objektumok listájából kétféleképpen nyerem ki az adatokat:

```
values :: [Integer]
values = [v | (n,v) <- Obj.v ]
```

```
values' = map fieldValue Obj.v
  where
  fieldValue (n,v) = v
```

A maximumkiválasztásra 3 saját implementációt is bemutatok a beépített maximum függvény mellé:

```
maximum' :: Ord a => [a] -> a
maximum' = foldr1 (\x y ->if x >= y then x else y)
```

```
maximum'' :: Ord a => [a] -> a
maximum'' [x]       = x
maximum'' (x:x':xs) = maximum'' ((if x >= x' then x else x'):xs)
```

```
maximum''' :: Ord a => [a] -> a
maximum''' [x]       = x
maximum''' (x:x':xs) | x >= x'   = maximum''' (x:xs)
maximum''' (x:x':xs) | otherwise = maximum''' (x':xs)
```

A nevek kiválogatására is mutatok egy lehetőséget:

```
maxNames m' a = map fieldName (filter (flip equalValue m') a)
  where
  fieldName (n,v) = n
  equalValue (n,v) m = (v==m)
```

Az adatok tárolását bemutató példa és a forráskódok teljes tartalma az A mellékletben található.

## 5. Mérés

Mérést végeztem a fent bemutatott kódokkal. A futtatáshoz egy Intel® Core™ i7-8750H 2.20GHz-es processzorú, 32GB memóriával rendelkező számítógépet használtam, melyen 64 bites Windows 10 Pro operációs rendszer futott. A JavaScript kódokat a Node v10.16.0 segítségével futtattam, míg a Haskell forráskódok fordításához a The Glorious Glasgow Haskell Compilation System, version 8.10.1 fordítóprogramot használtam.

A következő táblázatokban szereplő konkrét értékek egymáshoz való viszonyát érdemes figyelni.

Először csak egész számokat tartalmazó listából történő maximum érték meghatározásának idejét hasonlítottam össze:

| Implementáció | Idő [ms] (100.000) | Idő [ms] (1.000.000) |
|---|---|---|
| For | 5 | 6 |
| ForEach | 2 | 10 |
| Reduce | 2 | 10 |
| Math.max(… | 1 | HIBA |
| Math.max.apply( | 1 | HIBA |
| _.max | 1 | 3 |
| _.reduce | 2 | 4 |
| lodash.max | 2 | 6 |
| lodash.reduce | 2 | 3 |

A JavaScript-ben lévő beépített Math.max függvény 100.000 elemszámig nagyon gyors, de a fölött már nem működik. A továbbiakban 1.000.000 elemszámmal dolgoztam, hogy jobban látszódjanak az eltérések.

A maximumérték meghatározása az objektumok listájából:

| Implementáció | Idő [ms] (1.000.000) |
|---|---|
| For | 7 |
| ForEach | 15 |
| Reduce | 11 |
| _.max | 14 (20*) |
| _.reduce | 7 |
| lodash.max | 15 (20*) |
| lodash.reduce | 5 |

A csillaggal megjelölt implementációkban szükség volt egy map használatára is, aminek a tiszta ideje nagyobb, mint az összeépített futás. Ebből az látszik, hogy a compiler javít a futás teljesítményén és nem mappel át mindent.

Az objektumok listájából a maximumérték meghatározása Haskell kódok segítségével a következő futásidőket adta:

| Implementáció | Idő [ms] (1.000.000) |
|---|---|
| Maximum values | 320 |
| Maximum' values | 310 |
| Maximum" values | 135 |
| Maximum"' values | 20 |
| Maximum values' | 390 |
| Maximum' values' | 120 |
| Maximum" values' | 300 |
| Maximum"' values' | 20 |

Ez a táblázat bizonyítja, hogy a tisztán funkcionális nyelveknél is függ a futásidő a konkrét deklaratív leírástól. Ez a leírás nagyobb kifejező erővel rendelkezik, viszont nagyobb absztrakciót igényel és kevésbé áll közel a gépi kódhoz.

A következőkben az objektumok listájából már a maximumértékkel rendelkező összes név meghatározása jön JavaScripttel:

| Implementáció | Idő [ms] (1.000.000) |
|---|---|
| For 2x | 25 |
| ForEach 2x | 22 |
| _.max + _.filter | 22 |
| _.max + _.where | 95 |
| lodash.max + lodash.filter | 27 |

Látszik, hogy az Underscore `where` függvényénél a kód jól olvasható, de a futásideje háromszorosa a többihez képest.

Egy lépésben hatékonyabb a megoldás:

| Implementáció | Idő [ms] (1.000.000) |
|---|---|
| For | 12 |
| ForEach | 15 |
| Reduce | 14 |
| _.reduce | 10 |
| lodash.reduce | 9 |

Végül legyen itt a Haskell implementáció ideje is, ami most csak a második lépést, a kiválogatást tartalmazza. Ebben viszont benne van a kiírás ideje is, ugyanis a Lazy evaluation miatt nem kerülne feldolgozásra az adat.

| Implementáció | Idő [ms] (1.000.000) |
|---|---|
| maxNames | 20 |

## 6. Didaktikai megfontolások

Az előzőek alapján azt mondhatjuk, hogy módszeres programozás által nyújtott kérdések, sémák, algoritmusminták segítik a kezdő programozók oktatását [1], de szükséges, hogy ne csak az absztrakciós szinteket, hanem több gondolkodási formát, paradigmát megtanítsunk, hiszen a programozási nyelvekben is lehetőség van különböző megvalósítási módokra. Fontos, hogy a rendszert és az összefüggéseket is megértsék a hallgatók. Ne csak egymás hegyén-hátán legyenek az építőkövek, mert fontos a hatékonyság is. Nem csak a kód áttekinthetőségére, érthetőségére, hanem a futási idejére is tekintettel legyünk.

Úgy gondolom, hogy a sémakövetés nem kell, hogy a kreativitás rovására menjen. Meg kell tanítanunk a diákoknak, hogy ne csak kövessék, hanem értsék is, hogy mi történik. Használhatják az elinduláshoz, de tudniuk kell, hogy miért úgy van és keresniük kell a továbblépési lehetőségeket a legjobb megoldás elkészítéséhez. Később szükségük lehet arra is, hogy ne csak egy adott nyelv támogatását tanulják meg, hanem *mint eszköz* tekintsenek ezekre a lehetőségekre.

Azért, hogy a diákjaim megtanulják a mögöttes tartalmat és ne legyenek lusták gondolkodni, a kezdő programozók oktatásának korai stádiumában még a specifikációban is kérem a hosszabb leírásokat a `MAX`, `halmazfelsorolás` és hasonló rövidítések helyett. Így később nem csak egymás után dobálják a függvényeket, hanem átgondolják a megoldást.

## Irodalom

1. Tananyag kezdő programozók számára
http://progalap.elte.hu/downloads/seged/eTananyag/ (utoljára megtekintve: 2020.11.18.)
2. JavaScript referencia
https://developer.mozilla.org/en-US/docs/Web/JavaScript (utoljára megtekintve: 2020.11.18.)
3. Haskell hivatalos oldala
https://www.haskell.org/ (utoljára megtekintve: 2020.11.18.)
4. Haskell függvényosztályok
https://hackage.haskell.org/package/base-4.14.0.0 (utoljára megtekintve: 2020.11.18.)
5. Zsakó László, Szlávi Péter: Módszeres programozás, Műszaki Könyvkiadó, Budapest (1986)
6. Menyhárt László Gábor: Overview of Repetition, Central-European Journal of New Technologies in Research Education and Practice (2676-9425): 2020. volume 2, number 1, pp 34-75 (2020).
DOI: https://doi.org/10.36427/CEJNTREP.2.1

# Mellékletek

## A. JavaScript implementációk

**Tesztadatok**

data_9.js formátuma:

```
v = [232839,99370,16617,661198,606350,276061,658523,866515,742354];
exports.v=v;
```

obj_14.js formátuma:

```
v = [{n:"n273797",v:273797},
{n:"n216959",v:216959},
{n:"n524988",v:524988},
{n:"n834189",v:834189},
{n:"n569940",v:569940},
{n:"n187511",v:187511},
{n:"n205697",v:205697},
{n:"n14180",v:14180},
{n:"n857387",v:857387},
{n:"n549725",v:549725},
{n:"n620312",v:620312},
{n:"n23331",v:23331},
{n:"n571258",v:571258},
{n:"n855019",v:855019}];
exports.v=v;
```

**Maximum érték meghatározása egészek listájából**

```
const _ = require('underscore');
const lo = require('lodash');
const data = require('./data_9.js');
let it = data.it;

console.log("Max");

var timer_start;
var maxC;
var timer_end;

console.log("For");
timer_start = Date.now();

maxC=it[0];
for (let i=1; i<it.length; i++)
{
    if (it[i]>maxC) {
            maxC=it[i];
          }
}

timer_end = Date.now();
console.log(maxC);
elapsed=timer_end-timer_start;
console.log("For;time:",elapsed);
```

```
console.log("foreach");
timer_start = Date.now();

maxC=it[0];
it.forEach(elem => {
          if (elem>maxC) {
            maxC=elem;
          }
});

timer_end = Date.now();
console.log(maxC);
elapsed=timer_end-timer_start;
console.log("foreach;time:",elapsed);

console.log("reduce");
timer_start = Date.now();

maxC=it.reduce((tmpMax,elem) => {
            if (elem>tmpMax) return elem
            else return tmpMax;
          },
          it[0]
);

timer_end = Date.now();
console.log(maxC);
elapsed=timer_end-timer_start;
console.log("reduce;time:",elapsed);

/* This part works with max data_100000 / 100.000 element * /
console.log("Math.max(...");
timer_start = Date.now();

maxC=Math.max(...it);

timer_end = Date.now();
console.log(maxC);
elapsed=timer_end-timer_start;
console.log("Math.max(...;time:",elapsed);

console.log("Math.max.apply(");
timer_start = Date.now();

maxC=Math.max.apply(null,it);

timer_end = Date.now();
console.log(maxC);
elapsed=timer_end-timer_start;
console.log("Math.max.apply;time:",elapsed);
/* */

console.log("_.max");
timer_start = Date.now();
```

```
maxC=_.max(it);

timer_end = Date.now();
console.log(maxC);
elapsed=timer_end-timer_start;
console.log("_.max;time:",elapsed);

console.log("_.reduce");
timer_start = Date.now();

maxC=_.reduce(it,function(max,elem){
              if (elem>max) return elem
              else return max;
            },it[0]
           );

timer_end = Date.now();
console.log(maxC);
elapsed=timer_end-timer_start;
console.log("_.reduce;time:",elapsed);

console.log("lo.max");
timer_start = Date.now();

maxC=lo.max(it);

timer_end = Date.now();
console.log(maxC);
elapsed=timer_end-timer_start;
console.log("lo.max;time:",elapsed);

console.log("lo.reduce");
timer_start = Date.now();

maxC=lo.reduce(it,function(max,elem){
              if (elem>max) return elem
              else return max;
            },it[0]
           );

timer_end = Date.now();
console.log(maxC);
elapsed=timer_end-timer_start;
console.log("lo.reduce;time:",elapsed);
```

**Maximum érték meghatározása objektumok listájából**

```
const _ = require('underscore');
const lo = require('lodash');
const data = require('./obj_14.js');
let it = data.it;

console.log("Max");
```

```
var timer_start;
var maxC;
var timer_end;

console.log("For");
timer_start = Date.now();

maxC=it[0].c;
for (let i=1; i<it.length; i++)
{
    if (it[i].c>maxC) {
            maxC=it[i].c;
           }
}

timer_end = Date.now();
console.log(maxC);
elapsed=timer_end-timer_start;
console.log("For;time:",elapsed);

console.log("foreach");
timer_start = Date.now();

maxC=it[0].c;
it.forEach(elem => {
           if (elem.c>maxC) {
             maxC=elem.c;
           }
});

timer_end = Date.now();
console.log(maxC);
elapsed=timer_end-timer_start;
console.log("foreach;time:",elapsed);

console.log("reduce");
timer_start = Date.now();

maxC=it.reduce((max,elem) => {
            if (elem.c>max) return elem.c
            else return max;
           },
           it[0].c
);

timer_end = Date.now();
console.log(maxC);
elapsed=timer_end-timer_start;
console.log("reduce;time:",elapsed);

console.log("_.max");
timer_start = Date.now();

maxC=_.max(_.map(it,function(elem){return elem.c;}));
```

```
timer_end = Date.now();
console.log(maxC);
elapsed=timer_end-timer_start;
console.log("_.max;time:",elapsed);

console.log("_.map");
timer_start = Date.now();

_.map(it,function(elem){return elem.c;});

timer_end = Date.now();
elapsed=timer_end-timer_start;
console.log("_.map;time:",elapsed);

console.log("_.reduce");
timer_start = Date.now();

maxC=_.reduce(it,function(tmpMax,elem){
              if (elem.c>tmpMax) return elem.c
              else return tmpMax;
            },it[0].c
          );

timer_end = Date.now();
console.log(maxC);
elapsed=timer_end-timer_start;
console.log("_.reduce;time:",elapsed);

console.log("lo.max");

timer_start = Date.now();
maxC=lo.max(lo.map(it,function(elem){return elem.c;}));

timer_end = Date.now();
console.log(maxC);
elapsed=timer_end-timer_start;
console.log("lo.max;time:",elapsed);

console.log("lo.map");
timer_start = Date.now();

lo.map(it,function(elem){return elem.c;});

timer_end = Date.now();
elapsed=timer_end-timer_start;
console.log("lo.map;time:",elapsed);

console.log("lo.reduce");
timer_start = Date.now();

maxC=lo.reduce(it,function(tmpMax,elem){
              if (elem.c>tmpMax) return elem.c
              else return tmpMax;
            },it[0].c
          );
```

```
timer_end = Date.now();
console.log(maxC);
elapsed=timer_end-timer_start;
console.log("lo.reduce;time:",elapsed);
```

**Maximum érték és nevek meghatározása**

```
  const _ = require('underscore');
  const lo = require('lodash');
  const data = require('./obj_v2_1000000.js');
  let it = data.it;

  console.log("Max names");

  var timer_start;
  var maxC;
  var names=[];
  var timer_end;

  console.log("For x2");
  timer_start = Date.now();

  maxC=it[0].c;
  for (let i=1; i<it.length; i++)
  {
      if (it[i].c>maxC) {
              maxC=it[i].c;
            }
  }
  names=[];
  for (let i=0; i<it.length; i++)
  {
            if (it[i].c==maxC) {
              names.push(it[i].n);
            }
  }

  timer_end = Date.now();
  console.log(maxC);
  console.log(names.length);
  elapsed=timer_end-timer_start;
  console.log("For x2;time:",elapsed);

  console.log("ForEach x2");
  timer_start = Date.now();

  maxC=it[0].c;
  it.forEach(elem => {
            if (elem.c>maxC) {
              maxC=elem.c;
            }
  });
  names=[];
```

```
it.forEach(elem => {
           if (elem.c==maxC) {
             names.push(elem.n);
           }
});

timer_end = Date.now();
console.log(maxC);
console.log(names.length);
elapsed=timer_end-timer_start;
console.log("ForEach x2;time:",elapsed);

console.log("_.max+_.filter");
timer_start = Date.now();

maxC=_.max(_.map(it,function(elem){return elem.c;}));
names=_.filter(it,function(elem){return elem.c==maxC;});

timer_end = Date.now();
console.log(maxC);
console.log(names.length);
elapsed=timer_end-timer_start;
console.log("_.max+_.filter;time:",elapsed);

console.log("_.map");
timer_start = Date.now();

_.map(it,function(elem){return elem.c;});

timer_end = Date.now();
elapsed=timer_end-timer_start;
console.log("_.map;time:",elapsed);

console.log("_.max+_.where");
timer_start = Date.now();

maxC=_.max(_.map(it,function(elem){return elem.c;}));
names=_.where(it,{c:maxC});

timer_end = Date.now();
console.log(maxC);
console.log(names.length);
elapsed=timer_end-timer_start;
console.log("_.max+_.where;time:",elapsed);

console.log("_.map");
timer_start = Date.now();

_.map(it,function(elem){return elem.c;});

timer_end = Date.now();
elapsed=timer_end-timer_start;
console.log("_.map;time:",elapsed);

console.log("lo.max+lo.filter");
```

```
timer_start = Date.now();

maxC=lo.max(lo.map(it,function(elem){return elem.c;}));
names=lo.filter(it,function(elem){return elem.c==maxC;});

timer_end = Date.now();
console.log(maxC);
console.log(names.length);
elapsed=timer_end-timer_start;
console.log("lo.max+lo.filter;time:",elapsed);

console.log("lo.map");
timer_start = Date.now();

lo.map(it,function(elem){return elem.c;});

timer_end = Date.now();
elapsed=timer_end-timer_start;
console.log("lo.map;time:",elapsed);

// More effective - build together
console.log("For");
timer_start = Date.now();

maxC=it[0].c;
names=[];
names.push(it[0].n);
for (let i=1; i<it.length; i++)
{
    if (it[i].c>maxC) {
            maxC=it[i].c;
            names=[];
            names.push(it[i].n);
          } else if (it[i].c==maxC) {
            names.push(it[i].n);
          }
}

timer_end = Date.now();
console.log(maxC);
console.log(names.length);
elapsed=timer_end-timer_start;
console.log("For;time:",elapsed);

console.log("foreach");
timer_start = Date.now();

maxC=it[0].c;
names=[];
//names.push(it[0].n);
it.forEach(elem => {
          if (elem.c>maxC) {
            maxC=elem.c;
            names=[];
            names.push(elem.n);
```

```
            } else if (elem.c==maxC) {
              names.push(elem.n);
            }
});

timer_end = Date.now();
console.log(maxC);
console.log(names.length);
elapsed=timer_end-timer_start;
console.log("foreach;time:",elapsed);

console.log("reduce");
timer_start = Date.now();

maxO=it.reduce((tmpMax,elem) => {
             if (elem.c>tmpMax.maxC) return
{maxC:elem.c,names:[elem.n]}
             else if (elem.c==tmpMax.maxC) { newnames=tmpMax.names;
newnames.push(elem.n); return {maxC:tmpMax.maxC,names:newnames};}
             else return tmpMax;
            },
            {maxC:it[0].c,names:[]}
);

timer_end = Date.now();
console.log(maxO.maxC);
console.log(maxO.names.length);
elapsed=timer_end-timer_start;
console.log("reduce;time:",elapsed);

console.log("_.reduce");
timer_start = Date.now();

maxO=_.reduce(it,function(tmpMax,elem){
              if (elem.c>tmpMax.maxC) return
{maxC:elem.c,names:[elem.n]}
              else if (elem.c==tmpMax.maxC) { newnames=tmpMax.names;
newnames.push(elem.n); return {maxC:tmpMax.maxC,names:newnames};}
              else return tmpMax;
             },{maxC:it[0].c,names:[]}
            );

timer_end = Date.now();
console.log(maxO.maxC);
console.log(maxO.names.length);
elapsed=timer_end-timer_start;
console.log("_.reduce;time:",elapsed);

console.log("lo.reduce");
timer_start = Date.now();

maxO=lo.reduce(it,function(tmpMax,elem){
              if (elem.c>tmpMax.maxC) return
{maxC:elem.c,names:[elem.n]}
```

```
                else if (elem.c==tmpMax.maxC) { newnames=tmpMax.names;
newnames.push(elem.n); return {maxC:tmpMax.maxC,names:newnames};}
                else return tmpMax;
              },{maxC:it[0].c,names:[]}
            );

timer_end = Date.now();
console.log(maxO.maxC);
console.log(maxO.names.length);
elapsed=timer_end-timer_start;
console.log("lo.reduce;time:",elapsed);
```

## B. Haskell implementációk

**Tesztadatok**

Objv9.hs formátuma:

```
module Objv9 where
  v = [
     ("n354013",354013),
     ("n692809",692809),
     ("n238100",238100),
     ("n939538",939538),
     ("n242031",242031),
     ("n908242",908242),
     ("n283977",283977),
     ("n831996",831996),
     ("n650007",650007)];
```

**Maximum érték és nevek meghatározása**

```
import Control.Exception
import Data.Time
import Objv9 as Obj

maximum' :: Ord a => [a] -> a
maximum' = foldr1 (\x y ->if x >= y then x else y)

maximum'' :: Ord a => [a] -> a
maximum'' [x]       = x
maximum'' (x:x':xs) = maximum'' ((if x >= x' then x else x'):xs)

maximum''' :: Ord a => [a] -> a
maximum''' [x]       = x
maximum''' (x:x':xs) | x >= x'   = maximum''' (x:xs)
maximum''' (x:x':xs) | otherwise = maximum''' (x':xs)

values :: [Integer]
values = [v | (n,v) <- Obj.v ]

values' = map fieldValue Obj.v
    where
    fieldValue (n,v) = v

names = map fieldName Obj.v
```

```
    where
    fieldName (n,v) = n

--- maxNames :: Integer -> [String]
maxNames m' a = map fieldName (filter (flip equalValue m') a)
    where
    fieldName (n,v) = n
    --- equalValue :: (Integer -> a) -> Bool
    equalValue (n,v) m = (v==m)


main = do
    print Obj.v

    start <- getCurrentTime
    let m=maximum values
    print (m)
    end <- getCurrentTime
    print (diffUTCTime end start)

    start <- getCurrentTime
    let m=maximum' values
    print (m)
    end <- getCurrentTime
    print (diffUTCTime end start)

    start <- getCurrentTime
    let m=maximum'' values
    print (m)
    end <- getCurrentTime
    print (diffUTCTime end start)

    start <- getCurrentTime
    let m=maximum''' values
    print (m)
    end <- getCurrentTime
    print (diffUTCTime end start)


    start <- getCurrentTime
    let m=maximum values'
    print (m)
    end <- getCurrentTime
    print (diffUTCTime end start)

    start <- getCurrentTime
    let m=maximum' values'
    print (m)
    end <- getCurrentTime
    print (diffUTCTime end start)

    start <- getCurrentTime
    let m=maximum'' values'
    print (m)
    end <- getCurrentTime
```

```
print (diffUTCTime end start)

start <- getCurrentTime
let m=maximum''' values'
print (m)
end <- getCurrentTime
print (diffUTCTime end start)

--- print names
start <- getCurrentTime
print (maxNames m Obj.v)
end <- getCurrentTime
print (diffUTCTime end start)
```

# Az első ProgTábor: egy új tehetséggondozó program kezdete az algoritmikus programozásban

Nikházy László

nikhazy@inf.elte.hu
ELTE IK

**Absztrakt.** Hazánkban hiánypótló módon kidolgoztunk és kísérleti módon elindítottunk egy tehetséggondozó programot, a ProgTábort, amelynek célja a kiemelten tehetséges és informatika iránt érdeklődő felső tagozatos és középiskolás diákokkal az algoritmikus programozás szépségeinek megismertetése, színvonalas képzés biztosítása számokra ebben a témában, a nagy sikernek örvendő, Pósa Lajos neve által fémjelzett matematika táborok mintájára [1]. Az algoritmikus programozásban a hangsúlyt a problémamegoldási stratégiák, adatszerkezetek, nevezetes algoritmusok és algoritmusminták, és bizonyos elméleti ismeretek alkalmazására helyezzük, a programozási nyelvre (elengedhetetlen) eszközként tekintünk. A táborokban feladatokon keresztül, felfedeztető módszerrel tanítunk, a módszertanról, elméleti háttérről, tananyagról születtek már publikációink [2,3]. Ebben a cikkben röviden összefoglaljuk a gyakorlati megvalósítást, és bemutatjuk az első kísérleti alkalmat, amelyet online rendeztünk meg. Részletesen elemezzük a diákok visszajelzéseiből kapott adatokat, és megosztjuk tapasztalatainkat.

**Kulcsszavak:** programozás tehetséggondozás, algoritmusok és adatszerkezetek, tehetséggondozó táborok, felfedeztető tanítás.

## 1. Bevezetés

A szerző kutatási munkájának célja egy programozás tehetséggondozó rendszer kidolgozása, A Gondolkodás Öröme Alapítvány által szervezett, nemzetközi szinten is egyedülálló, felfedeztető módszertant alkalmazó matematika táborokat tekintve példaként. A kutatás abba a fázisba ért, amikor kísérleti jellegű megvalósítás mellett elemezzük és javítjuk tovább a rendszert, tulajdonképpen akciókutatás módszerét alkalmazva, a megtartott táborokra reflektálva. A kezdeményezésnek a ProgTábor nevet adtuk. Jelen cikkben röviden összefoglalva bemutatjuk a ProgTábor koncepcióját, amely egy új informatikai tehetséggondozó program. A cikk megírása előtt kevesebb mint egy hónappal tartottuk az első alkalmat, a 3. fejezetben ennek részleteit és menetét ismertetjük. A 4. fejezetben részletesen elemezzük a diákok visszajelzéseit, amelyeket több rövid, és egy hosszabb kérdőívvel gyűjtöttünk, kiemelve a legérdekesebb válaszokat és a megfigyelhető trendeket. Végül az 5. fejezetben egy rövid konklúzióval foglaljuk össze az eredményeket.

## 2. A ProgTábor koncepciója

Ebben a fejezetben röviden bemutatjuk a táborok céljait és megvalósítási részleteit áttekintő jelleggel, annak érdekében, hogy a továbbiakat kontextusba tudjuk helyezni.

### 2.1. Résztvevők

A tábort 7-11. osztályos, algoritmikus programozásban tehetséges diákoknak tartjuk. 20-30 fős csoportokat képzünk egy vagy két szomszédos évfolyamról, egy csoport együtt marad az 5 éven keresztül, de lehet belépni és kilépni. Később nyitni szeretnénk kisebb korosztály felé, 6., majd 5. osztályban kezdenénk.

## 2.2. Tananyag

Problémamegoldást tanítunk számítástudományi gondolkodással (problem solving with computational thinking), konkrétabban algoritmusokat, adatszerkezeteket és programozást. Mindezt feladatokon keresztül felépítve, ahogy A Gondolkodás Öröme Alapítvány matematika táboraiban is van [4]. Egy ilyen tananyag összeállításáról szól egy korábbi cikk [3]. A programozási nyelvet nem tanítjuk, csak eszközként tekintünk rá. Nagy hangsúlyt fektetünk arra, hogy az elméletben megfogalmazott algoritmusokat implementálják és teszteljék is a diákok. Korábbi versenyfeladatokat használunk erre, online kiértékeléssel.

## 2.3. Cél

A tábor célja az informatika feladatokon való gondolkodás örömét megmutatni a diákoknak, kinyitni ezt a világot a diákok számára, és ebben színvonalas képzést adni nekik, egy elfogadó, sokszínű közösségben. Nem elsődleges cél a versenyfelkészítés, de azt várjuk, hogy 5-10 éven belül jobban fel tudjuk venni a versenyt a környező országokkal (a lengyel, román, bolgár, szlovák, horvát diákok is előttünk járnak jelenleg az IOI érmek számát figyelembe véve [5]) a nemzetközi programozás versenyeken. Közösséget szeretnénk építeni, ami hosszú távon a program kiterjesztésére alkalmas.

## 2.4. Módszertan

A táborokban alkalmazott pedagógiai módszertant Pósa Lajos matematika tehetséggondozó táborai alapján dolgoztuk ki [2], amelyet a következő években a táborok alatt finomítunk, továbbfejlesztünk. Ez a módszer egyfajta irányított felfedeztető tanítás, és mivel a középpontban megoldandó problémák vannak, ezért problémaközpontú tanulásnak nevezzük. A tantervet egymásra épülő feladatok adják. A nevezetes algoritmusokat, adatszerkezeteket nem tálaljuk előre, hanem egy-egy bevezető feladat (vagy feladatsor) segítségével megpróbáljuk felfedeztetni a diákokkal. Így a problémamegoldáson van a hangsúly, és a problémák megoldásainak megbeszélésekor tárgyaljuk a nevezetes algoritmusokat. Ilyenkor természetesen megbeszéljük és tudatosítjuk az adott algoritmus vagy adatszerkezet „tankönyvi" verzióját és implementációját. Az egymáshoz kapcsolódó feladatok rendszerét a Pósa-módszerben egy úgynevezett *Web of Problem Threads* (Feladatszálak Hálózata) keretbe helyezhetjük [6]. Ilyen keretbe illesztve alakítottuk ki a tananyag kezdeti változatát [3], ami folyamatosan fejlődik és bővül, minden táborban kerülnek bele új feladatok.

Nagyon lényeges a foglalkozások szervezési módja. A munka legnagyobb része 2-4 fős csoportokban zajlik, amelyeket a gyerekek alakítanak, és egy hétvégére együtt maradnak. Minden csoport munkáját mentorok kísérik figyelemmel, akik a visszajelzések mellett segítségeket is adnak nekik. Frontális módszerrel csak a feladatok feladása, és a megoldások megbeszélése történik (miután már kellő idejük volt a megoldásra a diákoknak). A matematikától eltérő módon az informatikában a megoldást két lépésre is lehet bontani, külön megbeszélni az elméleti megoldást, és még ezután időt adni a kódolásra a diákoknak. Ilyenkor egy következő alkalommal nézzük meg közösen magát a programot. Az esetek többségében kívánatos ezt így tenni, mivel az önálló implementáció a megértéshez képest egy következő szintet jelent.

## 2.5. Megvalósítás

A Gondolkodás Öröme Alapítvány ennek a tehetséggondozó programnak a gazdája, de erősen kapcsolódunk az Eötvös Loránd Tudományegyetem Informatikai Karához és a Neumann János Számítógéptudományi Társasághoz is. A táborok vezetője egyelőre Nikházy László (a cikk szerzője), de pár éven belül a rendszeresen segítő mentorokat is szeretnénk ebben a szerepben látni. A mentorok korábbi versenyzők közül, illetve programozást vagy algoritmusokat tanítók közül kerülnek ki, és kezdetben támaszkodunk a matematika táborok közösségére is. Évek múltán a felnövő résztvevők veszik át ezt a szerepet. Közösséget szeretnénk építeni, ez igen fontos a jövőre nézve, a program kiszélesítése szempontjából.

A diákokat jelentkezés után választjuk ki, erre a célra kidolgozott feladatsorokkal. Emellett a versenyeken jól szereplő diákokat megkeressük külön is a jelentkezést ajánlva nekik. A táborokat tanítási időszakban, hétvégente tartjuk, egy csoportnak évi 3-4 alkalommal. Ilyenkor péntek délutántól vasárnap délutánig egy helyszínen gyűlünk össze, ahol a szállást és a napközbeni munkát össze tudjuk kötni. Az alábbi táblázatban megmutatjuk egy hétvége lehetséges programját, hogy még jobban el lehessen képzelni, mi zajlik majd egy ilyen táborban. Természetesen ez még alakul, és az időbeosztás csak hozzávetőleges.

| **PÉNTEK** | |
|---|---|
| **16:00** | Érkezés |
| **16:30 - 17:30** | Házi feladatok megbeszélése, kérdésekre válasz |
| **17:30 - 19:00** | Rövid egyéni „verseny", ahol felmérjük a korábbi anyagok elsajátítását |
| **19:15 - 20:00** | A feladatok megbeszélése, elemzése, jutalmazás |
| **20:00 - 22:30** | Vacsora, esti játék |
| **SZOMBAT** | |
| **9:00 - 12:30** | Csoportmunkás foglalkozások és közös megbeszélések váltakoznak, új témaköröket tanulunk, közben 30 perc szünet |
| **12:30 - 13:30** | Ebédszünet |
| **13:30 - 14:00** | Szabad program |
| **14:00 - 15:30** | Sport, vagy vetélkedő (gépek nélküli program!), utána rövid pihenő |
| **16:00 - 18:30** | Csapatverseny |
| **18:30 - 19:30** | Csapatverseny feladatainak elemzése |
| **19:30 - 22:30** | Vacsora, esti vendéggel beszélgetés vagy játék |
| **VASÁRNAP** | |
| **9:00 - 12:30** | Csoportmunka, megbeszélések, közben 30 perc szünet |
| **12:30 - 13:30** | Ebédszünet, szabad program |
| **13:30 - 15:30** | Valami mókás program (például blind contest) |
| **15:30** | Pakolás, hazautazás |

**1. táblázat:** Egy tábor lehetséges időbeosztása

## 3. Az első kísérleti tábor

Az első, kísérleti tábort a COVID-19 járvány miatt először el kellett halasztanunk 2020 tavaszáról későbbre, majd online térben tartottuk meg 2020. október 24-25-én (szombat-vasárnap). Így több szempontból is különleges volt, de a koncepcióra nézve is releváns tapasztalatokat és visszajelzéseket gyűjtöttünk, amelyeket a továbbiakban megosztunk. Ebben a fejezetben röviden bemutatjuk, hogy milyen szervezésben, kikkel és hogyan zajlott a tábor.

### 3.1. Csoport

A csoportot immár 10.-es évfolyamba lépő diákok alkotják, akiket meghívásos úton válogattunk össze, döntő többségben a Nemes Tihamér Programozási Verseny eredményei alapján, illetve ismeretségek útján. Feltétel volt az alapszintű C++ tudás, a tehetséget nem mértük fel magunk, mivel versenyeredményeket tekintettünk (adott esetben akár matematikából is). Az első táborban 24 diák

vett részt, 5 lány és 19 fiú. A társaság előzetes tudása a témakörben igen széles skálán mozgott, viszont a közösségépítés célja miatt nem akartuk kihagyni a diákolimpikonokat sem, és a tehetséges, programozás iránt érdeklődőket sem.

A táborban 2-4 fős csoportokban dolgoztak a diákok, összesen 9 csoport jött létre a tábor elején. Őket mentorok segítették, akik korábbi versenyző egyetemisták és fiatal szoftverfejlesztők, matematikusok. Ezúton is köszönöm a lelkes segítséget Leitereg Andrásnak, Dankovics Attilának, Radnai Lászlónak, Busa Máténak, Molnár-Sáska Zoltánnak, Vári-Kakas Andornak és Deák Bencének. A diákok visszajelzéseiből látszik, mennyire fontos volt az ő jelenlétük. Az online lebonyolítás miatt a segítőknek nem kellett végig ott lenniük, kiegészítették egymást, de mindig jelen volt legalább négy. Így a táborvezetővel együtt elosztva a csoportokat, minden mentornak egy vagy két csoport jutott.

## 3.2. Online megvalósítás

Az online kommunikáció és közös munka elsődleges színtere az erre a célra létrehozott ProgTábor Discord szerver volt. A Discord [7] egy kiváló kommunikációs platform, amelyen belül a „szerver" tulajdonképpen egy közösségi teret jelent, sok szöveges és hangcsatornával, különböző szerepekkel és jogosultságokkal, nagyon sok integrációval. A csoportmunkát remekül meg lehetett szervezni privát (hang- és szöveges) csatornákkal, ahová az adott csoport tagjain kívül a mentorok tudtak belépni. A mindenkit érintő tartalmakat is külön szöveges csatornákba tudtuk rendezni, például egy-egy külön csatornát használtunk a feladatok listájára, a fontos hirdetményekre, és a megbeszélések közbeni gyors információmegosztásra is. A közös megbeszélések egy hangcsatornában zajlottak, a közös látványt alternatív módszerekkel oldottuk meg, amelyek jobb kollaborációt tesznek lehetővé, és jobban skálázhatóak, mint a videó stream.

Például a közös kódolást egy CodeInterview [8] elnevezésű (demó módban tökéletesen jól) ingyenesen használható online megosztott programozási környezetben valósítottuk meg. Ezt programozás állásinterjúkra és párban programozásra fejlesztették, és remekül alkalmas arra, hogy valós időben többen írjanak programot egy felületre (vagy csak nézzék azt). A fordítást és futtatást egy gombnyomásra elvégzi egy-egy megadott bemenetre, így a táborban szereplő feladatokra nagyon kényelmesen használható. A diákok is pozitív visszajelzéseket adtak róla. A megbeszélésekkor az ábrát igénylő magyarázatokhoz A Web Whiteboard [9] nevű online megosztott táblára rajzoltunk, ami a célnak szintén kiválóan megfelel, és teljesen ingyenes. Mindkét eszköz anonim módon használható, egy link küldése elég az együttműködéshez, ezt a Discord-on könnyen meg tudtuk tenni. Megjegyezzük, hogy ezen eszközök közül egy személyesen megrendezett táborban is használtuk volna a Discord-ot, a CodeInterview-t (kisebb mértékben).

A tábor programját az online lebonyolításhoz alakítottuk. Másrészt a résztvevők elfoglaltságai miatt a két és fél naposra tervezett tábort utolsó pillanatban két naposra csökkentettük. A végső programot az alábbi táblázat mutatja.

| SZOMBAT | |
|---|---|
| **10:00 - 13:00** | Bevezető feladatok, csoportok megalakulása, csoportmunka, végén közös megbeszélés |
| **13:00 - 14:30** | Ebédszünet |
| **14:30 - 16:30** | Feladatok csoportmunkában, megbeszélés |
| **16:30 - 17:30** | Szünet |
| **17:30 - 19:00** | Rövid csapatverseny |
| **19:00 -** | Esti játék |
| **VASÁRNAP** | |
| **10:00 - 13:30** | Korábbi feladatok megbeszélése, csoportmunka |
| **13:30 - 15:00** | Ebédszünet, szabad program |
| **15:00 - 17:30** | Hosszú csapatverseny |
| **17:30 - 18:00** | Szünet |
| **18:00 - 19:00** | Csapatverseny feladatainak megbeszélése, visszajelzések, búcsú |
| **19:00 -** | Esti játék |

**2. táblázat:** A kísérleti, online tábor időbeosztása

A táborban nagyon erősen támaszkodunk az online feladatbankokra, amelyekben jellemzően korábbi programozási versenyfeladatok vannak. Így a feladatok megosztása online alapból megoldott volt. Viszont egy hagyományos táborban is ezen feladatbankok automatikus online kiértékelőjével ellenőrizzük a megoldások helyességét és hatékonyságát. Ebben a táborban három ilyen rendszert használtunk, az angol nyelvű Codeforces [10] és HackerRank [11], valamint a magyar nyelvű Mester [12] weboldalt. A Codeforces rendszerben meg tudtuk szervezni a csapatversenyt is, a feladatbankban elérhető korábbi feladatokból készített „Mashup" versenyként, szinte pillanatok alatt össze lehet állítani, és a verseny alatt is nagyon jól lehet követni a diákok munkáját. A verseny végeztével ők is meg tudják tekinteni egymás kódjait, a társaktól tanulást nagyon fontosnak tartjuk.

## 3.3. Tananyag, feladatok

Mivel ez volt az első tábori alkalom és egyes résztvevőknek az első találkozása az algoritmusok világával, ezért nagyon alapvető, klasszikusnak számító, de nem könnyű témákat terveztünk a táborra. A két fő témakör a gráfok és a dinamikus programozás voltak. Mindkét témakörben nagyon egyszerű bevezető feladatokkal indítottunk, viszont a végére elég komplex feladatokat vettünk. Ezen kívül a táborban végig voltak extra feladatok, amelyek a haladóbb diákok számára is folyamatosan nyújtottak gondolkodnivalót. A témakörök mellett néhány példa feladatot felsorolunk itt, a teljesség igénye nélkül, megadva az adott weboldalon belül az elérési útvonalat, illetve az azonosítót.

A gráfok esetében indításképpen a gráfreprezentációkra csináltunk néhány feladatot. Például, egy 5 pontú gráfra el kellett dönteni, hogy van-e benne teljes, vagy üres háromszög (*Codeforces / 94B Friends*), vagy ki kellett választani a legrövidebb élt két csúcshalmaz között (*Codeforces / 707B Bakery*). Ezután az elárasztásos kitöltés (flood-fill) algoritmuson keresztül eljutottunk a gráfbejárásokhoz, majd azok alkalmazásai voltak a tábor fő fókuszában, sok szép feladattal. Például meghatároztuk egy gráf komponenseit (*Codeforces / 893C Rumor*), két színnel színezését (*Codeforces / 1176E Cover it!*), súlyozatlan gráfban legrövidebb utakat, csúcshalmazból is (*Codeforces / 1272E Nearest Opposite Parity*). És még számos más feladatot csináltunk. Két gyönyörű, bejárásra vezető haladó feladatot említenénk még (*Codeforces / 196B Infinite Maze, Codeforces / 788C The Great Mixing*).

A dinamikus programozást három vállfaján keresztül építettük fel. A diákok erős matematikai hátteréből adódóan kombinatorika feladatokkal kezdtük (*Mester / Kombinatorikai algoritmusok / 11.*

*Lépcsők, Codeforces / 474D Flower*), amelyek aztán a két másik típushoz is kapcsolódtak. Csináltunk általunk „lépegetős DP"-ként emlegetett feladatokat (*Codeforces / 2B The least round way, Codefores / 41D Pawn*). Másrészt a Pénzfelváltás-problémán keresztül (*HackerRank / Algorithms / Dynamic Programming / The Coin Change Problem, Mester / Dinamikus programozás / 73. Postabélyegek*) eljutottunk a Hátizsák-problémakörhöz, amelyet több feladatban is kellett alkalmazni (*Mester / Dinamikus programozás / 61. Malacpersely legkisebb értékre, Codeforces 19B Checkout Assistant*).

# 4. A visszajelzések elemzése, tapasztalatok

A diákoktól a tábor közben és a végén is kértünk visszajelzést egy űrlap kitöltésével. A tábor közben minden program után egy egyszerű kérdőív kitöltését kértük, amelyen az alábbi kérdések szerepeltek:

1. Mennyire élvezted a legutóbbi foglalkozást? (1-5)
2. Mennyit tanultál/fejlődtél a legutóbbi foglalkozáson? (1-5)
3. Bármilyen rövid szöveges visszajelzés

A tábor vége után néhány nappal egy hosszabb kérdőíven kértük a diákok visszajelzését. A következő kérdéseket tettük fel nekik:

1. Érzésed szerint milyen típusú foglalkozások során fejlődtél a legtöbbet?
2. A tábornak melyik része tetszett a legjobban neked?
3. Melyik feladat tetszett a legjobban az egész táborban?
4. Mikor volt olyan helyzet, amikor egy feladat megoldása során saját magatok által kitalált módszert használtatok?
5. Milyen új dolgokat tanultál a tábor alatt?
6. Hogy működött a csoportmunka nálatok?
7. Hogy tetszett az online lebonyolítás?
8. Milyenek voltak a használt platformok?
9. Mennyire voltak érdekesek a feladatok? (1-5)
10. Mennyire volt sok a feladat (a csapatversenyeket leszámítva)? (1-5)
11. Mennyire volt gyors a tempó? (1-5)
12. Mennyire voltak érthetőek a megoldások, magyarázatok? (1-5)
13. Milyen pozitívumokat emelnél ki?
14. Milyen negatívumokat tapasztaltál?
15. Milyen változtatási javaslataid vannak?

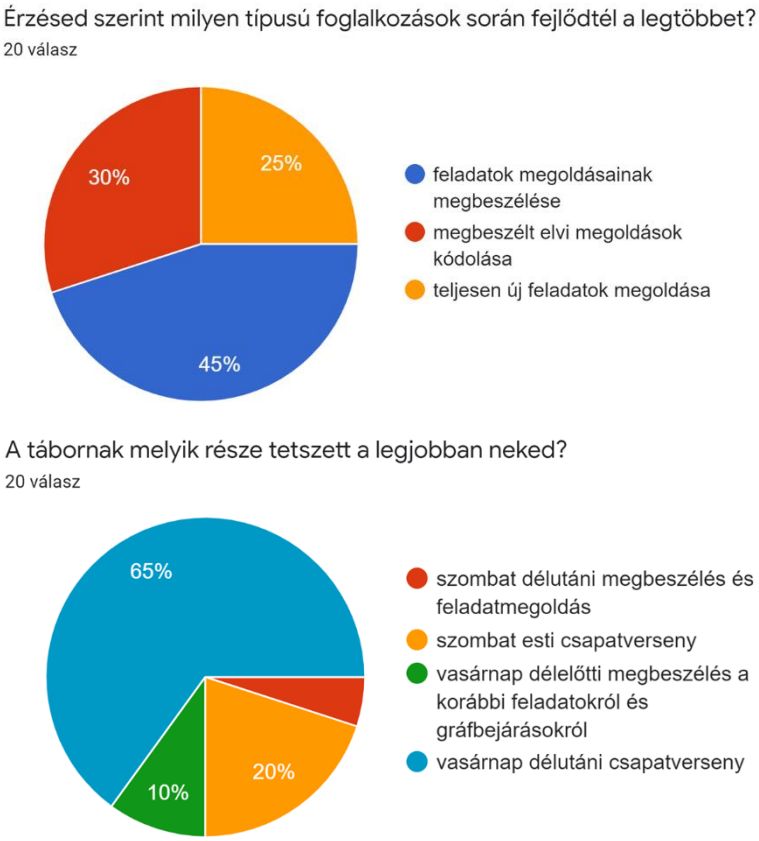
Az alábbiakban kiemelünk néhány vizsgált szempontot, amelyek kapcsán érdekes észrevételeket tettünk.

## 4.1. Élvezet és fejlődés

Önmagában is érdekes, hogy a gyerekek mennyire élvezik a foglalkozásokat, és mennyit fejlődnek (saját maguk szerint). A felfedeztető tanítás során azt várjuk, hogy jobban élvezik a tanulást, viszont kevésbé érzik úgy, hogy fejlődnek, ahhoz képest, mint ha tálalnánk nekik a tudást, és gyakoroltatnánk velük. De különösen foglalkoztat minket az, hogy a két mérőszám között milyen összefüggés van a különböző foglalkozás típusokra nézve. Mivel hasonló táborokat alapvetően csak felfedeztető módszertannal szeretnénk tartani, ezért az vizsgáljuk, hogy a különböző tevékenységek (gondolkodás feladatokon, megoldások megbeszélése, kódolás) során hogyan alakulnak az értékek. A rövid kérdőívekkel erről érdekes adatokat tudtunk gyűjteni.

Elsőként a hosszú kérdőív kapcsolódó kérdéseinek statisztikái láthatók az 1. ábrán. A diákok több, mint fele (55%) úgy érzi, hogy önálló (vagy csoportos) munka során fejlődik a legtöbbet. A válaszaik majdnem egyenlő mértékben oszlanak meg az új feladatok megoldása (25%) és a megbeszélt megoldások implementálása (30%) között. Ezzel is megerősítést nyer az a nézetünk, hogy a

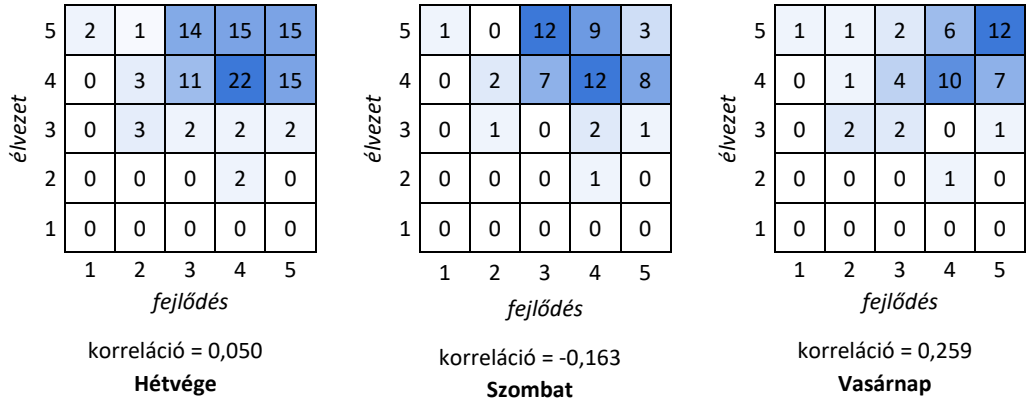
kódolást fontosnak tartjuk. A diákok legnagyobb hányada (45%) viszont a három lehetőség közül a közös megbeszélések során fejlődött a legtöbbet a saját bevallása szerint. Szeretnénk elérni, hogy ez az arány csökkenjen, ahogy egyre inkább támaszkodunk a korábbi ismeretekre, és egyre többször csak a tudatosítás, rendszerezés szerepét töltik be a közös megbeszélések, miután már szinte mindenki megoldotta a feladatokat. Az 1. ábra alsó részén pedig az látható, hogy melyik foglalkozást tetszett a legjobban a gyerekeknek. Itt elsöprő a sikere a csapatversenyeknek, összesen 85%-ban választották őket (65% a vasárnapi hosszú, 20% a szombati rövid csapatversenyt). Így annak ellenére, hogy nem a versenyzést szeretnénk középpontba állítani, ezek a vetélkedő-szerű programok kihagyhatatlanok a táborból. Másrészt az erre szánt idő (például vasárnap 2,5 óra) színtiszta gondolkodással és feladatmegoldással telik el, tehát pedagógiai céljainkra is felhasználhatók, például a korábban megismertek elmélyítésére.




**1. ábra:** A gyerekeknek legjobban tetsző, illetve legtöbb fejlődést nyújtó foglalkozások (a saját válaszaik alapján)

A rövid kérdőívek elemzésében a két mennyiség korrelációjáról érdekes megfigyelésre jutottunk. A döntő többségében pozitív visszajelzések miatt nem mozognak széles skálán az adatok. Az élvezetre vonatkozó válaszok 90%-a 4-es vagy 5-ös, a fejlődésre vonatkozóak 92%-a 3-as, 4-es, vagy 5-ös. Viszont érdekes kérdés, hogy vajon akkor fejlődnek többet a gyerekek, amikor élvezik is a foglalkozást, vagy pedig az élvezet negatívan hat a fejlődésre. A 2. ábra mutatja a válaszok együttes eloszlását táblázatos formában (a színezés mértéke csupán a számok relatív nagyságát érzékelteti). Az ábra baloldali része a teljes hétvégén gyűjtött válaszokat összesíti, és az ebből számított korreláció

(0,050) szerint a két mérték között egyik irányú kapcsolat sem figyelhető meg egyértelműen. Különválasztottuk azonban a két napot egymástól, és itt már az ábrán is szembetűnő, hogy szombaton a mellékátlóval párhuzamosan vannak nagy értékek, vasárnap pedig a főátlóban, vagyis csak a szombati adatokat tekintve a korreláció negatív, csak a vasárnapi adatokat tekintve pozitív. A két nap ugyanolyan típusú foglalkozások voltak (csoportmunka, megbeszélés, csapatverseny). Azzal tudjuk ezt magyarázni, hogy még azonos tanár által ugyanannak a csoportnak tartott, azonos típusú foglalkozások esetén is lehet minőségbeli különbség. Ráadásul ezzel összhangban van a táborvezető saját személyes érzése a két nappal kapcsolatban. Szombaton sok volt az új anyag, mégsem úgy haladtunk, ahogy terveztem, hiányérzetem volt, míg vasárnap úgy éreztem, hogy minden jól sikerül, jó a hangulat, elvégeztük azt, amit terveztem. Ez alapján az észrevétel alapján a későbbiekben meg szeretném vizsgálni azt a kérdést, hogy vajon *az a jól sikerült foglalkozás, amikor korrelál a fejlődés mértéke az élvezet mértékével?* Mindenképp több mérésre van szükség erről való értekezéshez.




**2. ábra:** Élvezet és fejlődés értékelésének együttes eloszlása és korrelációja

## 4.2. A mentorok szerepe

A kérdőíven nem kértük azt, hogy értékeljék a segítőket. Mégis, a pozitívumoknál a válaszok 20%-a a mentorokról szólt. Ezekből az derül ki, hogy a felfedeztető tanításban kulcsfontosságú az, hogy az egyéni- és csoportmunkában figyelemmel kísérjük, és segítsük a gyerekek munkáját. Ez a tény nem újdonság, a nemzetközi szakirodalom is kiemeli. Kirschner és társai [13] sok kutatás áttekintésével alátámasztják, hogy minimális útmutatással a tanítás nem működik. Válaszként Hmelo-Silver és kollégái [14] kifejtik, hogy a problémaközpontú tanulásban (problem-based learning, PBL) és a kutatás alapú tanulásban (inquiry learning, IL) folyamatos segítséget és iránymutatást kapnak a diákok, ezzel támogatják a tanulást a pedagógusok.

A mentorok a táborvezető szempontjából még egy monitorozó szerepet is betöltenek, folyamatosan információt nyújtanak a csoportok haladásáról. Erre egy megosztott táblázatot használtunk azon túl, hogy időnként beszéltünk egy külön szobában. Minden mentornak egy vagy két csoport jutott, többet nem is lehet folyamatosan nyomon követni.

Néhány példa a gyerekek visszajelzéseiből:

- *„A mentor állandó jelenléte és tanácsai sokkal aktívabbá, hasznosabbá tette számomra a tábort."*
- *„A mentorok megpróbáltak segíteni a feladatoknál, ami többnyire sikerült is nekik. A délelőtti tanulás alatt azonban hiányoltam a folyamatos segítséget, kicsit magunkra voltunk utalva."*
- *„A segítők mindig segítettek, ha egy feladat megoldásánál nem jöttünk rá, hogy mi az adott feladat megoldásának elmélete, vagy ha elírtunk valamit a programban, ami miatt nem futott le vagy nem megfelelően működött."*
- *„A segítőkész mentorokat."*

## 4.3. A csoportmunka ereje

A pozitívumokra vonatkozó kérdésre a válaszok 20%-a szólt a csoportos feladatmegoldás élményéről. Így a mentorok mellett ez volt a legerősebben reprezentált téma a pozitívumok között. Nincsenek erre vonatkozó adataink, de meggyőződésünk, hogy egyéni munka esetén sokkal kevésbé lenne élvezetes a tábor. Pósa Lajos személyes beszámolóiban és előadásaiban (például [15]) is rendszerint kiemeli, hogy akkor kezdtek el igazán jól működni a táborai, amikor bevezette, hogy a gyerekek külön szobákban lévő csoportokban dolgoznak.

Nagyon fontos megerősítés számunkra, hogy a diákok a szabad szavas visszajelzésnél ennyien említették a csoportmunkát. Néhány példa a visszajelzésekből:

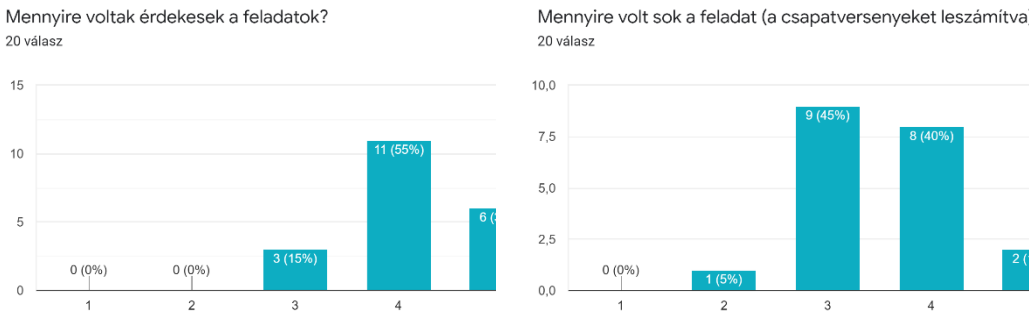
- *„Tetszett, hogy csapatokban dolgoztunk."*
- *„A csapatmunkákat, a jó időbeosztást, hogy lehetett kérdezni, és - szerintem - mindenkinek volt benne kihívás (talán az olimpiai csapattagoknak nem)."*
- *„Nagyon sokat tanultam, a barátaimmal együtt."*
- *„Jó volt hallani a feladatokra többféle megoldást, illetve ez a 3 fős csoportos ötlet nagyon tetszett nekem."*

A használt eszközöknek köszönhetően (Discord, Codeinterview) online is jól megvalósítható volt a csapatmunka. A technikai eszközökre vonatkozó kérdéseknél ez gyakran szempont volt a gyerekek értékeléseiben:

- *„A Codeinterview kifejezetten jó volt, mert mindketten láttuk a kódot."*
- *„Tökéletesen működtek, nagyban segítették a tábori munkát."*
- *„A Discord az ilyen célokra az általam látott programok közül a legoptimálisabb, a Codeinterview-t most láttam először, és jól működött, a Codeforces szintén rendben volt."*

## 4.4. Tananyag és tempó

A pozitívumokra vonatkozó válaszok 25%-a szólt a tananyagról, feladatokról vagy az időbeosztásról. Dicsérik a feladatok összeállítását és a tananyag felépítését, ami a táborvezető munkájának elismerése. Ugyanakkor a negatívumoknál is megjelennek ehhez kapcsolódó vélemények, amelyek szinte kivétel nélkül a gyors tempót emlegetik. Mindegyik téma olyan, hogy a kérdőíven szerepelt egy korábbi kérdésben, szóval nem meglepő, hogy az ottani számszerű választ fejtették ki szövegesen a pozitívumoknál és negatívumoknál.

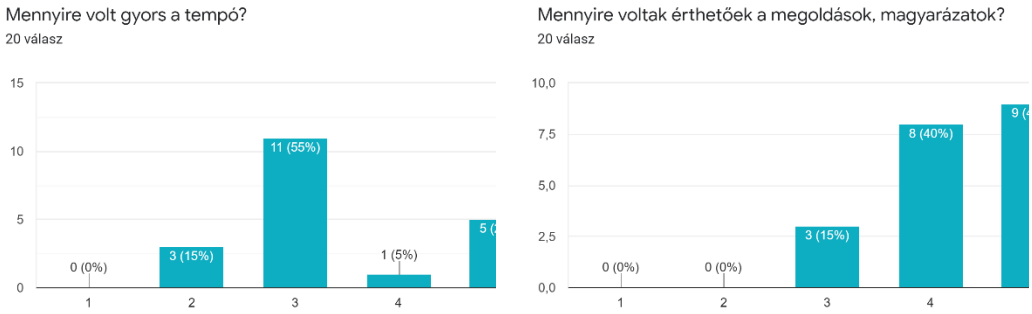



**3. ábra:** Értékelések eloszlása a feladatok érdekességére és mennyiségére vonatkozóan

A 3. ábra mutatja a feladatok érdekességére vonatkozó válaszok eloszlását. Néhány szöveges visszajelzés ehhez kapcsolódóan.

- *„Érdekesek, gondolkodtatók voltak a feladatok, mindegyik újdonsággal. Jól volt felépítve az időbeli felosztás a délutáni csapatversennyel. Látszott, hogy minden szervező felkészült és segítőkész. A tábor hatására mindenképpen intenzívebben fogok foglalkozni a programozással, mint eddig."*
- *„Voltak könnyebb és nehezebb feladatok is."*
- *„Jó időbeosztás, jól felépített tábori tananyag."*
- *„A feladatok 3-4 temakorbol voltak, nem az osszesbol, igy nem volt kavarodas a fejunkben, illetve nem ereztuk soknak az egeszet."*
- *„Tetszett hogy egy tematika körül oldottunk meg feladatokat, csak kicsit más alaphelyzettel"*

A 4. ábrán látható, hogy a tempó megítélése nagyon megosztott volt, nem emlékeztet a haranggörbére a válaszok eloszlása. A többség jóra értékelte, néhányan kicsit lassúra, és volt néhány diák, aki nagyon gyorsra. Ezek a diákok jellemzően a negatívumokban is írtak erről, és többen is említik, hogy valószínűleg ők lógnak ki a csoportból. Sajnos ezzel a helyzettel tisztában voltunk előzetesen is, nagyon széles skálán mozgott a diákok tudása, és ugyan igyekeztünk olyan szinten tartani, hogy mindenkinek érthető legyen minden, a kódolás időigényes volta miatt a kevésbé tapasztalt diákok nem tudtak olyan ütemben haladni, amely minden feladat teljesítéséhez kellett. A közös megbeszélések viszont az ő tudásuknak megfelelő szinten zajlottak, amelyet egy-két haladó diák értékelt negatívumként.




**4. ábra:** Értékelések eloszlása a tempóra és a magyarázatok érthetőségére vonatkozóan

Szöveges visszajelzések a fentiekkel kapcsolatban:

- *„Gyorsnak tűnt, de valószínűleg inkább csak én voltam lemaradva. Igyekszem pótolni a hiányosságokat."*
- *„Számomra nagyon gyorsan haladtunk, sokszor egyáltalán nem volt időnk egy feladatot elkezdeni, csak az elvi megbeszélés után"*
- *„Néha nem volt elég idő a csapatban megbeszélni es le is kódolni a feladatot."*
- *„A gyors tempó elég sok nyomást helyezett ránk, sokszor nem volt idő egy jó ötletet leprogramozni." (A javaslatokban pedig ezt írta ő: „Nem mondom, hogy legyen lassabb tempó, mert az átlagnak szerintem pont megfelelő volt, majd sietünk legközelebb").*
- *„A gyors haladás miatt nem értettem mindent teljesen"*
- *„Talán az egy kicsit negatívum, hogy viszonylag vannak nagy különbségek az emberek tudása között, és ezért nem sok újat tanultam. De remélem, idővel ez a különbség csökken."*
- *„Néha hosszú megbeszélések, túl rövid tábor (extra péntek délután vagy hétfő délelőtt még belefért volna)"*

## 4.5. Az online formátum előnyei és hátrányai

Természetesen minden résztvevőnek és szervezőnek az a véleménye, hogy egy hagyományos formában, személyes találkozással megszervezett tábor sokkal jobb lett volna. Azonban az adott lehető-

ségek mellett az online tábor nagyon pozitív élmény volt mindenkinek. Később ecseteljük a hátrányokat, de elsőként említsünk meg néhány előnyt, a gyerekek visszajelzéseit is idézve.

Az online tábor ingyenes, nagyon könnyű megszervezni (például nem kell szállás, étkezés, utazás), és sok tekintetben flexibilis. Például nem probléma, ha ütközik egy családi- vagy sportprogrammal, néhány órára lehet hiányozni. A mentorok részére sem jelent egész hétvégés kötöttséget, válthatják egymást. Mindenki a saját, kényelmes környezetében dolgozik. A rendelkezésre álló szoftverekkel és infokommunikációs eszközökkel elég jól tudunk online is együtt dolgozni és kommunikálni. Ráadásul egy részüket a hagyományos táborban is használnánk. A gyerekek visszajelzéseiből néhány:

- *„Természetesen élőben lett volna a legjobb, de véleményem szerint az online tér adta lehetőségeket maximálisan kihasználtuk, a jelenlegi helyzetben ez egy kiválóan helytálló megoldás volt."*
- *„Szépen, szervezetten lett megoldva. A kis csoportok és a közös megbeszélések is jók voltak."*
- *„Persze élőben sokkal jobb lenne, de a vártnál sokkal közelebb éreztem magam a jelenlévőkhöz."*
- *„Szerintem a foglalkozás nem vesztett sokat az értékéből, így is jó volt."*
- *„Nekem kényelmes volt, ami nyilván nem egy utolsó szempont. Mivel nagyjából ismerem a csapatot, ezért a nicknamekről is tudtam, hogy valójában ki áll mögötte."*
- *„Szerintem élőben sokminden jobb lett volna, hatékonyabban tudtunk volna dolgozni, viszont voltak előnyei is, pl le tudtam magam némítani/süketíteni, nem zavart be semmi."*

Külön rákérdeztünk a használt platformokra, ezekről lényegében kizárólag pozitívan nyilatkoztak, például az alábbiakat írták:

- *„A Discord az ilyen célokra az általam látott programok közül a legoptimálisabb, a Codeinterview-t most láttam először, és jól működött, a Codeforces szintén rendben volt."*
- *„Tökéletesen működtek, nagyban segítették a tábori munkát."*
- *„A Codeinterview kifejezetten jó volt, mert mindketten láttuk a kódot."*

A fentiek viszont mind kevesek arra, hogy a negatívumokat ellensúlyozzák. Így amint lehet, a jövőben személyesen tartjuk a táborokat. Egy hagyományos táborban sokkal több a személyes interakció, sokkal erősebb a közösségi élmény, mélyebb személyes kapcsolatok születnek. Ez rendkívül fontos a közösségépítés szempontjából. A megbeszélések és a csoportmunka is hatékonyabb élőben, mint online. Ráadásul ezek során nem kellene végig a gépnél ülni (a megbeszélések táblánál zajlanának), ami az online rendezésnek egy elég komoly problémája, és említették is a diákok. Még a játékokat is gépnél kellett játszanunk most, közös sportra nem volt lehetőség. Végezetül, de nem utolsósorban, technikai problémák is felléphetnek, ami hátráltathat egy-egy diákot, vagy adott esetben a táborvezető kiesésével az egész tábor működését is. Szerencsére ez nem volt jellemző, de egy gyerek említette a kérdőíven. A negatívumként írt szabad szöveges válaszok nagy része ezekről szólt, néhány példa:

- *„Nem jó túl sokat számítógép előtt ülni."*
- *„Nem szerencsés egy nap ennyi időt egy képernyő előtt tölteni."*
- *„A közösségi élmény kicsit hiányzott és néha úgy éreztem, hogy nehezebben tudtam koncentrálni a feladatismertetésnél mint ha élőben ott lettünk volna, de a csapatonkénti programozásnál minden jól működött."*
- *„Az egyetlen negatívum az online szervezés volt."*
- *„Volt ami nekem kicsit lassú volt, az online megoldás miatt néha lemaradtam 1-2 mondatról ami zavaró volt."*
- *„Csak a helyzet nem tetszik, hogy nem egy helyen vagyunk sajnos."*

## 5. Konklúzió

A cikkben a ProgTábor kezdeményezés rövid összefoglalása után részletesen bemutattuk az első, online megtartott alkalmat, nagy figyelmet szentelve a visszajelzésekből gyűjtött adatok elemzésének. Ez alapján kijelenthetjük, hogy a kísérleti tábor sikeres volt, és értékes tapasztalatokat gyűjtöttünk a jövőre nézve. A két nap alakulását tekintve arra a következtetésre juthatunk, hogy mindenképp hasznos egy bevezető, fél napos péntek délutánt tartani. A gyerekek szemszögéből ez a ráhangolódást szolgálja, a táborvezetőnek diagnosztikus célú, a második napot sokkal jobban lehet tervezni. A rövid visszajelzések elemzésével azt láttuk, hogy a tábor két napján más volt a gyerekek által érzett fejlődés és élvezet összefüggése, a saját tapasztalatok alapján ezt tovább szeretnénk vizsgálni, azt sejtve, hogy a két értékelés korrelációja valamilyen módon jellemzi a foglalkozások minőségét.

Sok információt nyújtottak a gyerekek szöveges válaszai a tábor pozitívumairól és negatívumairól. Megerősítést nyert, hogy a csoportmunka jelentősen feldobja a tanulás élményét, különös tekintettel a problémaközpontú felfedeztető módszertanra. A mentorok aktivitása, hozzáállása, szakértelme és személyisége nagy szerepet játszik abban, hogy a diákok miként élik meg az alapvetően minimalizált instrukcióra törekedő oktatást. A táborvezető elmúlt két évben folytatott kutatási-tervezési munkáját dicséri az, hogy a diákok méltatták a jól felépített tananyagot, ugyanakkor köszönettel tartozom segítőimnek is, kiváltképp Deák Bencének, a feladatok gyűjtésében nyújtott segítségéért.

A nagy tudásbeli különbségek és az online megvalósítás hátrányai ellenére a diákok közössége jól működött, a hétvége – a gyűjtött információk és reakciók szerint – remek légkörben telt és sokaknak adott inspirációt, ez a munkánk folytatására motivál minket.

## Irodalom


1. Juhász, P., Katona, D.: *Pósa method: Talent Nurturing in Weekend Math Camps*. In: Including the Highly Gifted and Creative Students – Current Ideas and Future Directions: Proceedings of the 11th International Conference on Mathematical Creativity and Giftedness. WTM, Münster (2019) 270–276.
2. Nikházy, L.: *Algoritmusok tanítása problémaközpontú módszerrel*. In: Bihari, Erika; Molnár, Dániel; Szikszai-Németh, Ketrin (szerk.) Tavaszi Szél 2019. I. kötet, DOSZ, Budapest (2020) 557–570.
3. Nikházy, L.: *A Problem-based Curriculum for Algorithmic Programming*. Central-European Journal of New Technologies in Research, Education and Practice 2(1) (2020) 76–96. https://doi.org/10.36427/CEJNTREP.2.1.399
4. Katona, D., Szűcs, G.: *Pósa-method & cubic geometry: A sample of a problem thread for discovery learning of mathematics*. In: Karlovitz, T.J. (ed.) Differences in pedagogical theory and practice (2017) 17–34. https://doi.org/10.18427/iri-2017-0079
5. *International Olympiad in Informatics Statistics*. Az országok listája az összes szerzett érmek száma szerint rendezve. https://stats.ioinformatics.org/countries/?sort=total_desc (utoljára megtekintve: 2020.11.15.)
6. Katona, D.: *Web of problem threads (WPT)—a theoretical frame and task design tool for inquiry-based learning mathematics*. Eleventh Congress of the European Society for Research in Mathematics Education. No. 15. Freudenthal Institut, ERME (2019)
7. *Discord*. https://discord.com/ (utoljára megtekintve: 2020.11.15.)
8. *CodeInterview*. https://codeinterview.io/ (utoljára megtekintve: 2020.11.15.)
9. *A Web Whiteboard*. https://awwapp.com/ (utoljára megtekintve: 2020.11.15.)
10. *Codeforces*. https://codeforces.com/ (utoljára megtekintve: 2020.11.15.)

11. *HackerRank*.
https://www.hackerrank.com/ (utoljára megtekintve: 2020.11.15.)
12. *Mester – online programozási feladatbank*.
http://mester.inf.elte.hu/ (utoljára megtekintve: 2020.11.15.)
13. Kirschner, P., Sweller, J., Clark, R.E.: *Why unguided learning does not work: An analysis of the failure of discovery learning, problem-based learning, experiential learning and inquiry-based learning*. Educational Psychologist, 41(2) (2006) 75–86.
14. Hmelo-Silver, C.E., Duncan, R.G., Chinn, C.A.: *Scaffolding and achievement in problem-based and inquiry learning: a response to Kirschner, Sweller, and Clark (2006)*. Educational psychologist, 42(2) (2007) 99–107.
15. Pósa, L.: *Matematika táboraim*. Természet Világa, 132. évfolyam, 3. szám (2001)
http://www.termeszetvilaga.hu/tv2001/tv0103/posa.html (utoljára megtekintve: 2020.11.15.)

# Algoritmus-vizualizációs környezetek: Az interaktivitás tanulási eredményekre való hatása

Osztián Pálma Rozália[1], Osztián Erika[2], Kátai Zoltán[3]

{[1]osztian.palma, [2]osztian, [3]katai_zoltan}@ms.sapientia.ro
Sapientia EMTE Marosvásárhelyi Kara Matematika-informatika tanszék

**Absztrakt.** A számítógépes gondolkodás az egyik legfontosabb és legalapvetőbb készség, amellyel minden XXI. századi embernek rendelkeznie kell ([21]). A számítógépes algoritmusok bevezetése az oktatásba nemcsak a hallgatók programozási képességeit, hanem számítógépes gondolkodását is javíthatja ([5]). Turing ([19]) szerint a számítógépes algoritmusok működésének megértése azt feltételezi, hogy a hallgatók el tudnak képzelni egy *„egyértelmű mentális képet a gép állapotáról a számítás minden egyes pillanatában"*. Mivel a számítógépes algoritmusok absztrakt dinamikus folyamatok, ezek megjelenítésének leggyakoribb eszközei lehetnek az animációk. Berney és Bétrancourt ([2]) kihangsúlyozzák, hogy egy koherens mentális modell animációkból való felépítését nagymértékben befolyásolhatják a tanulók egyéni jellemzői (például előzetes ismeretek). További kritikusan fontos tényező lehet az oktatási anyag bemutatása, a tanulási feladatok jellemzői, valamint az, hogy milyen mértékben vannak a felhasználók bevonva a tanulási folyamatba.

Ebben a tanulmányban elsősorban arra összpontosítottunk, hogy a hallgatók algoritmus-vizualizációs (AV) folyamatokba való bevonása hogyan befolyásolhatja a tanulási eredményt. Az AV területén végzett kutatások vegyes eredményekhez vezettek. Az egyik legfontosabb tanulmány ([9]), mely ennek hatékonyságáról számol be, arra a következtetésre jutott, hogy a módszer, amelyet alkalmazunk a vizualizációk során sokszor fontosabb, mint maguk a vizualizációk. Shaffer és munkatársai ([18]) gondolataival élve kijelenthetjük, hogy az AV-k pedagógiailag hasznosak kell legyenek, támogatniuk kell a hallgatói interakciót és az aktív tanulást. Éppen ezért, kutatásunkhoz olyan online környezetet választottunk, amely mindezt lehetővé tudja tenni.

Tanulmányunkat az AlgoRythmics ([7]) tanulási környezetben valósítottuk meg, amely tánckoreográfiai illusztrációkkal és interaktív absztrakt animációkkal szemléltet tíz számítógépes algoritmust (rendezési és keresési stratégiák). Jelen tanulmány a Shell rendezés algoritmusa köré épült. Az algoritmus lépéseit szemléltető AlgoRythmics animációk három úgynevezett interaktív „jóslás" funkcióval rendelkeznek: *nincs-interaktivitás* (az oktatási anyag passzív megtekintése: a hallgatók független megfigyelők), *fél-interaktivitás* (a hallgatók részlegesen vannak bevonva az algoritmus-vizualizációs folyamatba: bizonyos kulcsfontosságú pillanatokban az animáció hirtelen megáll, és felhasználói beavatkozás szükséges) és *teljes-interaktivitás* (a felhasználók teljes irányítást kapnak: a hallgatókat felkérjük, hogy „vezényeljék le" a teljes algoritmusra vonatkozó lépéseket). Ennek megfelelően vizsgálataink arra összpontosítottak, hogy az interaktivitás különböző szintjei milyen hatással vannak a hallgatók tanulására. A kísérletet elsőéves hallgatókkal végeztük.

**Kulcsszavak:** interaktivitás, algoritmusok, animációk, online oktatási környezetek

## 1. Bevezető

A technológia és a világ előrehaladásával az online oktatási környezetek egyre inkább elterjedtek és egyre népszerűbbek az algoritmus-oktatásban. Azt tapasztalhatjuk, hogy ezek az online környezetek nagyon gyakran tartalmaznak látványos vizualizációkat melyek jelentősen elősegítik az algoritmusban való elmélyülést és annak megértését. Ezeknek a vizualizációknak egyik leggyakoribb formája a

kifejező animációk, melyek absztrakt jellegüknek köszönhetően manapság már kritikusan fontos ábrázolási technikának számítanak. A vizualizáció mellett, egy másik nagyon fontos jellemzője az online oktatási környezeteknek az, hogy milyen szinten vonják be a felhasználót a tanulási folyamatba.

Az aktív tanulás különböző formáiról szóló előzetes szakirodalmi kutatások vegyes eredményekhez vezettek. Míg néhány kutató azt vallotta, hogy a felhasználói irányítás hasznos lehet, mások azt a következtetést vonták le, hogy az animációs folyamat megszakításának negatív hatása is lehet.

Kutatásunk megvalósításához az AlgoRythmics online oktatási környezet választottuk, mely tíz algoritmus vizualizációját tartalmazza (rendező- és kereső stratégiák). Emellett, lehetővé teszi az aktív tanulás (interaktivitás) különböző szintjeivel való oktatást, melyek közül jelen kutatásban hármat használtunk fel: nincs-interaktivitás (passzív megtekintés), fél-interaktivitás (előre meghatározott kulcsmomentumokban az animáció megáll, és felhasználói interakcióra van szükség), teljes-interaktivitás (a tanulók „le kell vezényeljék" az algoritmus különböző mozzanatait elejétől a végéig). Jelen tanulmányban elsősorban azt tűztük ki célul, hogy az aktív tanulás különböző formái melyeket az algoritmus-vizualizációk (AV) során használunk milyen mértékben befolyásolják a tanulás eredményességét. Ezt a jelenséget az AlgoRythmics környezetben megtalálható Shell rendezés vizualizációjának segítségével vizsgáltuk. A kutatásban résztvevői elsőéves egyetemista hallgatókból tevődött össze, akik különböző programozási tapasztalattal rendelkeztek. Kutatásunk eredményei közül néhányat bemutattunk a Svédországban megszervezett *Frontiers in Education konferencián*. Jelen tanulmányban szeretnénk ennek kibővített változatát bemutatni.

## 2. Az aktív tanulás algoritmus-vizualizációra vonatkozó formái

Az interaktív vizualizációkat már az 1980-as évektől kezdődően alkalmazták az oktatásban. 2002-ben Naps és munkatársai [15] azt állapították meg, hogy a vizualizáció segíthet (szinte minden válaszadó, aki részt vett a felmérésben egyetértett a következő kijelentéssel: *„A vizualizációk segít a tanulóknak a számítástechnikai fogalmak elsajátításában"*). Érdekes módon, ugyanabban az évben megjelent egy metaanalízis [9], mely ezzel ellentmondásos és vegyes eredményeket mutatott be, ami a vizualizációk pedagógiai felhasználását illeti. Amikor Naps és munkatársai újra elemezték a Hundhausen és munkatársai [9] által végzett metaanalízisben szereplő huszonegy kísérletet észrevették, hogy tizenkettőből tíz (83%) kísérlet, amely alkalmazta a felhasználók tanulási folyamatba való bevonását, jelentős eredményt hozott. A további kilenc kísérlet közül, amely manipulált ábrázolásokat tartalmazott, csak három (33%) vezetett jelentős eredményhez. Ezen megfigyelés alapján Naps és munkatársai [15] kijelentették, hogy az, amit a tanulók tesznek sokkal nagyobb kihatással lehet a tanulási eredményekre, mint az, amit csak látnak. Más szóval arra a következtetésre jutottak, hogy az AV technológia csekély oktatási értékű, hacsak nem vonja be a tanulókat egy aktív tanulási tevékenységbe.

Emellett, Naps és munkatársai [15] javasoltak hat kulcsfontosságú elvet, mely az aktív tanulás szintjeinek különböző formáit szemlélteti: 1) vizualizáció nélküli, 2) megtekintés (a hallgatók passzív módon látják az AV-t), 3) válaszadás (a hallgató válaszol a tartalommal kapcsolatos kérdésekre, miközben AV-t néz), 4) változtatás (a hallgató megváltoztatja az AV-t, például az algoritmus bemeneti adatainak megadásával), 5) levezénylés (a hallgató interaktív módon levezényli egy adott bementre az AV-t), 6) bemutatás (a hallgató bemutat egy AV-t másoknak).

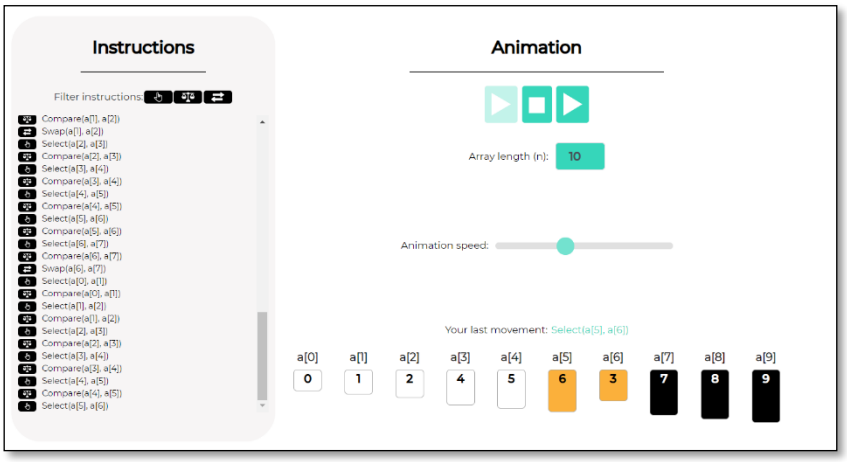
Továbbá, egy Grissom, McNally és Naps [6] által kidolgozott kutatás összehasonlította három csoport eredményét, akik az aktív tanulás különböző szintjén bemutatott vizualizációkkal tanultak: 1) vizualizáció nélküli (nem tartalmazott vizualizációt), 2) megtekintés (a vizualizáció egyszerű megtekintése), 3) válaszadás (felhasználói interakciók a vizualizáció során). A kutatók azt a következtetést vonták le, hogy minél magasabb szinttű felhasználói bevonást alkalmaztak, annál jobb tanulási eredményeket értek el.

Jelen kutatás kapcsán három szintjét különböztettük meg az aktív tanulás formájának: megtekintés, válaszadás és levezénylés. A válaszadás elvét az „interaktív jóslás" segítségével valósítottuk meg, míg a levezénylés szint biztosításához Karavirta és Shaffer [11] tanulmányát vettük alapul. Ők azt fogalmazták meg, hogy ez a típusú szemléltetési forma egy adatszerkezet és egy algoritmust ad alapul, és elvárja a hallgatóktól, hogy szimulálják az algoritmust. Ez tulajdonképpen azt jelentette, hogy az AV előrehaladásának érdekében a hallgatókat felkérjük az algoritmus konkrét lépéseinek a levezénylésére. Mindezt úgy, hogy a környezet nyújtotta interfészt ennek megfelelően manipulálják. Ezt a három feltételt a továbbiakban nincs-interaktivitás, fél-interaktivitás és teljes-interaktivitás fogalmakkal fogjuk azonosítani.

A dinamikus vizualizációkkal való tanítás és tanulás sokrétű kutatási terület, amely összefonódó tényezőket is magában foglal. Két fontos alapelv, melyek a dinamikus reprezentációk interakciótervezéséhez kapcsolódik: szegmentálás és ütemezés [16]. Megfigyelhető, hogy a fél-interaktivitás implicit módon szegmentálást generál, és a teljes-interaktivitás a műveleteknek a tanuló általi ütemezését, levezénylését követeli. Továbbá, a tanulási stílus szempontjából megkülönböztethetjük a funkcionális interaktivitást és a kognitív interaktivitást [16]. Bár a tanulmány a kognitív interaktivitás jellemzőire összpontosít, az elemzett körülmények különbözhetnek a funkcionális interaktivitás szempontjából. Általánosabban megfogalmazva: több interdiszciplináris tényező együttes hatása (pszichológiai tényezők, animációs formatervezés, tanulási környezet, didaktikai megvalósítás stb.) kínálhat megbízható magyarázatot arra, hogy e kutatási terület eredményei miért ellentmondásosak.

## 3. Az AlgoRythmics online oktatási környezet

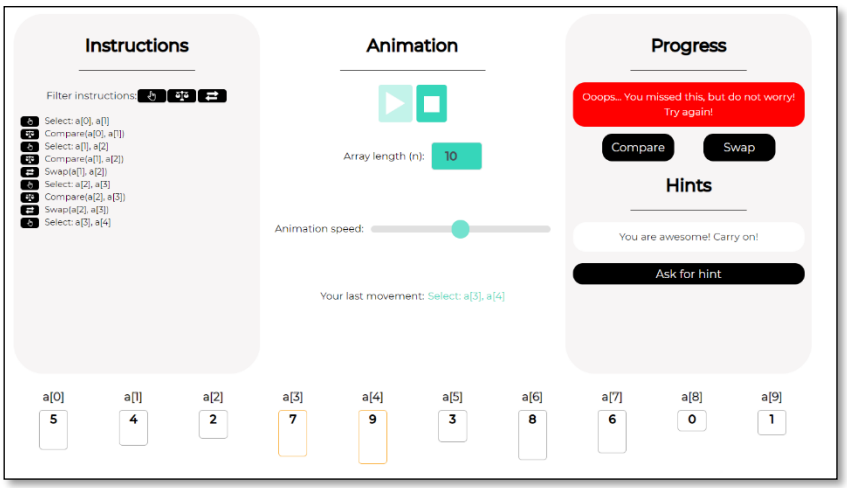
Az AlgoRythmics oktatási környezet, mely segítségével a kutatást végrehajtottuk, öt tanulási lépés köré épül (videó, animáció, irányítás, kódépítés és életre kelt kód) és lehetőséget biztosít arra, hogy különböző módon vonjuk be a felhasználókat a tanulási folyamatba. Az animációk három üzemmódban működnek: megtekintés, válaszadás és levezénylés. Megtekintési módban (nincs-interaktivitás) a hallgatók független megfigyelők, és a következő alapvető felhasználói lehetőségek állnak rendelkezésre: lejátszás, szünet, leállítás gombok és animációs sebességcsúszka (**1. ábra**).




**1. ábra:** Megtekintés mód (nincs-interaktivitás)

Válaszadó módban (fél-interaktivitás) a hallgatók részlegesen vannak bevonva a tanulási folyamatba, ami azt jelenti, hogy egy előre meghatározott kulcsfontosságú pillanatban az animáció hirtelen megáll, és felhasználói beavatkozásra van szükség. A levezénylés módban (teljes-interaktivitás) a

tanulóknak irányítaniuk kell a teljes animációs folyamatot, ők az algoritmus, a műveletek megvalósítójának építői. A következő helyes művelet kiválasztásához az alábbi felhasználói lehetőségek állnak rendelkezésre mind válaszadó, mind levezénylés módban: elemek kiválasztása, összehasonlítás és csere (**2. ábra**). Amennyiben a felhasználó nem tudja, hogyan kell folytatni a helyes műveletek sorozatát, a környezet segítséget biztosít a folytatáshoz.




**2. ábra:** Válaszadás (fél-interaktivitás) és levezénylés (teljes-interaktivitás) mód

Tanulmányunkban a három interaktivitási szinthez egy-egy jól meghatározott csoportot társítottunk: G1 (nincs-interaktivitás), G2 (fél-interaktivitás) és G3 (teljes-interaktivitás). Kísérletünkben minden résztvevőnek egyszer állt lehetőségében elvégezni az adott tanfolyamhoz tartozó lépéseket, illetve minden AV esetében a bemenet egy tanár által előre meghatározott számsorozat volt.

## 4. Kutatási kérdések

Az előzetes szakirodalmi kutatások alapján azt feltételeztük, hogy az aktív tanulás szintjének növelésével a tanulás eredménye is növekedni fog. Ennek kapcsán a következő kutatási kérdéseket fogalmaztuk meg:

- **K**$_1$: Hogyan befolyásolja az interaktivitási szint a tanulási eredményt?
- **K**$_2$: Milyen mértékben befolyásolja az algoritmus megértését a hallgatók előzetes programozási tapasztalata?
- **K**$_3$: Van-e szignifikáns különbség a nemek teljesítménye között?
- **K**$_4$: Van-e összefüggés az interaktivitás szintje és az elsajátított tudás jellege között?

## 5. Módszertan

A kutatás egy három fázisból *(előteszt, tanulási fázis, utóteszt)* álló kísérletet foglalt magába, mely a 2019/2020 – as tanévben valósult meg, a Sapientia Erdélyi Magyar Tudományegyetem Marosvásárhelyi karán.

### 5.1. Résztvevők

A kísérleten összesen 137 hallgató vett részt, melyek közül három nem töltötte ki helyesen az előzetes kérdőívet, így a kutatás során 134 hallgató (14% nő) eredményeit összesítettük. A résztvevők a

Sapientia Erdélyi Magyar Tudományegyetem elsőéves hallgatóiból tevődtek össze, 5 különböző szakról: Automatika, Gépészmérnöki, Informatika, Mechatronika és Számítástechnika. A résztvevőket három kategóriába soroltuk az előzetes programozási tapasztalatok alapján: *Nincs előzetes programozási tapasztalat* (*NP*: egy évet sem tanultak programozást a középiskolás évek során), *Alap programozási tapasztalattal* (*BP*: 1,2 vagy 3 évet tanultak programozást a középiskolás évek során, természettudomány osztály diákjai; heti 1-2 programozás óra) és *Magas programozási tapasztalattal* (*HP*: 4 évet tanultak programozást a középiskolás évek során, matematika-informatika osztály diákjai; heti 5-7 programozás óra) rendelkező diákok. A különböző programozási tapasztalat szerinti csoportokhoz tartozó diákok véletlenszerűen lettek elosztva három csoportba: $G_1$, $G_2$ és $G_3$.

A tanulási fázis, valamint az utóteszt során 46 hallgató képezte a $G_1$ csoport tagjait, míg a $G_2$ és a $G_3$ csoportokba 44 hallgató tartozott (a tesztet helytelenül kitöltő hallgatók a $G_2$, illetve $G_3$ csoporthoz tartoztak).

A nők átlagos eloszlása a különböző csoportokra vonatkozóan a következő volt: 10.86%, 18.18%, 13.63%, melyek nem vezettek szignifikáns különbséghez (Chi-négyzet próba: $p = 0.6 > 0.05$). Az előzetes programozás tapasztalat szerinti átlag 2.17, 2.11, illetve 2.20 (év) a $G_1$, $G_2$ és $G_3$ csoportok esetén, mely szintén nem eredményezett szignifikáns különbséget (Chi-négyzet próba: $p = 0.98 > 0.05$) (1. táblázat).

| Összesen: 134 hallgató | | $G_1$ | $G_2$ | $G_3$ |
|---|---|---|---|---|
| Nem | Férfi | 41 | 36 | 38 |
| | Nő | 5 | 8 | 6 |
| Előzetes programozási tapasztalat | NP | 14 | 13 | 19 |
| | BP | 15 | 11 | 18 |
| | HP | 13 | 13 | 18 |
| Előzetes programozási tapasztalat szerinti átlag | | 2.17 | 2.11 | 2.20 |

**1. táblázat**: A résztvevők eloszlása az előzetes programozási tapasztalat és nemek szerint

## 5.2. Kutatási eszközök

Az előzetes kérdőív, előteszt és utóteszt a Socrative online platformon volt megvalósítva. Az előzetes kérdőív 8 kérdésből állt: 1 személyes adatok feldolgozására vonatkozó kérdés, 2 demográfiai adatokra vonatkozó kérdés és 5 kérdés az előzetes programozási tapasztalattal kapcsolatban (Hány évet tanultak programozást a középiskolás évek során? Ismerősek-e számukra a következő rendező algoritmusok: buborék-, kiválasztó-, beszúró- és shell-rendezés?).

Annak érdekében, hogy felmérjük a résztvevők számítógépes gondolkodását, az előteszt kérdései közé 8 olyan kérdést választottunk, melyeket programozási tapasztalattól függetlenül meg tudtak válaszolni a hallgatók. Ezeket a Bebras verseny [8] honlapjáról választottuk ki.

Az AlgoRythmics online oktatási környezet biztosította a tanulási fázis megvalósítását, mely során a shell rendezés animációi kerültek bemutatásra (7 elemű számsorozaton), mivel a résztvevők közül csak 7% - nak volt ismerős ez a rendezési stratégia, és egyikük sem tanulta a középiskolában. A kísérlet magába foglalta az animációnak mindhárom formáját: megjelenítés, válaszadás és levezénylés.

Az utóteszt a következő 12 kérdésből állt (pontozás: 0-12):

- **$K_{1\text{-}6}$**: Tekintsük a növekvő sorrendbe rendező „3-1 Shell rendezést" (3 és 1 lépésköz) az x[0..6] = {1, 19, 7, 8, 12, 11, 9} hét-elemű számsorozaton. Művelet alatt hasonlítást vagy cserét értünk. Melyik az első három lépés (jelölje meg a műveletet és a cserélendő vagy összehasonlítandó elempárokat)? ELSŐ/MÁSODIK/HARMADIK (x[?], x[?])
- **$K_{7\text{-}8}$**: Tekintsük a növekvő sorrendbe rendező „3-1 Shell rendezést" (3 és 1 lépésköz) az x[0..6] = {1, 19, 7, 8, 12, 11, 9} hét-elemű számsorozaton. Melyik két elem kerül összehasonlításra az összehasonlít(x[3], x[6])/ összehasonlít(x[5], x[6]) műveletek után?
- **$K_{9\text{-}12}$** (általánosítás): Hány összehasonlítás/ csere műveletet hajt végre egy növekvő sorrendbe rendező „3-1 Shell rendezést" (3 és 1 lépésköz) egy 7 hosszúságú növekvő/ csökkenő sorozat esetén?

## 5.3. A kutatás menete

A kutatás 2 órát vett igénybe. Ebből 30 perc volt az előteszt, amelyet egyszerre tartottunk meg mindhárom csoportnak az egyetem egy előadótermében. Mivel a résztvevők közül sokak számára nem volt ismerős az AlgoRythmics online oktatási környezet, a tanulási fázis felvezetőjeként tartottunk egy rövid bemutatót a platformról. A 15 perces szemléltetés során a beszúró rendezés animációit mutattuk be, mindhárom módban.

Miután mindhárom csoport ($G_1$, $G_2$ és $G_3$) résztvevői külön laboratóriumba vonultak kezdetét vette a tanulási fázis. Ezalatt, mindenkinek regisztrálnia kellett az online oktatási környezet oldalára, ahol azt követően elvégezhették a számukra kijelölt tanfolyamtípust. Ez legtöbb 30 percet vett igénybe, mely időkorlát a $G_3$ csoport függvényében volt meghatározva, hiszen a $G_1$ és $G_2$ csoportok lévén, hogy egyszerűbb feladatot kaptak, hamarabb be kellett fejezzék a tanfolyam elvégzését). Minden csoport két alkalommal követhette végig az algoritmus-vizualizáció animációját. Először, mindhárom csoport a vizualizáció alapformáját [15] tekinthette meg, vagyis egyszerű megtekintést *(nincs-interaktivitás)*. Ezt követően, a második megtekintés során a $G_1$ csoport *nincs-interaktivitás*, a $G_2$ csoport *fél-interaktivitás*, míg a $G_3$ csoport *teljes-interaktivitás* módban tekinthette meg az animációt.

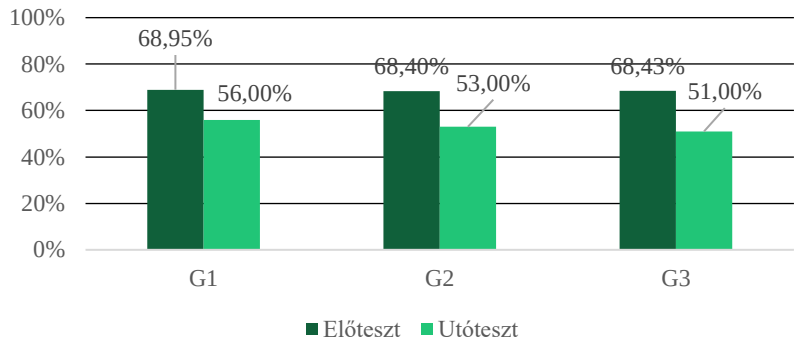
A kutatás utolsó fázisaként következett az utóteszt, melyet minden csoport ki kellett töltsön. Ennek kitöltése hozzávetőlegesen 20 percet vet igénybe.

# 6. Eredmények

Az eredmények feldolgozásához SPSS statisztikai szoftvert alkalmaztunk. A következő eredmények már bemutatásra kerültek a *Frontiers in Education* svédországi konferencián.

Az előteszten elért eredményeket egyszempontos variancia-analízissel (ANOVA) vizsgáltuk. A független változó az aktív tanulásra vonatkozó oktatási feltétel volt (nincs-, fél-, teljes-interaktivitás), míg a függő változó az előteszten elért pontszám. A Levene próba azt mutatta, hogy a csoportok azonos varianciájúak voltak, így a szóráshomogenitás elve teljesült ($p = 0.93 > 0.05$). Ahogy azt feltételeztük, nem jutottunk szignifikáns különbséghez ($F(2,130) = 0.0007$, $p = 0.99 > 0.05$).

Következő lépésként elemeztük a résztvevők eredményeit az utóteszten szerzett pontszámok alapján is (**3. ábra**). Ennek vizsgálatához kovariancia-elemzést használtunk (ANCOVA). A független változónk ismét az oktatási feltétel volt, és a függő változó pedig az utóteszten elért pontszám (a homogenitás elve teljesült: $p = 0.27 > 0.05$). Mint kovariancia érték, az előteszten elért pontszámot vettük alapul. Bár egy mérsékelt csökkenés ($G_1$: 56%, $G_2$: 53%, $G_3$: 51%) figyelhető meg, ez nem vezetett lényegesen eltérő különbségekhez ($F(2,130) = 0.846$, $p = 0.432 > 0.005$).




.

**3. ábra**: Elő- és utóteszt pontszámok az interaktivitás függvényében

Ezek az eredmények nem igazolták az előzetes feltételezésünket, miszerint a tanulási eredmény növekedni fog, ha az interaktivitási szint is növekszik. Mindez arra enged következtetni, hogy általánosan meghatározható optimális interaktivitási szint nincs. Ez a kijelentés számos előzetes szakirodalmi kutatásnak ellentmond. Például Shaffer és munkatársai [18] az a következtetést vonták le ([15] és [9] alapján), hogy egyre több bizonyíték utal arra, hogy a legfontosabb tényező, amely hozzájárul az AV-vel való tanulás hatékonyságához a hallgatók figyelmének bevonásán alapszik.

Ennek ellenére, számos olyan kutatás is ismeretes, amelyek a mi következtetéseinkkel harmonizálnak. Például Jarc, Feldman és Heller [10] sem jutott szignifikáns különbségekhez a tanulási eredmények tekintetében, amikor összehasonlították az interaktív (interaktív jóslás) módban, illetve interaktivitás nélküli (passzív megtekintés) módban bemutatott AV-t. Erre egy ésszerű magyarázat lehet Shaffer és munkatársainak [18] következtetése, miszerint a hallgatók figyelmének felkeltése többféleképpen is elérhető. Ez arra utal, hogy az egyszerű megtekintése az AV-nak is lehet érdekes a diákok számára, akkor is, ha nem tartalmaz semmilyen jellegű interaktivitást. Továbbá, Myller és Laakso [14] is hasonló eredményekhez jutott. Ők is azt tapasztalták, hogy a megjelenítés két különböző formája: egyszerű megtekintés és változtatás módok nem vezettek jelentősen eltérő eredményekhez.

Hasonló eredményeket találtunk az animációra vonatkozó ütemezéssel kapcsolatosan. Az előzetes kutatások vegyesen vélekednek ennek hatékonyságáról. Néhányan úgy vélték, hogy van előnye az animáció irányításának, levezénylésének ([3], [13], [17]), míg mások nem tapasztalták ennek jelentős hasznát ([1]). Az egyik legfontosabb következtetést Berney és Bétrancourt ([2]) fogalmazta meg, akik szerint az animáció pozitív hatása a statikus elemekkel szemben csak akkor volt érzékelhető, ha a tanulók nem kellett irányítsák az animációs folyamatot, az AV ütemezését.

Továbbá, Jarc és munkatársai ([10]) az interaktív jóslás módszerének használata kapcsán azt tapasztalták, hogy olykor a diákok egy találgató játékként kezelik a megjelenő kérdéseket (melyik a következő művelet), és nem veszik komolyan a válaszadást. Ennek következtében a hallgatók aktív tanulásba való bevonása valójában nem történik meg.

Az eredményeink alapján arra tudunk következtetni, hogy mindhárom aktív tanulásra vonatkozó módnak vannak előnyei és hátrányai egyaránt. Az, hogy az interaktív feltétel nem vezetett jobb eredményekhez azzal is magyarázható, hogy az animációs folyamat megszakítása, feldarabolása ronthat az összképen, és ennek következtében a tanulók kevésbé tudnak elmélyülni az algoritmus vizualizációjában. Az, hogy kellőképpen megértsék az algoritmust, azt feltételezi, hogy az apró részletekből (algoritmus lépései) a végén tudjanak alkotni egy átfogó képet is. A vizualizáció töredezettsége megakadályozhatja a felhasználókat ennek a mentális képnek a kialakításában.

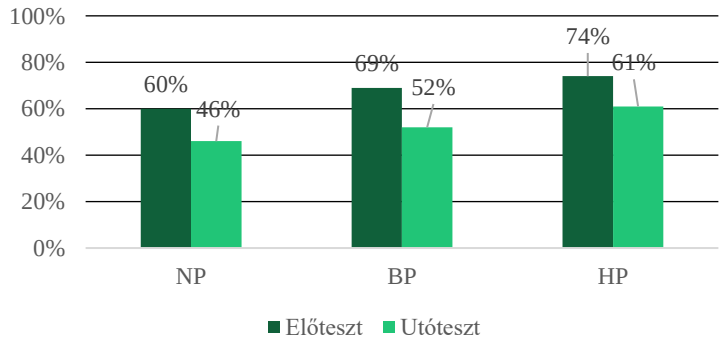
Fontos megemlíteni azt is, hogy ezeket az eredményeket a környezet használatához kapcsolódó nehézségek is befolyásolhatták. Bár kifejezetten összpontosítottunk arra, hogy a 15 perces bemutató során kellő rálátást nyújtsunk az online környezet funkcionalitásaira, az animáció levezénylésének módjára, ezzel minimálisra csökkentve a platform használatának nehézségét, elképzelhető, hogy ennek ellenére a teljes interaktivitás módban nehezebben boldogultak a hallgatók. Ezt természetesen figyelembe vettük, és a tanulási fázis során a $G_3$ csoport tagjainak több idő állt rendelkezésükre a tanfolyam elvégzésére.

## 6.1. Teljesítmény vizsgálata az előzetes programozási tapasztalat függvényében

A hallgatók teljesítményét az előzetes programozási tapasztalatok függvényében is elemeztük (**4. ábra**), úgy az előteszten, mint az utóteszten elért pontszám alapján (független változó: NP, BP vagy HP, függő változó: elő-/ utóteszten elért pontszám). Mindkét esetben, ahogy azt feltételeztük, az ANOVA elemzés szignifikáns különbségekhez vezetett $(előteszt: F(2,130) = 3.455, p = 0.034 < 0.05; utóteszt: F(2,130) = 11.299, p = 0.00 < 0.05)$. Amikor részletesebben megvizsgáltuk az eredményeket (páronkénti kontraszt-analízis) azt tapasztaltuk („<" jelentése: nincs szignifikáns növekedés, „<<" jelentése: marginálisan szignifikáns növekedés, „<<<" jelentése: szignifikáns növekedés), hogy:

- Előteszt: NP << BP < HP;
- Útóteszt: NP < BP <<< HP;

Érdekes módon, míg az NP és HP kategóriákba tartozó diákok eredményei közti különbségek majdnem teljesen megegyeztek mindkét esetben (előteszt: 60% vs. 74%, utóteszt: 46% vs. 61%), a BP kategóriába tartozó diákok előteszten elért eredménye a HP kategóriába tartozó kollegáik teljesítményéhez volt közel, az utóteszten elért pontszámuk az NP kollégák eredményét közelítette. Ennek egy lehetséges magyarázata az lehet, hogy az utóteszt kérdéseinek 66%-a ($K_{1\text{-}8}$) az algoritmus konkrét műveleteire, míg 33%-a ($K_{9\text{-}12}$) az algoritmus bonyolultságra vonatkozó kérdésekre épült, mely csak a HP kategóriába tartozó hallgatók középiskolás tanterve tartalmazott.

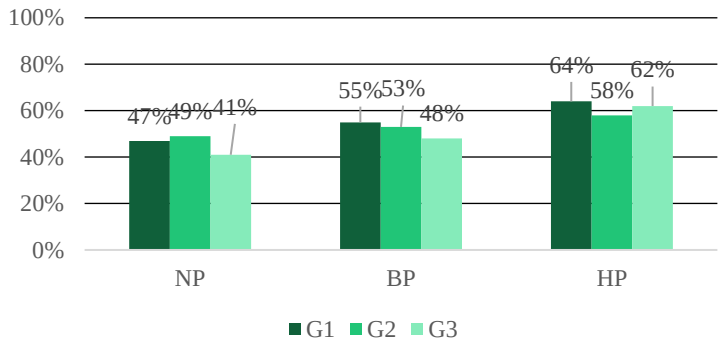



**4. ábra**: Elő- és utóteszten elért pontszámok az előzetes programozási tapasztalat függvényében

Arra is kíváncsiak voltunk, hogy vajon hogyan teljesítettek egymáshoz képest a különböző kategóriába (NP, BP, HP) tartozó hallgatók a különböző interaktivitással rendelkező csoportok ($G_1$, $G_2$, $G_3$) esetén (**5. ábra**). Ahhoz, hogy erre választ kapjunk, kétszempontos varianciaanalízist alkalmaztunk, ahol a két független változónk az oktatási feltétel (nincs-, fél-, teljes-interaktivitás) és az előzetes programozási tapasztalat (NP, BP, HP) volt. Függő változóként a résztvevők utóteszten elért pontszámát választottuk (A Levene-próba igazolta, hogy azonosak a varianciák: $p$ =1.54 > 0.05).

Nem jutottunk lényegesen eltérő különbségekhez ($p = 0.6 > 0.05$). Megvizsgáltuk az eredményeket három különböző ANOVA elemzéssel is minden kategóriára vonatkozóan (NP, BP, HP), de ez sem vezetett szignifikáns különbségekhez.

Ezek az eredmények összhangban vannak néhány előzetes szakirodalmi kutatás eredményeivel. Myller, Laakso és Korhonen ([14]) például hasonlóképpen két csoportba osztotta a kutatásban résztvevőket az előteszt eredményeik alapján: NPK (nincs előzetes tapasztalat az adott témát illetően) és SPK (van előzetes tapasztalat a témát illetően). Bár a szerzők megemlítik, hogy a felhasználók magasabb szinttű bevonása jobban kedvezett az NPK csoport diákjainak, végkövetkeztetésként ők is arra jutottak, hogy nem volt statisztikailag szignifikáns eltérés a két csoport eredményei között.



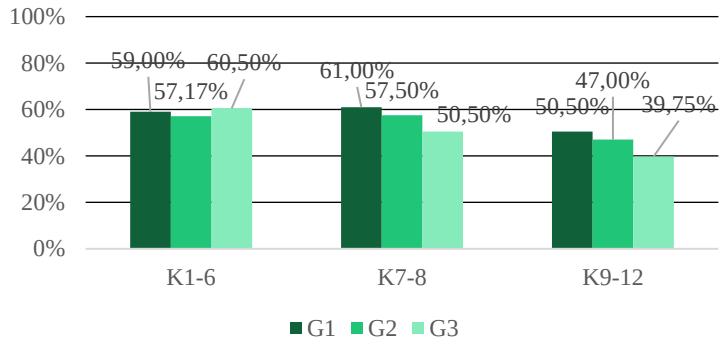

**5. ábra**: Utóteszten elért pontszámok az interaktivitás és az előzetes programozási tapasztalat függvényében

## 6.2. Az interaktivitási szint és az elsajátított tudás jellege közti összefüggés

Az utóteszt kérdéssora összeállításakor három különböző kérdéstípust határoztunk meg. Az első 6 kérdés az algoritmus egymásután következő műveleteire vonatkozott (első három művelet). Azt feltételeztük, hogy ezek a kérdések leginkább a teljes-interaktivitással tanuló csoportnak fognak a legjobban sikerülni, hiszen nekik fel kellett építeniük az algoritmust lépésről lépésre (ki kellett választaniuk a helyes elempárt, majd az ezekre vonatkozó helyes műveletet, majd újra és újra). A 7-8 kérdések véletlenszerűen kiválasztott lépésekre vonatkoztak, mely a lépéssorozat valamely tagját képezték. Ezek a kérdések leginkább a fél-interaktivitás móddal harmonizáltak (interaktív jóslás). Az utolsó négy kérdés az algoritmus bonyolultságra vonatkozó kérdésekre épült, vagyis arra, hogy milyen mértékben sikerült a résztvevőknek megérteni az algoritmus startégiáját és kialakítani erről egy összefüggő mentális képet. Úgy véltük, ezt leginkább az nincs-interaktivitás mód teszi lehetővé.

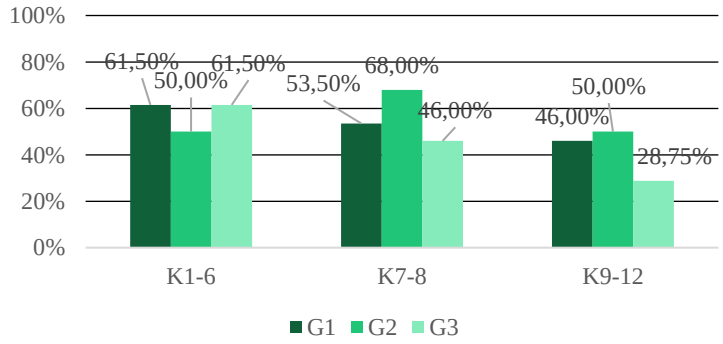
A hallgatók összteljesítménye csökkent, ahogy egyre komplexebb kérdések kerültek megválaszolásra ($K_{1\text{-}6}$: 59%, $K_{7\text{-}8}$: 56%, $K_{9\text{-}12}$: 45%). Míg a $K1_{\text{-}6}$ és $K_{7\text{-}8}$ kérdések eredményei közel voltak egymáshoz, a $K_{9\text{-}12}$ kérdéseken elért pontszám jelentősen alacsonyabb volt ezeknél. A Bloom rendszertant [12] alapul véve azt mondhatjuk, hogy ez egy várható eredmény, mivel az első 8 kérdés az alkalmazási szintre vonatkozott, míg az utolsó 4 az elemzési szintre.

Amikor külön megvizsgáltuk a csoportokat az egyetlen (marginálisan) szignifikáns eredmény a $G_3$ csoportra vonatkozó $K_{9\text{-}12}$ kérdések esetén született, az ennek megfelelő $G_1$ és $G_2$ csoportok eredményeivel szemben (ANOVA, kontraszt értékek: (-1, -1, 2); $p = 0.058$) (**6. ábra**). Egy lehetséges magyarázat arra, hogy a $G_3$ csoport diákjai miért teljesítettek gyengébben a $K_{9\text{-}12}$ kérdések esetén az lehet, hogy a teljes-interaktivitás a vizualizáció szaggatottságát eredményezte, és így a hallgatók nehezebben tudtak kialakítani egy átfogó képet az algoritmusról.




**6. ábra**: Eredmények az interaktivitás és a kérdéstípus függvényében

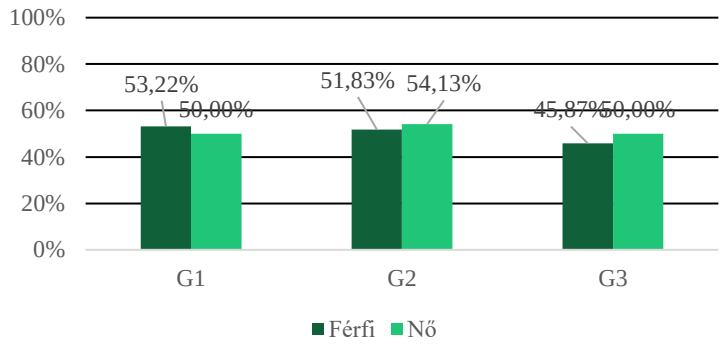
Következő lépésként elemeztük az eredményeket minden kategória (NP, BP, HP) esetén különkülön. Az egyetlen szignifikáns különbség a BP hallgatók esetén volt látható (**7. ábra**). Amikor külön megvizsgáltuk a három csoportot ($G_1$, $G_2$, $G_3$), érdekes következtetésekhez jutottunk. Annak ellenére, hogy a $G_1$ és $G_3$ csoport teljesítményénél egy lineáris csökkenés figyelhető meg, az eltérés változó ($G_1$: 61.5%, 53.5%, 46%; $G_3$: 61.5%, 46%, 28.8%). További érdekességnek mondható az, hogy a $G_2$ csoport (fél-interaktivitás) hallgatói legjobban a $K_{7\text{-}8}$ kérdéseken teljesítettek, ahol ők érték el a legnagyobb pontszámot. Mindez, arra enged következtetni, hogy van összefüggés az interaktivitás szintje (amellyel tanulmányozták az algoritmust) és az elsajátított tudás jellege közt.




**7. ábra:** Eredmények azinteraktivitás és kérdéstípus függvényében - BP

## 6.3. Teljesítmény vizsgálata nemek szerint

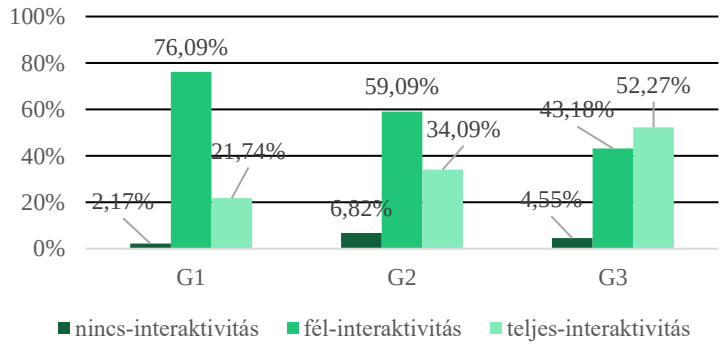
Mivel a leghatékonyabb módszer eltérő lehet nők és férfiak esetén úgy gondoltuk, hogy elemezzük az utóteszten elért pontszámokat a nemek szerint is. Amikor az összteljesítmény szerint hasonlítottuk össze az eredményeket, nem jutottunk lényegesen eltérő különbségekhez. Ezt követően megvizsgáltuk az eredményeket a különböző csoportok teljesítménye alapján is (**8. ábra**). Ennek elemzésére kétszempontos varianciaanalízist alkalmaztunk (ANOVA), ahol a két független változó az oktatási feltétel (nincs-, fél-, teljes-interaktivitás) és a nem (férfi, nő) volt, valamint függő változóként az utóteszten elért pontszám szolgált (a Levene-próba igazolta, hogy azonosak a varianciák: $p = 0.8 > 0.05$). Azt tapasztaltuk, hogy míg a férfiak jobban teljesítettek a nincs-interaktivitás módban, ez az eredmény a másik két esetben (fél- és teljes-interaktivitás) megfordult.




**8. ábra**: Eredmények nemek szerint

## 6.4. Legkedveltebb tanfolyamtípus

Az utóteszt befejezését követően minden hallgatónak válaszolnia kellett arra a kérdésre, hogy melyik tanfolyammal (interaktivitási szinttel) tanulna szívesebben, függetlenül attól, hogy melyik csoporthoz tartozott. Amint az az ábrán is látható (**9. ábra**), a $G_1$ csoport hallgatóinak 76.09%-a szeretett volna egy szinttel magasabb ($G_2$) és 21.74% két szinttel magasabb ($G_3$) interaktivitást. Ehhez hasonlóan a $G_2$ csoport tagjainak 34.09%-a vágyott egy szinttel magasabb ($G_3$) és 59.09%-a meg volt elégedve a hozzájuk rendelt ($G_2$) interaktivitás szintjével, vagyis a fél-interaktivitással. A $G_3$ csoport esetén a diákok nagyrésze (52.27%) arra szavazott, hogy meg vannak elégedve ehhez a csoporthoz tartozó interaktivitással (teljes-interaktivitás).



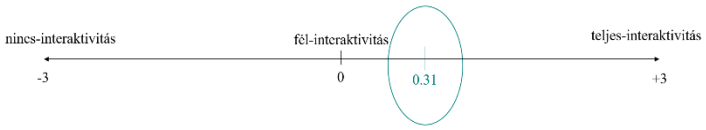

**9. ábra**: Diákok elégedettségi szintjének ábrája

Ahhoz, hogy egy kicsit részletesebben is szemügyre vegyük az eredményeket, a résztvevők válaszait a következőképpen kódoltuk (**2. táblázat** és **10. ábra**):

- 0: azon résztvevők száma, akik meg voltak elégedve a számukra kijelölt tanfolyam típusával
- -1 vagy -2: azok száma, akik kevesebb interaktivitást szerettek volna (egy- vagy két szinttel kevesebbet)
- +1 vagy +2: azok száma, akik több interaktivitást szerettek volna (egy vagy két szinttel többet)

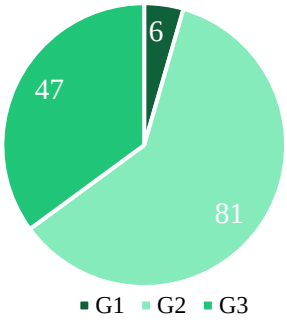
| Előre meghatározott tanfolyam | -2 | -1 | 0 | +1 | +2 | Átlag |
|---|---|---|---|---|---|---|
| $G_1$ | - | - | 1($G_1$) | 35($G_2$) | 10($G_3$) | 1.19 |
| $G_2$ | - | 3($G_1$) | 26($G_2$) | 15($G_3$) | - | 0.27 |
| $G_3$ | 2($G_1$) | 19($G_2$) | 23($G_3$) | - | - | -0.52 |
| Összesített átlag | | | | | | 0.31 |

**2. táblázat:** Diákok elégedettségi szintje




**10. ábra**: Diákok megelégedési szintjének számtengelye

A válaszok kódolását és feldolgozását követően egy 0.31 átlagértékhez jutottunk. Ez a pozitív érték arra utal, hogy a felhasználók átlagosan a magasabb interaktivitási szintre szavaznak. Amint az az ábrán is látható, ez az érték a fél-interaktivitás módhoz áll a legközelebb (**11. ábra**). Ez tulajdonképpen egy bíztató eredmény hiszen számos előzetes szakirodalmi kutatás foglal álláspontot emellett az aktív tanulási forma mellett. Egy Byrne, Catrambone és Stasko ([4]) által végzett kutatásban a videó volt felhasználva, mint a tanulási fázis során alkalmazott eszköz. A videó bizonyos kulcsmomentumokban megállt, és a diákoknak meg kellett válaszolniuk, hogy mi lesz a következő lépés. Hasonlóképpen, Grissom, McNally és Naps ([6]) is azt fogalmazta meg, hogy azon diákok érték el a legnagyobb fejlődést, akiknek kérdésekre kellett válaszolniuk az AV során.




**11. ábra**: Legkedveltebb tanfolyamtípus

Naps és munkatársai ([15]) említenek egy olyan jelenséget egyik kutatásukban, hogy bár az oktatók teljes mértékben hisznek abban, hogy a vizualizációk segíthetnek, ez a kijelentés nincs összehangban a kutatás eredményeivel. Érdekes módon, mi is egy hasonló következtetéshez jutottunk. A diákok több interaktivitásra vágynak, mégis az eredményeik nem mutatják azt, hogy ez jelentősen hozzájárul a tanulási eredményeik növekedéséhez.

## 7. Limitációk

A kutatás egyik limitáció az, hogy a felmérés csak egy algoritmusra alapszik: az AlgoRythmics környezetben megtalálható Shell rendezés animációjára.

Egy másik limitáció lehet az, hogy a résztvevőket mi soroltuk három különböző csoportba. Egy következő lépés lehetne az, hogy ők választhassák ki szabadon, hogy melyik formáját szeretnék kipróbálni a szemléltetésnek.

Továbbá, az is elképzelhető, hogy az eredményeket befolyásolta az adott tanfolyam nehézsége és az, hogy hogyan kellett használni a platform nyújtotta funkciókat. Mint jövőbeni terv, bevezethetünk egy extra utótesztet, amely ezeket a befolyásoló tényezőket és ezek hatását mérné fel.

## 8. Következtetések

Kutatásunkkal szerettünk volna egy új betekintést nyújtani az AV-ra vonatkozó kutatási területbe és értékes útmutatásokat szerettünk volna megfogalmazni, melyek elősegíthetik az oktatásszervezést. A Naps ([15]) által meghatározott aktív tanulás formáira alapozva, jelen tanulmányba három interaktivitási szintet különböztettünk meg: megtekintés, válaszadás és levezénylés. Ezeket a módszereket az AlgoRythmics online oktatási környezetbe is beépítettük, mely kapcsán a Shell rendezés bemutatását szolgáló animációkat vettük alapul. Az említett három tanulási formához három oktatási fogalmat társítottunk: nincs-interaktivitás (passzív megtekintés), fél-interaktivitás (interaktív jóslás) és teljes-interaktivitás (algoritmus „levezénylés"). Az egyik legfontosabb következtetésünk az, hogy általánosan kijelenthető optimális interaktivitási szint nem határozható meg.

Bár az eredmények alapján nem tudjuk kijelenteni az előzetes feltételezésünket (az AlgoRythmics környezet Shell rendezésre vonatkozóan), miszerint az interaktivitás növekedésével a tanulási eredmény is növekedni fog, azt kijelenthetjük, hogy mindhárom interaktivitási szintnek megvannak az előnyei és hátrányai egyaránt. Ahogy azt Urquiza-Fuentes és Velázquez-Iturbide ([20]) is kijelentette, számos szakirodalmi cikk elemzése után az AV hatékonyságára vonatkozóan, az ismeretszerzés fejlesztése az interaktivitás bármely szintjén elérhető, úgy mi is úgy gondoljuk, hogy ezek mindegyike egyaránt hozzájárulhat az algoritmus megértéséhez.

Bebizonyosodott, hogy a résztvevők függetlenül attól, hogy mennyi év előzetes programozási tapasztalatot tudhatnak maguk mögött, egyaránt érvényesülni tudtak a tanulási formáknak köszönhetően. Továbbá, kijelenthető az is, hogy az interaktivitás és az elsajátított tudás jellege között van összefüggés.

Mindenki megérdemli, hogy tanulhasson, és mivel nem vagyunk egyformák, a leghatékonyabb tanulási stílus eltérő lehet. Éppen ezért kijelenthetjük, hogy az online oktatási környezeteknek tartalmazniuk kell az aktív tanulás minden formáját, és lehetőséget kell biztosítsanak arra, hogy az AV-t különböző interaktivitási szintekkel lehessen tanulmányozni. Ha mindezt lehetővé teszik, minden felhasználó meg tudja találni a számára leghatékonyabb tanulási stílust, és ez minden bizonnyal az eredmények javításához vezet majd.

## Irodalom


[1] Adesope, O. O., & Nesbit, J. C. (2012). Verbal redundancy in multimedia learning environments: A meta-analysis. Journal of Educational Psychology, 104(1), 250.

[2] Berney, S., & Bétrancourt, M. (2016). Does animation enhance learning? A meta-analysis. Computers & Education, 101, 150-167.

[3] Boucheix, J. M., & Guignard, H. (2005). What animated illustrations conditions can improve technical document comprehension in young students? Format, signaling and control of the presentation. European Journal of Psychology of Education, 20(4), 369-388.

[4] Byrne, M. D., Catrambone, R., & Stasko, J. T. (1999). Evaluating animations as student aids in learning computer algorithms. Computers & education, 33(4), 253-278

[5] Denning, P. J. (2017). Remaining trouble spots with computational thinking. Communications of the ACM, 60(6), 33-39.

[6] Grissom, S., McNally, M. F., & Naps, T. (2003, June). Algorithm visualization in CS education: comparing levels of student engagement. In Proceedings of the 2003 ACM symposium on Software visualization (pp. 87-94).

[7] https://algorythmics.ms.sapientia.ro/

[8] https://www.bebras.org/

[9] Hundhausen, C. D., Douglas, S. A., & Stasko, J. T. (2002). A metastudy of algorithm visualization effectiveness. Journal of Visual Languages & Computing, 13(3), 259-290.

[10] Jarc, D. J., Feldman, M. B., & Heller, R. S. (2000). Assessing the benefits of interactive prediction using web-based algorithm animation courseware. ACM SIGCSE Bulletin, 32(1), 377-381.

[11] Karavirta, V., & Shaffer, C. A. (2015). Creating engaging online learning material with the JSAV javascript algorithm visualization library. IEEE Transactions on Learning Technologies, 9(2), 171- 183.

[12] Krathwohl, D. R. (2002). A revision of Bloom's taxonomy: An overview. Theory into practice, 41(4), 212-218.

[13] Mayer, R. E., & Chandler, P. (2001). When learning is just a click away: Does simple user interaction foster deeper understanding of multimedia messages?. Journal of educational psychology, 93(2), 390.

[14] Myller, N., Laakso, M., & Korhonen, A. (2007, June). Analyzing engagement taxonomy in collaborative algorithm visualization. In Proceedings of the 12th annual SIGCSE conference on Innovation and technology in computer science education (pp. 251-255).

[15] Naps, T. L., Rößling, G., Almstrum, V., Dann, W., Fleischer, R., Hundhausen, C., ... & Velázquez-Iturbide, J. Á. (2002). Exploring the role of visualization and engagement in computer science education. In Working group reports from ITiCSE on Innovation and technology in computer science education (pp. 131-152).

[16] Plass, J. L., Homer, B. D., & Hayward, E. O. (2009). Design factors for educationally effective animations and simulations. Journal of Computing in Higher Education, 21(1), 31-61

[17] Schwan, S., & Riempp, R. (2004). The cognitive benefits of interactive videos: learning to tie nautical knots. Learning and instruction, 14(3), 293-305.

[18] Shaffer, C. A., Cooper, M. L., Alon, A. J. D., Akbar, M., Stewart, M., Ponce, S., & Edwards, S. H. (2010). Algorithm visualization: The state of the field. ACM Transactions on Computing Education (TOCE), 10(3), 1-22.

[19] Turing, A. M. (2009). Computing machinery and intelligence. In Parsing the Turing Test (pp. 23-65). Springer, Dordrecht

[20] Urquiza-Fuentes, J., & Velázquez-Iturbide, J. Á. (2009). A survey of successful evaluations of program visualization and algorithm animation systems. ACM Transactions on Computing Education (TOCE), 9(2), 1-21.

[21] Wing, J. M. (2006). Computational thinking. Communications of the ACM, 49(3), 33-35.

# Informatikai gondolkodás fejlesztésének dimenziói

Pluhár Zsuzsa

pluharzs@inf.elte.hu
Eötvös Loránd Tudományegyetem, Informatikai Kar

**Absztrakt.** Az informatikai gondolkodás fejlesztése nem csupán számítógéppel, programozással lehetséges. Több nemzetközi és hazai törekvés, irányzat létezik, melyek a számítógép nélküli megoldásokat kutatják és támogatják. Az egyik ilyen a nemzetközi bebras kezdeményezés, mely immár tíz éve van jelen Magyarországon is. Ezidő alatt nem csak az alapokat jelentő verseny megszervezésére került sor minden évben – 2019-ben már több, mint 27ezer diák részvételével –, de a háttérben az algoritmikus gondolkodás mérésére és fejlesztésére használt tesztek és kurzusok kidolgozására, kiegészítő aktivitások kialakítására is. Jelen tanulmány az e-hód kezdeményezés tíz éve alatt folytatott kutatásokat, illetve ezek egymásra épülését foglalja össze.

**Kulcsszavak:** informatikai gondolkodás, algoritmikus gondolkodás, számítógép nélkül (cs unplugged), bebras, hód

## 1. Informatikai gondolkodás vizsgálatának dimenziói

Az informatikai gondolkodás (computational thinking) fogalma Wing [1] megfogalmazása óta sokat alakult, formalizálódott.

A fejlesztéséről, méréséről folyó kutatások alapja minden esetben annak megfogalmazása, mely dimenzióit emeljük ki, illetve az egyes dimenzióit hogyan tudjuk szétválasztani. A leginkább elterjedt elmélet [2, 3] három dimenzióra osztja az informatikai gondolkodást: (1) az elméleti fogalmak – program struktúrák és használatuk – ismerete (CT concepts), (2) a gyakorlati ismeretek (practicies) mint a problémamegoldás, a probléma megfogalmazása, a dekompozíció, az absztrakció, a tesztelés és hibakeresés folyamatai, és a (3) perspektívák (perspectives), a körülöttünk lévő világ megismerése és megértése, kritikus gondolkodás.

A felhasznált eszközök tekintetében három nagyobb vizsgálati csoportot különíthetünk el:

Az első esetében a programozás, a programozási folyamatok kerülnek hangsúlyozásra – a programkészítés folyamatán, a programozási lépéseken, használt program struktúrák megismerésén, a tervezéstől a tesztelésig végzett tevékenységeken keresztül kerül sor az absztrakció, a dekompozíció, az algoritmikus gondolkodás fejlesztésére. Ebben a csoportban a harmadik (3) dimenzió általában kisebb hangsúlyt kap, van, hogy teljesen kimarad. [4-6]

A második csoport projektfeladatok, projektmunkák során beintegrálja az informatikai gondolkodás elemeit. Ebben az esetben a harmadik dimenzió (3) kap nagyobb szerepet és a programozási elemek, elméleti fogalmak (1) sokszor nem is jelennek meg, vagy kisebb hangsúllyal szerepelnek. A kutatások többsége a STEAM (science, technology, engineering, art and math, azaz természettudományok, technológia, mérnöktudományok, művészet és matematika) területeken fogalmazza meg a projekteket, aktivitásokat. [7-9]

A harmadik csoportba sorolhatjuk azokat a kezdeményezéseket, melyek számítógép nélküli aktivitásokkal valósítják meg az informatikai gondolkodás fejlesztését, vizsgálatát. Ilyen kezdeményezések pl. a legkisebbeknek szóló cs unplugged [10], az idősebbeket megszólító cs4fun [11] vagy a legszélesebb életkori és dimenziókat összefoglaló bebras [12, 13].

## 2. e-hód

A bebras kezdeményezés céljait [12, 13] kiterjesztve, illetve testreszabva fogalmaztuk meg a magyarországi megvalósulás, az e-hód célkitűzéseit [15, 16].

A feladatok, aktivitások témájában és megfogalmazásában fontos, hogy:

- felkeltse az érdeklődést az informatika iránt;
- feloldja az informatikával kapcsolatos félelmeket, negatív érzéseket;
- megmutassa az informatika területének sokszínűségét, felhasználási lehetőségeit és területeit.

Mindezt úgy, hogy informatikai előképzettséget nem igényel.

Célcsoportként nem csupán a diákokat nevezhetjük meg, de ugyanilyen fontosak számunka a közoktatásban résztvevő és a hamarosan bekerülő tanárok is. Az ő számukra célunk támogatást adni az egyes informatikai témakörök motiváló probléma-felvetéseihez, példa alapú megoldásaihoz, valamint az informatika sokszínűségének és integrálásának lehetőségeihez.

A kezdeményezés keretein belül létrehozott anyagok többségére a CC BY-NC-SA 4.0 licensz vonatkozik, így a közoktatásban résztvevők könnyen felhasználhatják azokat.

### 2.1. Verseny

A kezdeményezés alapjait egy verseny jelenti, melyben 18 rövid, gyorsan (kb. 3 perc alatt) megoldható, informatikai előképzettséget nem igénylő feladatot kell megoldaniuk a résztvevőknek.

A feladatok három nehézségi szinten (nehéz, közepes és könnyű) megfogalmazott, érdekes problémákat mutatnak be. Nem tesztek, inkább szórakoztató gondolkodtató feladványok, melyekkel a résztvevők új ismeretekre tehetnek szert, illetve meglévő ismereteiket mélyíthetik el.

A feladatok előkészítését, pontosítását a nemzetközi csapat egy műhelykonferencia keretein belül végzi, kiválogatva és módosítva a résztvevő országokból beküldött kérdéseket. Ezután az egyes országok honosítják a kérdéseket, és a náluk megrendezett versenyhez testre szabják azokat.

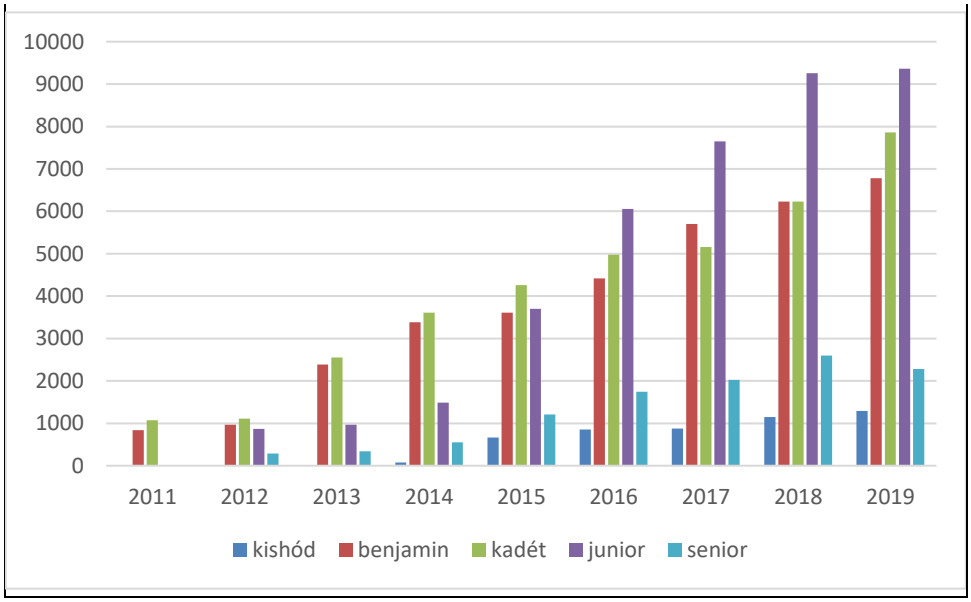
2010 óta Magyarország is – az ELTE IK vezetésével és tanárszakos hallgatók bevonásával – részt vállal ebben az előkészítő folyamatban úgy, hogy 2011 óta megszervezzük a versenyt is. 2014 óta pedig mint National Bebras Organiser teljes jogú tagjai lettünk az International Bebras Committee-nek.

Az első alkalommal 2 életkori kategóriában (benjamin, kadét) 1907 diák versenyzett 24 iskolából. 2019-re 202 iskola több, mint 27ezer diákja (27 570) öt életkori kategóriában mérette meg magát.

| Korcsoport kategória | Osztály |
|---|---|
| kishód | 4. |
| benjamin | 5–6. |
| kadét (első évben meteor) | 7–8. |
| junior | 9–10. |
| senior | 11–12. (13.) |

**1. táblázat:** Életkori kategóriák a hód versenyeken.

Az egyes életkori kategóriákban (1. táblázat) résztvevők számának alakulásából (1. ábra) leolvasható az is, hogy a fiatalabbak átlépnek az idősebb korosztályos csoportokba, viszik magukkal a versenyt.




**1. ábra:** Résztvevőszámok alakulása életkori kategóriánként a magyarországi hód versenyeken.

A verseny országos lefedettsége egyre erőteljesebb.

A versenyek eredményességéről, célkitűzéseink megvalósulásáról több fronton igyekszünk megbizonyosodni. Folyamatosan fogadjuk a résztvevők visszajelzéseit. Nem csak kötetlen formában, de minden verseny után kérdőívet is készítünk, melyet a koordinálást segítő tanárok önkéntesen töltenek ki és küldik el számunkra nem csak a saját, de diákjaik észrevételeit, tapasztalatait is.

Időről időre kérdőíves kitöltésre kérjük fel a diákokat is, melynek segítségével igyekszünk vizsgálni, hogy a sikeres részvételüket milyen dimenziók, paraméterek segítik, illetve a részvételük a versenyen milyen egyéb területen fejleszti őket, ad pluszt számukra.

Minden évben, a teljesítmények tükrében megvizsgáljuk a kérdések erősségét, besorolásuk helyességét is. Vizsgáljuk azokat a helytelen válaszlehetőségeket, melyeket kiemelkedően sok diák választ ki egy-egy feladatnál.

Összességében – korábban is publikált eredményeinket tekintve [15, 16] – elmondhatjuk, hogy a kérdések nehézségi szintje megfelelő, illetve általánosan jó nehézségi szint besorolást kapnak a verseny során.

A nemek szerinti bontásban sem az egész minta vonatkozásában, sem a korcsoportonkénti bontásban nem látható szignifikáns különbség a fiúk és lányok teljesítményében egyik évben sem, valamint a versenyen való sikeres szereplést leginkább befolyásoló tényező a korábbi részvételek száma volt.

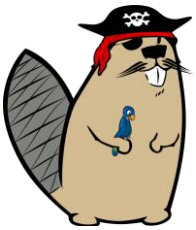
A résztvevőszámok alakulásából következtetve, illetve a kötetlen és kérdőíves visszajelzésekből megerősítést kaptunk, hogy a motivációs céljaink – érdeklődés felkeltése, sokszínűség megmutatása – valóban megfelelően alakul mind a diákok, mind a tanárok esetében.

## 2.2. Feladat kiterjesztések

A versenyen használt feladatok többsége megállja a helyét egy interaktív vagy számítógép nélküli aktivitás során is. A motivációs, fejlesztő és ismeretgazdagító célokra ráerősíthet a játékosítás.

A kiterjesztett aktivitások tervezésénél fontos szempont, hogy több korosztály is sikerélménnyel gazdagodhasson, akár családi tevékenységként is végezhetőek legyenek. További célom volt, hogy mintát mutassak az egyes feladatok felhasználhatóságára – akár a tanórákon, akár azokon kívüli és motiváljam a közoktatásban dolgozókat a hód célok között megfogalmazott további felhasználásra [15].

Két aktivitást emelnék ki a többi közül. Mindkettőben a kiválasztott feladatok játékosítása fizikai megvalósítással történt: azaz a feladatok megoldásához valamilyen fizikai tevékenységre van szükség. Súlyok pakolása, kódtárcsa tekerése, hamburger, puzzle összerakása, kártyák tologatása, élőtársas, ... mind előfordul a megvalósítások között. Az előkészített feladatok nyomtatott, laminált papírokkal, gyurmával, horgolt eszközökkel – azaz bárki által elkészíthető alapokkal, olcsón beszerezhető eszközökkel készülnek, készültek el.



Az első aktivitás egy kalóz kincsesládájának a kinyitása, ahol négy feladat megoldása során összegyűjtött kóddal egy kincsesláda nyitható. A négy feladat összeválogatásánál figyeltem arra, hogy mind könnyebb, mind nehezebb feladatok szerepeljenek – de a megoldáshoz akár kisebb segítségekkel pár perc alatt mindenki el tudjon jutni. A megoldás során a diákok, családok akár párhuzamosan is megválaszolhatják a kérdéseket, illetve fontos momentum, hogy akár meg is beszélhetik azokat. Koordinátornak egy ember is elegendő – aki az éppen megakadó résztvevőknek segít, illetve az egyes feladatok eszközeit alapállapotba visszaállítja.

Ezt az aktivitást teszteltem a Jövő utcájában családos programként, az ELTE IK nyílt napjain az érdeklődő végzős középiskolások körében, Kutatók Éjszakáján is.

A másik projekt egy akadályverseny, ahol az egyes állomásokon kell hód feladatokat megoldani, s így minél több pecsétet szerezve eljutni a célig. Az állomások kidolgozásában az ELTE IK tanárszakos hallgatói vettek, vesznek rész – folyamatosan bővítve a „bevethető" feladatok tárházát. A legközelebbi akadályversenyt a 10. hód verseny díjkiosztójára tervezzük megvalósítani.

## 2.3. Programozók tesztelése

Az ELTE Informatikai Karának első éves alapozó kurzusa a Programozás, illetve az angol nyelvű Computer Science BSc képzésében a Programming elnevezésű kurzus, mely programozásmódszertani bevezetésként szolgál, célja az alapvető problémamegoldó algoritmusok megismerése és használata különböző egyszerű és összetettebb adatszerkezetekkel. Olyan általános sémákat igyekszik a hallgatók számára elérhetővé tenni, amely segítheti őket a problémák megértésében, a problémák részekre bontásában, az algoritmusok megalkotásában, majd végül a programok elkészítésében.

A korábbi évek tapasztalata alapján a hallgatók – s kiemelten az angol nyelvű BSc képzésben résztvevő hallgatók egy részének már az egyszerűbb algoritmusok megértése, felhasználása, majd később nehezebb feladatokban történő alkalmazása is problémákat okoz. A bonyolultabb algoritmusok, komplexebb problémák pedig óriási kihívást jelentenek számukra.

Az egyetemünkre jelentkező és angol nyelvű BSc képzésre felvételt nyert hallgatóknak egy matematikai és egy angol nyelvi tesztet kell megírniuk, és akik ezt a tesztet nem teljesítik megfelelő szinten, azok egy úgynevezett előkészítő féléven kezdhetik meg tanulmányaikat. A Programming kurzus tapasztalatai alapján 2019 szeptembere óta már egy algoritmikus gondolkodás teszttel is

bővült az év eleji felmérés, és 2020 szeptemberétől a magyar nyelvű hallgatók csoportbeosztását is egy hasonló teszttel segítjük.

A tesztek kidolgozásánál cél volt az, hogy az algoritmikus gondolkodás különböző szintjeit mérje, és ne támaszkodjon programozói előismeretekre. A differenciáló erő viszont megmutassa azt a határt, ami alatt a képzésbe lépőknek fejlesztési támogatást nyújthatunk egy előkészítő kurzuson.

Ehhez a teszt első verziójához [17] háttérkérdőívet is készítettünk, melynek önbevalláson alapuló eredményeit összevetettük a Programming képzés során nyújtott teljesítményekkel.

A tesztben szereplő kérdések a kiterjesztett Bloom-taxonómia alkalmazás, analízis, szintézis és alkotás szintjeit vizsgálták egy-egy könnyű, két-két közepes és egy-egy nehéz feladattal. A felidézési szint a kezdeti célok miatt – programozói előismeretek nélkül is megoldható –, és mivel a programozó képzésnek nem előfeltétele valamilyen programozási környezet vagy programozási nyelv ismerete, nem került tesztelésre.

Az eredmények előzetes feldolgozása alapján kijelenthetjük, hogy a teszt felülről megfelelő differenciáló erővel bír – azok kerültek az angol nyelvű képzésben előkészítő félévre, akiknek szükségük volt az erősítésre. Folyamatban van annak elemzése, hogy a teszten mutatott teljesítmény és a Programming, illetve Programozás kurzus elvégzésének nehézsége között fennáll-e valamilyen összefüggés. Ehhez háttérkérdőívvel kiegészítve a kurzusokon nyújtott teljesítményt vizsgáljuk. Az eredmények tekintetében fogalmazhatjuk meg majd azt, hogy a teszten első évre bejutó hallgatóknak nem lett volna szüksége a bevezető támogatásra, azaz alulról is megfelelő a teszt differenciáló ereje.

## 2.4. Egyetemi fejlesztő kurzus

2017-ben kezdtem el kidolgozni az első éves BSc programozó, angol nyelvű képzésbe bekerülők felzárkóztató kurzusát [18], mely az algoritmikus gondolkodás fejlesztésén kívül a csoportmunka, kooperáció, a kommunikáció és problémamegoldás területén is igyekezett erősíteni a résztvevőket.

A kurzus célját a következőképpen fogalmaztam meg a résztvevők számára:

- az algoritmikus gondolkodás és a problémamegoldási alapok biztosítása;
- annak megmutatása, hogy a problémamegoldás területén milyen szerepet játszhat az informatika (computer science);
- segíteni egyszerűbb célok megvalósítására készülő kisebb alkalmazások, programok megtervezésében és megvalósításában (illetve ezen folyamatok elsajátításában).

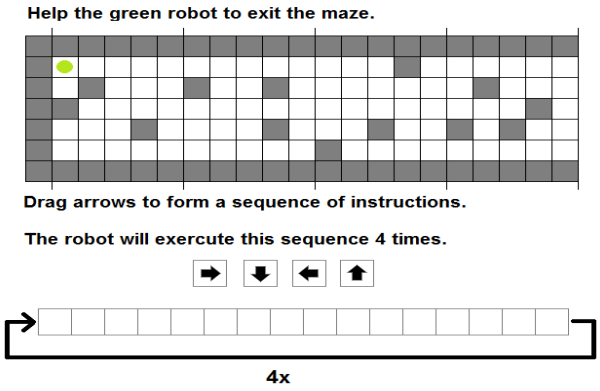
A kurzusban is helyet kapott a bebras kezdeményezés. A kurzus első moduljában („informatikai problémák és megoldásaik") megfogalmazott céljaink:

- képet adni az informatikáról, számítástudományról;
- támogatni a hallgatókat abban, hogy elkezdjenek informatikai problémákról gondolkodni;
- helyet, időt biztosítani a problémamegoldásról, stratégiákról való beszélgetésekhez és a problémák, megoldási lépések megfogalmazásához.
- motiváltá tenni a hallgatókat a problémamegoldásra és az egyes megoldási stratégiáról való gondolkodásra.

Ehhez egy rövid problémamegoldással kapcsolatos bevezetőt követően a diákokkal hód feladatokat oldunk meg és értelmezünk. A hangsúly nem csupán a megoldáshoz való eljutáson van – illetve annak kielemzésén, hogy esetlegesen több út is elvezethet a megoldáshoz. A csoportmunka és megbeszélések a fent összefoglalt célokat is támogatják.

A kurzus második és harmadik moduljában az utasítások pontosságának fontosságát, a program struktúrákat sajátítják el a hallgatók még mindig programozási környezet, konkrét programnyelv

használata nélkül. Itt többek között olyan hód feladatokat válogatunk össze, melyek ebben a témakörben adnak kihívásokat. Például a több nehézségi szinten megoldandó labirintus feladat (2. ábra), melyben a hallgatóknak a minta megmutatásával segítséget nyújthatunk a megoldásban anélkül, hogy elárulnánk a helyes választ.




**2. ábra:** Labirintus feladat – közepes nehézségű szintje.

Az idei évben végez a második komolyabb csoport a kurzuson, így a következő szemeszter lezárásával már kiértékelhetjük a kurzus hatását a hallgatók programozói képzésben nyújtott teljesítményének függvényében.

Az első végzett csoport eredményeinek tekintetében megállapíthatjuk, hogy a kurzus során szerzett tudás beépült. A hallgatók többsége (91%) sikeresen vette az első féléves kurzusokat. Megfigyelhető volt, hogy nem csak biztosabb gondolkodási struktúrákkal rendelkeztek, de a tanuláshoz való hozzáállásuk (határidők betartása, előadások követése, felkészülés az órákra, csalások elkerülése, előadásokon való aktív részvétel) is pozitív változásokat mutatott, és képesek voltak kérdéseik pontos megfogalmazására is. Annak vizsgálata még hátra van, hogy mindezen attribútumokat mennyire és milyen szinten az adott kurzus, és mennyire a többi előkészítő kurzus (nyelvi, matematikai stb.), illetve az előkészítő félév látogatott kurzusainak összessége befolyásolta. A kurzus hatása mindenképpen érezhető az elsőéves tárgyak teljesítési eredményeinek tükrében.

## 2.5. Tanárképzés

Az ELTE IK tanárképzése nem csak az eszközhasználat és az általános pedagógiai módszertanok elsajátítását kívánja a hallgatóitól. Képzésünk fontos része az informatikai eszközökkel támogatott módszertani megoldások megismertetésén felül azok gyakorlati hasznosításának és megvalósulásának bemutatása. Ennek részeként több kurzusban is teret kapott az e-hód kezdeményezés megismerése, illetve mind a verseny feladatok elkészítésében, mind a kiegészítő aktivitásokban való részvétel.

# 3. Konklúzió

A bebras kezdeményezés megismerésével, Magyarországra hozásával és kiterjesztésével töltött 10 évem komoly elgondolások és kutatások alapját jelentik. A magyar megvalósulás immár nem csupán a versenyt jelenti, és nem csak a közoktatásban oktató pedagógusokat tudja segíteni, de az algoritmikus gondolkodás mérésére és fejlesztésére szűkített feladathalmaz a felsőoktatásban is megállja a helyét.

További feladatomnak tartom a verseny és a kiterjesztéseinek szélesebb körbe való eljuttatását, a pedagógusok, informatika tanárok támogatását mind a tanórákon, mind az azokon kívüli tevékenységeikben.

A felsőoktatásban kifejlesztett és használt tesztek, valamint a fejlesztést támogató kurzus folyamatos értékelésével, a hallgatók nyomkövetésével igyekszem támogatni egyetemünk célkitűzését a lemorzsolódások csökkentésében és az oktatási színvonal megtartásában.

## 4. Köszönetnyilvánítás

A verseny és a hód kiterjesztések nem tudnának ilyen sikeresen helyt állni az ELTE IK tanárszakos hallgatóinak közreműködése nélkül. További köszönet illeti azokat a lelkes pedagógusokat, akik diákjaik részvételét lehetővé teszik, valamint a nemzetközi Bebras csapatnak, akik évről évre töretlenül munkálkodnak azon, hogy érdekesebbnél érdekesebb feladványok kerülhessenek terítékre.

## Irodalom

1. J. M. Wing (2006) „*Computational Thinking,*" Commun. ACM, vol. 49, no. 3, pp. 33–35.
2. ISTE and CSTA, „*Operational Definition of Computational Thinking,*" (2011) Elérhető: http://www.iste.org/docs/ctdocuments/computational-thinking-operational-definition-flyer.pdf.
3. K. Brennan, M. Resnick (2012) „*New frameworks for studying and assessing the development of computational thinking,*" pp. 1–25.
4. Casey, P., J. (1997): „*Computer programming: A medium for teaching problem solving.*", New York: The Haworth Press. Computers in the Schools, XIII, 41–51.
5. OECD, 2010;
6. Chen-Chung, L., Yuan-Bang, C., Chia-Wen, H. (2011): „*The effect of simulation games on the learning of computational problem solving*", Computers & Education, 57. 1907–1918.
7. Brennan, K, Resnick, M. (2012): „*New frameworks for studying and assessing the development of computational thinking*", AREA.
8. Brennan, K. (2011): „*Creative computing: A design-based introduction to computational thinking*", – elérhető: http://scratched.media.mit.edu/sites/default/files/CurriculumGuide-v20110923.pdf (utoljára megtekintve: 2016. 10. 25.)
9. Aiken, J. M., Caballero, M. D., Douglas, S. S., Burk, J. B., Scanlon, E. M, Thoms, B. és Schatz, M. F. (2012): „*Understanding Student Computational Thinking with Computational Modeling*", PERC Proceedings.
10. Bell, T., Witten, I. H., Fellows, M. (2010): „*Computer Sciene Unplugged*", – elérhető: http://csunplugged.org/books (utoljára megtekintve: 2016. 10. 25.)
11. CS4fun – elérhető: http://www.cs4fn.org/ (utoljára megtekintve: 2020. 10. 20.)
12. Dagiene, V. (2006): „*Information technology contests – introduction to computer science in a attractive way*", Informatics in Education, 5. 1.s., 37–46.
13. Cartelli, A., Dagiene, A., Futschek, G. (2010): „*Bebras Contest and Digital Competence Assessment: Analysis of Frameworks*", International Journal of Digital Literacy and Digital Competence. Január-Március. 24-39.
15. Pluhár, Zs. (2016): „*Az informatikai gondolkodás és a hód*", InfoDidact 2016: Informatika Szakmódszertani Konferencia, Zamárdi, Magyarország: Webdidaktika Alapítvány, (2016)
16. Pluhár, Zs. (2014): „*Az informatikai műveltség egyes dimenzióinak mérése*", In: Buda András (szerk.) XIV. Országos Neveléstudományi Konferencia. Oktatás és nevelés – gyakorlat és tudomány: tartalmi összefoglalók., Debrecen, 472

17. Pluhár, Zs., Torma, H., Törley, G. (2018): „*Hallgatói teljesítményértékelés az algoritmikus gondolkodás tükrében*", In: Szlávi, Péter; Zsakó, László (szerk.) InfoDidact, Budapest, Magyarország: Webdidaktika Alapítvány, 10 p.
18. Pluhár, Zs. Torma, H. (2019): „*Introduction to Computational Thinking for university students*", In: Pozdniakov, Sergei N.; Dagienė, Valentina (szerk.) Informatics in Schools. New Ideas in School Informatics Cham, Svájc: Springer International Publishing, pp. 200-209., 10 p.

# Az online értékelés módozatai

Rumbus Anikó

rumbus@inf.elte.hu
ELTE IK

**Absztrakt.** Cikkünkben az iskolai értékelés online formáival foglalkozunk. Az értékelés nem csak a diákok az ellenőrzőjében, bizonyítványában megjelenő számadatok lehetnek, hanem hétköznapi életünknek is fontos része. Cselekedeteiket és elért eredményeiket is különféle skálán mérik, melyek alapján besorolják őket. Ezek a vélemények befolyással lehetnek további tevékenységeikre. Az iskolai értékelés a tanulási folyamat szerves része, amelyeknek nem csak az a célja, hogy számszerű adatokat kapjanak a tanárok és a szülők a diákok iskolai teljesítményéről, hanem az is, hogy a tanulók szellemi és szociális fejlődésére hatással legyenek. Az osztályzatok differenciálnak, meghatározzák a diákok beiskolázási, továbbtanulási lehetőségeiket, azaz lényeges szerepet kapnak tanulmányaik során. Az alap cél az objektív, és egységes értékelés, aminek a megvalósítása sok esetben nehezen kivitelezhető. Különösen így van ez a mostanság fennálló pandémiás helyzetben, amikor az online oktatásra való áttérésre helyeződött a hangsúly. Ezáltal az amúgy is nehéz helyzetben lévő pedagógus társadalom még nehezebb helyzetbe került az értékelés, osztályozás területén is. A korábban bevált technikákat, szóbeli felelet, röpdolgozat, témazáró dolgozat stb., egy más kontextusba kellett áthelyezni, mégpedig az online világba, úgy, hogy az ne veszítsen az értékéből, az eddig elvárt, és előírt követelményeknek továbbra is megfeleljen. Egyelőre nem ismert olyan alkalmazás, amely minden, az osztályozásra vonatkozó követelménynek megfelelne. Több alkalmazást is kipróbálva javaslatokat teszünk a pedagógusok számára, melyek elősegítik az online téreben való értékelést. Konkrétan a következő négy programot részletezzük: Redmenta, Kahoot, Socrative, Google Űrlap. Megmutatjuk, hogy milyen számonkérési formáknál alkalmazhatók, feltárjuk előnyeiket, hátrányaikat.

**Kulcsszavak:** online értékelés, iskolai értékelés, online alkalmazás, oktatás

## Bevezetés

Attól kezdve, hogy kikerülünk a családi miliőből, bölcsődébe, óvodába, iskolába járunk, egy olyan értékelési rendszerrel is találkozunk, amelyet intézményesített kereteken belül végeznek. Már nem csak szüleink, közeli rokonaink, barátaink értékelnek bennünket, hanem számunkra idegen emberek. Az addigi értékelés minőségében is változik, hiszen az emberi tulajdonságaink mellett tantárgyi tudásunk osztályozása is előtérbe kerül.

Cikkemben az iskolai értékelést vizsgálom. A digitális technika fejlődése nem csak tanítási módszereink megváltoztatását eredményezte, hanem az értékelési, osztályozási mechanizmusainkat is. Kitérünk arra, hogyan hatott az internet széles körű elterjedése az értékelési módszereinkre, melyek a leginkább alkalmazott online értékelési appok.

### Az iskolai értékelés

Az iskolai rendszer működtetésének egyik alapvető tevékenysége az értékelés. Hozzátartozik az iskolai élet mindennapjaihoz. Meghatározza a tanórák munkájának ritmusát, a tanulók viszonyát a tanuláshoz, a tantárgyakhoz és az iskolához. A jegyek sorsdöntő jelentőségűek, hiszen az osztályzatokon múlik a gyerekek életpályája. Meghatározzák az iskolatípus választását, a felsőoktatási intézménybe való bejutás esélyeit.

Az, hogy a pedagógusok hogyan értékelnek, már akkor elkezd kialakulni, amikor ők maguk is még az iskolapadokban ülnek, és őket értékelik. Későbbiekben a tanárképzés ideje alatt a tanítási

gyakorlatok során tanulnak értékelési módokat, majd kikerülve a munkahelyükre tapasztaltabb kollégáktól is tanulhatnak értékelési módszereket, lehetőségeket.

## Elvárások az osztályzatokkal szemben

### Objektivitás

A jegyek legyenek tárgyszerűek, ne függjenek az értékelő személyétől. Nem befolyásolhatja a tanár aktuális hangulata, előítéletei, beállítódása, érzelmei. Fontos az objektivitásban az is, hogy a tanuló korábbi eredményei se befolyásolják az aktuális teljesítményének értékelését.

### Reliabilitás

Az értékelés legyen megbízható, hiszen akkor az osztályozáshoz választott módszerrel akárhányszor is végezzük el a mérést, mindig ugyanazt az eredményt kapjuk.

### Validitás

Az osztályozásnak azt a tudást kell tükrözni, amelyre az adott osztályzatot használni akarjuk. Ezt érvényességnek nevezzük. Érvényességi, validitási problémát jelent például, ha a tanuló felszereléshiány miatt kap elégtelent, hiszen az nem tantárgyi tudását tükrözi. Egy másik ilyen probléma, ha a tanuló nem készít házi feladatot, és arra kap negatív értékelést. Nem tudhatjuk minden esetben, hogy a lecke elmulasztásának lustaság volt az oka, vagy a tananyag nem tudása. [1]

Elvárnánk a jegyektől azt is, hogy a különböző iskolákból tovább tanuló diákok jegyei mögött azonos tudás álljon.

Az osztályzatok differenciáló erővel bírnak. Az ötfokozatú skála alkalmazásával csak öt kategóriába sorolhatjuk a tanulókat, azonban ismert az a tény, hogy két azonos osztályzaton belül is lehetnek tudásbeli különbségek, például gyenge négyes, erős négyes. Más országokban a probléma feloldása érdekében több (hét- tíz) fokozatú skálát is használnak. Hazánkban a középiskolából az egyetemekre bejutáshoz már nem csak az ötfokozatú skálát használják, hanem a felvételi pontszámba az érettségin elért eredmények százalékos értékét számítják. A differenciálás így pontosabb.

## Az értékelés

Sokunkban az értékelés szó hallatán az iskolai osztályzatok jutnak eszünkbe, pedig az értékelés nem azonos az osztályozással. Az osztályozás a tanulók iskolai teljesítményét számokkal, vagy szimbólumokkal minősíti, az értékelés azonban ennél összetettebb. Az értékelésnek két formája van, a minőségi (kvalitatív), és a mennyiségi (kvantitatív) értékelés. Minőségi értékelés során szóban, vagy írásban minősíthetjük a tanuló teljesítményét. Hátránya, hogy nem számszerű, sok teret ad a szubjektivitásnak, így nem összehasonlíthatók más értékelésekkel. A mennyiségi értékelésnek három típusa van, a megítélés, mely átmenet a minőségi és mennyiségi értékelés között. A becslés, melynek segítségével egy skálán helyezzük el a teljesítményt. A harmadik típusa a mérés, mely során egy rögzített skálát mérünk hozzá a teljesítményhez. [2]

Napjainkban a technika fejlődésével változnak a tanítási és az értékelési módszereink is. A papíralapú értékelések mellett online lehetőségeink is vannak. Web alapú vetélkedőket tarthatunk, mérhetjük a diákok tudásszintjét akár anonim módon is. Azonnali visszajelzést biztosítunk, valamint a tantermi aktivitást növelhetjük az okoseszközök használatával.

### Az értékelés típusai

Az alfejezetben táblázatos formában mutatom be az értékelés típusait. A típusok ismeretében az általam választott alkalmazások módszertani elemzése egyszerűbb és érthetőbb.

**Funkciója szerint**

| Típus | Funkció | Eszköz, módszer |
|---|---|---|
| Diagnosztikus (helyzetfeltáró) | Helyzetfeltárás. | Megfigyelés, diagnosztikus teszt. |
| Formatív (formáló-segítő) | Segítés, fejlesztés. | Portfólió, tanulói naplók, megbeszélések, tanulói önértékelés, társak értékelése, csoportmegbeszélés. |
| Szummatív (összegző-lezáró) | Minősítés. | Témazáró dolgozat, vizsga, bizonyítvány. |

**1. táblázat**: Az értékelés típusai funkciója szerint

**A vonatkozási kör szerint**

| Kritérium-orientált | Normaorientált | A diákhoz viszonyított |
|---|---|---|
| A tanulók teljesítményének értékét a tanulási követelményekhez viszonyítva határozzuk meg. | A tanulók teljesítményét egy adott diákpopulációhoz viszonyítjuk. | A diákokat önmagukhoz viszonyítjuk annak érdekében, hogy önismeretüket fejlesszék. |

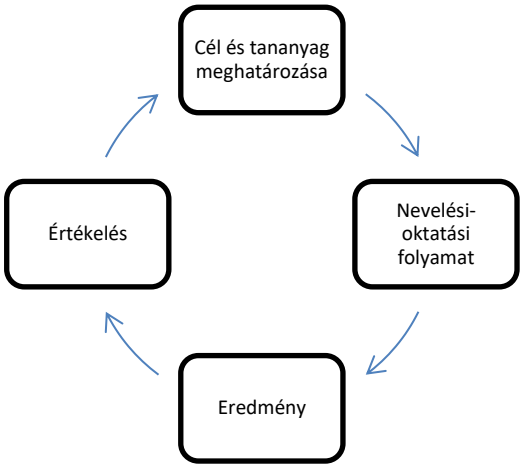
**2. táblázat**: Az értékelés típusai vonatkozási kör szerint

## Online alkalmazások az értékelésben

A diákok határozottan támogatják a mobil technológia beépítését az oktatásba. [3] Szívesen használják az oktatási célokra kifejlesztett appokat tanórákon, valamint az otthoni felkészüléshez is. Motiváltabbnak érzik magukat az eszközök használata által. Egyfajta biztonságot jelent számukra, hogy ezeknek az alkalmazásoknak a legtöbbje azonnali visszajelzést küld a nekik az aktuális teljesítményükről. Mivel az okostelefonok hétköznapi rutinjaik része, az iskolákban való használatuk komfortosabbá, otthonosabbá teszi a tantermi miliőt is. Kevésbé stresszelnek azokon a megmérettetéseken, amelyeken tudják, hogy használható az okoseszköz. Érdekesebbé, színesebbé, élvezetesebbé tehető így a tanóra.

Az alkalmazások használatában van egy nagy előnyünk. Előre fel tudunk készülni, előkészíthetjük a használni kívánt feladatsorokat, ellenőrző kérdéseket. Egyszerűbben és gyorsabban adhatunk otthoni gyakorlásra mintapéldákat. Eleinte ugyan több időbe telik a felkészülés, de a későbbiekben könnyebben alakíthatjuk az aktuális évfolyamhoz, az aktuális csoporthoz. Javítja a tanulók tanuláshoz való viszonyát is, hiszen kevésbé izgulnak egy-egy megmérettetés előtt, komfortosabban érzik magukat okostelefonjuk használatával. [4]

Az értékelés egy megerősítési, visszacsatolási folyamat, mely során az egész tanítási-tanulási folyamatot, annak hatékonyságát értékeljük. [5] Az értékelésnek ezt a funkcióját az online térben is meg kell tartanunk. Az értékelés egy ciklikus folyamat része, melynek elemei a cél- és tananyag meghatározása, a nevelési-oktatási folyamat, annak eredménye, majd az értékelés. Ezeket a lépéseket ismételjük.




**1. ábra:** Értékelés ciklikus folyamata

Az online értékelés további előnyei, hogy megfelelő internetkapcsolat megléte esetén bárhonnan elvégezhető. Egy időben egyszerre többen is értékelhetünk. A gyakorló feladatsorokkal fejleszthetjük a diákjaink önértékelési képességét, mely által önismeretük is fejlődik. Előnyei mellett azonban hátrányaival is számolnunk kell. A tanulók személyiségi jogait szem előtt kell tartanunk. Az eredmények kivetítése során ügyeljünk, hogy azt a diákok egyértelmű beleegyezésével tegyük. Meg kell tanítanunk a tanulókat az alkalmazások helyes használatára. Minél több appot vezetünk be az értékelésbe, annál több felületet kell tudniuk kezelni a diákoknak. Ez azonban még akár előny is lehet, hiszen így egy univerzális tudást is adhatunk a kezükbe. Figyelnünk kell arra, hogy a diákok más nevében ne tudjanak bejelentkezni. Ebben a veszélyforrásban több buktató is van. Előfordulhat olyan is, hogy segíteni szeretnének egymásnak, de olyan is lehetséges, hogy társuknak ártani szeretnének, és egy szándékosan rossz megoldássort töltenek ki társuk helyett. Meg kell oldanunk a felhasználónév/jelszó ismeretének problémáját. Sok esetben tapasztaltam, hogy amennyiben ritkábban használják az appokat, elfelejtik a jelszavukat, felhasználó nevüket, melynek pótlása időigényes. Ügyelnünk kell arra is, hogy ne tekintsék játéknak a felméréseket annak ellenére sem, hogy okostelefonnal végezhetik azt el. Technikai hiba esetére is legyünk felkészülve. Például lehessen kinyomtatni a feladatsort, ha valami ok folytán nincs internet kapcsolat.

## Kahoot

2013-ban fejlesztett norvég oktatási célú program. Az alkalmazás működése könnyen megérthető, megtanulható. Alacsony a belépési küszöb. Már az első alkalommal tudják a diákok teljes mértékben használni. A használata során nagyszámú diák mozgatható egyszerre. [6]

Lehetőséget ad videók, képek, szövegek, zenei elemek beépítésére, valamint az időkorlát meghatározására.

A tanárok a get!Kahoot alkalmazást használva készíthetik el feladataikat, a diákok a kahoot.it-n keresztül érik el azokat.

A Kahoot használata a tanítási, értékelési folyamat játékosításának, gamifikálásának ad lehetőséget. A diákok szívesebben tanulnak játszva, mobiltelefonjukat használva. [7] A számos, mások által készített, szabadon felhasználható feladatlapokat könnyen a saját igényeinkhez alakíthatjuk. A tanulók saját okoseszközeiket használhatják, így nem látják egymás válaszait. Valós időben kapjuk a visszajelzést mind a tanári, mind pedig a diák oldalon. Az azonnali visszajelzés pozitívan hat a tanulókra. Nagyobb a motivációjuk olyan értékelési módokkal szemben, melyekről tudják, hogy azonnal

látják eredményeiket. A tanároknak is kedvező ez a funkció, hiszen a javítási, értékelési idő jelentősen lerövidülhet.

A Kahoot használatához okoseszközökkel és megfelelő internet kapcsolattal kell rendelkezni. Ezen feltételek biztosítása elengedhetetlen. Egyik hátránya tehát, hogy megfelelő infrastruktúra nélkül nem tudjuk alkalmazni. Napjainkra a BYOD (Bring Your Own Device), a diák használhassa saját eszközeit az iskolában is, egy fontos kutatási terület lett az oktatásban, mely a Kahoot használatával kapcsolatban is felmerül. Amennyiben az iskolának nincs elegendő eszköze, ám mégis szeretnénk módszertani repertoárunkat ezzel az appal bővíteni, lehetővé kell tennünk a BYOD zökkenőmentes használatát. A feladatlapok elkészítése eleinte több időt vesz igénybe, ám a későbbiekben ez a befektetés megtérül. A feladatlapokat többször is felhasználhatjuk. Könnyen testre szabhatjuk az aktuális csoportjainknak megfelelően.

Óvatosan kell bánni a kvízek közben tett kiemelésekkel, megjegyzésekkel. A technológia használata ne cél, hanem eszköz legyen. Ha a tanítandó tananyaghoz nem megfelelő, a tanár nem felkészült, ha a módszertani eszköztár szegényes, ha az óra nem tanuláscentrikus, ha a tanulók passzívak, akkor alkalmazhatunk bármilyen appot, tabletet, a tanóra nem lesz eredményesebb. Megtehetjük, hogy a tanulók tartalmakat, játékokat készítsenek egymásnak. Rengeteg lehetőséget kínál arra, hogy az egyéni munkájukat a tanár önértékelő eszközökkel, leírással, társértékeléssel támogassa.

Legismertebb Kahoot funkció a kvíz. Feleletválasztós kérdések jelennek meg a kivetítőn, melyekre a diákok a saját készülékükön (mobil appon, tableten, vagy asztali pc-n) válaszolnak úgy, hogy a megfelelő válasz színét kiválasztják. Minden kérdés után megtudjuk az aktuális pontszámot, melyeket a válasz helyessége, és a válasz gyorsasága befolyásol. Egyéni és csoportos versenyekre is lehetőséget ad. Ingyenes és fizetős verziója létezik. Az ingyenes verzióban a kérdések száma nincs limitálva, ám a játékosok száma maximum 50 fő lehet.

## Socrative

A hagyományos, frontális típusú előadások nem kötik le a mai kor hallgatóit, passzívnak érzik magukat, monotonnak, egyoldalúnak tartják az előadásokat. A Socrative alkalmazás támogatja az aktív tanulást. [7] Az együttműködésen, csoportmunkán alapuló aktív tanulás növeli a diákok érdeklődését. A közös tanulás során a tanulók tudása hatékonyabban bővülhet.

A Socrative is valós idejű visszacsatolást biztosít. A diák és tanár számára is azonnal látható az elért eredmény. Különböző feladattípusokat készíthetünk az appal, melyek az alábbiak:

- Quiz (kvíz);
- Space race („Űrhajós verseny”);
- Exit ticket (Elérte-e a diák a megfelelő tudásszintet);
- Multiple choice (Többszörös választás);
- True/false (Igaz/hamis);
- Short answer (Rövid válasz).

A feladatlapok készítése során fontos, hogy egy kérdést pontosan, jól fogalmazzunk meg. A kiértékelés automatikusan történik, így a tanulók azonnal megtudják elért eredményüket. Könnyen felkészülhetünk ad hoc helyzetekre, hiszen lehetőség van az elkészített értékelő teszt pdf letöltésére. Használata szinte megegyezik a Kahoot alkalmazásával. Különbségek a feladattípusokban vannak. A Space Race típus kifejezetten a versenyeztetésre alkalmas, és fokozza a diákok órai aktivitását. A tanórákat érdekesebbé, játékosabbá teszi.

A tanár által megadott belépőkóddal lehet belépni, így csak azok tudnak részt venni a játékban, akik megkapták a kódot. Kevésbé lehet csalni, de megeshet, hiszen csak a szobakódot kell megadni. Ugyanazzal a jelszóval csak egy ember tud bejelentkezni.

Van benne képfeltöltési funkció, valamint médiatartalmat is csatolhatunk a kérdésekhez. Pro verzióban (fizetős verzió) maximálisan 10 csoportot tudunk létrehozni, ami nem biztos, hogy egy tanárnak elég is lenne, így több felhasználói fiók létrehozásával tudunk tíznél több csoportot létrehozni. Nem ad részpontokat, ami nehezíti a pontgyűjtést, ezért törekedjünk olyan kérdések feltevésére, amelyekre a válaszok nem összetettek, hogy igazságosabb legyen a pontozás. Nem csak a helyes válasz számít a végső rangsorba, hanem a válaszra adott idő is. A helyes válaszadók közt az kerül előbbre, aki hamarabb adta a jó megoldást. [8]

A Socrative is alkalmas a tanulási folyamat játékosítására. Az elkészített feladatlapjainkat megoszthatjuk, a szabadon felhasználhatókat pedig saját igényeinknek megfelelően alakíthatjuk. Könnyen aktualizálhatók a feladatlapok. Az évek múlásával könnyen bővíthetjük feladatbankjainkat. Könnyű testre szabni az aktuális csoportjainkra a már meglévő példáinkat. A Socrative használatával interaktívabbá, érdekesebbé és színesebbé tehetjük tanóráinkat.

A Socrative használatához is felmerül az eszköz- és a megfelelő internet igénye, valamint a BYOD kérdése. Ügyeljünk arra, hogy a túl gyakori használata unalmassá teheti az appot.

## Google Űrlap

A tanuló értékelése gyakran a tanítás értékelésének elsődleges eszköze. A vélemények hatással lehetnek a teljesítésre, magabiztosságra és tovább haladásra. Fontos a tanulási folyamat rendszeres nyomon követése, ne csak a téma végén értékeljünk. A formatív értékelés kapjon nagyobb hangsúlyt.

A Google Űrlap egy Google szolgáltatás, mely Google fiókkal ingyenesen hozzáférhető. A diákok által kitöltött feladatlapok eredményeiből Web alapú táblázatot tudunk készíteni az adatok elemzéséhez, vagy akár Microsoft Excel táblába is importálhatjuk azokat. Az eredmények elemzése, kiértékelése így könnyebb, egyszerűbb, pontosabb. Diagrammokat, ábrákat készíthetünk az adatokból, hogy még szemléletesebbé tegyük vele a kiértékelést. [9]

Innovatív oktatási módszer kialakítását teszi lehetővé a Google Űrlap használata. Diákok körében nagy a népszerűsége, hiszen azonnali visszajelzést ad, valós idejű munkát végezhetünk vele. A valós idejű munka annyit jelent, hogy amint végeztünk a feladatlap kitöltésével, azonnal láthatjuk az elért eredményünket.

Megfelelően alkalmazva a diákok motiváltabbak, aktívabbak lesznek, tanóráink színesebbé, interaktívvá válnak. A Google Űrlap készítése nem igényel komolyabb szoftverkezelési ismereteket, kezelése könnyen elsajátítható. [10]

Az alábbi feladattípusokat készíthetjük el, valamint kérdőívkészítési lehetőségekkel rendelkezik a Google Űrlap:

- Feleletválasztás;
- Rövid válasz;
- Legördülő lista;
- Lineáris skála;
- Feleletválasztós rács;
- Jelölőnégyzetrács;
- Jelölőnégyzetek;
- Kép, és videó hozzáadás.

A feladatlapok internet hozzáféréssel, megfelelő eszközzel bárhonnan elérhetők. Sokféle feladattípusból válogathatunk az űrlapok elkészítéséhez. A Google Űrlap nem csak feladatlapok készítéséhez alkalmas, hanem kutatáshoz szükséges kérdőívek elkészítéséhez és kiértékeléshez is. Többféle értékelési típushoz alkalmazható. Hosszabb kifejtést igénylő esszé típusú számonkéréshez is, de rövidebb, feleletválasztós kérdéssorhoz is megfelelő.

Internet és eszközigényes. Felmerül a BYOD problémája is az alkalmazhatóságnál. Az űrlapok elkészítése, valamint azok későbbi testre szabása az aktuális csoportjainkhoz időigényesebb. A befektetett idő a kiértékeléskor térül meg. A kérdőívek készítése komoly előkészületeket igényel. Alaposan végig kell gondolnunk, hogy melyek azok a feladattípusok, amelyekkel a kért számonkérést megfelelően elvégezhetjük.

### Redmenta

A modern oktató egyik módszertani eszköze. Azonnali visszajelzést ad a diákok teljesítményéről. Dolgozhatunk benne online, okostelefonon, vagy tableten. Az oldal logikusan felépített, könnyen kezelhető, és sok feladattípus készítésére alkalmas. Két magyar fiatal, Mérő Bálint és Bordás Ádám kezdték el fejleszteni 2013-ban.

Felhőalapú feladatkészítő applikáció. Tesztlapokat készíthetünk és oszthatunk meg diákjainkkal. Csoportokat hozhatunk létre, melyekhez hozzárendelhetjük a csoporttagokat. Lehetőségünk van a csoporttal megosztani a feladatlapokat, nem szükséges egyesével a tanulókhoz rendelni azokat.

Az app használata és funkciói ingyenesen használhatók. Készíthetünk gyakorló és felmérő feladatlapokat egyaránt. Amennyiben nem csak gyakorló feladatlapok megosztására, hanem számonkérésre is szeretnénk használni, érdemes a diákokat valódi nevükkel regisztráltatni, hogy beazonosíthassuk őket. Egyik hátránya pont ebből adódik, hogy az anonimitást így nem tudjuk biztosítani.

Az elkészített feladatlapokat könnyen szerkeszthetjük, bővíthetjük, és alakíthatjuk át az adott csoportunkhoz. Bárhonnan elérhető megfelelő internetkapcsolattal. Akár az otthon lévő tanulókkal is kitöltethetjük a tesztet. A Redmentában vannak olyan beállítási funkciók is, melyekkel a kitöltések számát, a kitölthetőség idejét is meg tudjuk adni. Ha például egy adott napon a második órában szeretnénk a felmérést elvégeztetni, akkor beállíthatjuk, hogy 9:45-10:40-ig lehessen a feladatlapot kitölteni. Ez a funkció hasznos lehet akkor is, ha helyettesítés van a számonkérés ideje alatt. Ahhoz, hogy a diákok ne tudják egymás kijelzőiről leolvasni a válaszokat, beállíthatjuk azt, hogy a feladatok véletlenszerű sorrendben jelenjenek meg. A pontozás beállítására is lehetőségünk van. A kifejtős válaszoknál azonban érdemes a gépi pontozást felülbírálni. A Redmenta egyelőre úgy van fejlesztve, hogy csak az előre rögzített választ fogadja el, így ha a diák a saját szavaival ad helyes választ, nem fogadja azt el, és nem ad rá pontot.

A Redmenta ön- és társértékelésre nem alkalmas. A csalás lehetősége is felmerülhet, hiszen a feladatok megoldása közben lehetőség van a visszalépésre.

## Módszertani ajánlás az alkalmazások használatához

Innovatív tanítási módszer. A tanulók aktív részvételükkel fejleszthetik saját tudásukat. Szabadabb kommunikációt biztosítanak a tanár és tanulók között. Anonimitást is tudunk biztosítani. Az azonnali visszajelzés növeli a tanulók fejlődés iránti vágyát, fejleszti kritikus gondolkodásukat. Valós időben méri fel a hallgatók tudását. Az elsajátított tudás, és a megértése a tananyagnak azonnal észlelhető. Használatuk során motiváltabbak a diákok. A szoftverek egyre gazdagabb visszajelzési lehetőségeket nyújtanak. Az online értékelésbe beépíthetünk hang, kép, szöveg, klip, szimulációs anyagokat is, melyeknek ösztönző erejük van. [11]

Az alkalmazások internet igényesek, emiatt alternatív megoldásokat is be kell iktatnunk, amennyiben az iskolai feltételek nem megfelelőek. Például legyenek nyomtatott feladatlapjaink internethiány esetére. A diákok többsége már rendelkezik mobil internettel, melyet akár meg is oszthat hotspoton keresztül, vagy beoszthatjuk őket úgy csoportokba, hogy mindegyik csoportban legyen legalább egy tanuló, akinek van internet hozzáférése. Ehhez természetesen a jogi feltételeket előzőleg ki kell dolgozni, és be kell építeni a házirendbe. [12]

Az appok bármilyen jók is, nem elegendőek a teljes megújuláshoz. Ezeket az alkalmazásokat is lehet nem megfelelően használni. A túl sokszori felhasználás unalmassá teheti az appokat és a tanórákat. Vigyázzunk arra, hogy ne váljunk sablonossá. A mai generációnak fontos a változatosság, így törekedjünk arra, hogy többféle módszert váltogassunk tanóráinkon.

A Kahoot használata során az egy tanulóra eső figyelem elenyésző, sok egyéni visszajelzés elmarad. Nem szerencsés szummatív értékelésre használni, vagy az órai versengésre kiélezni. A Kahoot kifejezetten alkalmas formatív, vagy diagnosztikus visszajelzés céljából.

A Socrative-ben egy helyen található játékos verseny, értékelés és komoly teszt. Alkalmas formatív és szummatív értékelésre egyaránt. Visszajelzést kapunk a diákok tudásának aktuális állapotáról, pontgyűjtéses feladatlapokkal nagyobb aktivitást érhetünk el. Használata során vidámabb hangvételű órákat tarthatunk, és akár a versenyzést is beépíthetjük módszereink közé. Leginkább a formatív, és diagnosztikus értékelésre alkalmazzuk.

A Google Űrlap felmérések készítése mellett alkalmas közvélemény kutatásra, kérdőívek készítésére. Az egyik olyan alkalmazás, mellyel a formatív, és diagnosztikus értékelés mellett szummatív értékelésre is alkalmazzunk.

A Redmenta a másik olyan alkalmazás, amelyet szummatív értékelésre is használhatunk a formatív és diagnosztikus értékelés mellett. A diákok csoportba sorolása lehetővé teszi a tanár számára a csoportok közti elemzést.

Az alkalmazások módszertani felhasználhatóságát az alábbi táblázatban foglaltam össze:

| App neve: | Felhasználhatóság: |
|---|---|
| Kahoot | – Az új szavak, fogalmak memorizálása;<br>– Meglévő tudás felelevenítése;<br>– Tudásfelmérő:<br>– Téma átismétlése;<br>– Tananyag begyakorlása;<br>– Csoportmunkában új kvízeket készíthetnek.<br>– Óra elején a korábbi anyag felelevenítése;<br>– Óra végén az anyag főbb pontjainak ismétlése, rögzítése, összefoglalása;<br>– A diákok eredményének megismerése, érdeklődés felkeltése, vita kezdeményezése;<br>– Tanulmányi, vagy osztálykirándulás, szalagtűző ruha, zene kiválasztása stb. megvitatása.<br>– Szóbeli felelet ön-és társértékelése. |
| Socrative | – Az új szavak, fogalmak memorizálása;<br>– Meglévő tudás felelevenítése;<br>– Tudásfelmérő:<br>– Téma átismétlése;<br>– Tananyag begyakorlása;<br>– Óra elején a korábbi anyag felelevenítése;<br>– Óra végén az anyag főbb pontjainak ismétlése, rögzítése, összefoglalása;<br>– Önértékelés;<br>– Szóbeli felelet ön-és társértékelése. |
| Google Űrlap | – Meglévő tudás felelevenítése;<br>– Tudásfelmérő:<br>– Téma átismétlése;<br>– Tananyag begyakorlása;<br>– Otthonra gyakorló és házi feladatok készítése;<br>– Kérdőív készítése és megosztása. |
| Redmenta | – Meglévő tudás felelevenítése;<br>– Tudásfelmérő:<br>– Téma átismétlése;<br>– Tananyag begyakorlása;<br>– Otthonra gyakorló és házi feladatok készítése. |

**3. táblázat**: Az alkalmazások módszertani felhasználása

## Összegzés

Cikkünkben bemutattunk négy online alkalmazást, Kahoot, Socrative, Google Űrlap, Redmenta, melyeket az iskolai értékelésben is felhasználhatunk. Az appokkal interaktívvá, élvezetessé tehetjük tanóráinkat. Több tanulmány szerint is fontos, hogy oktatási módszereink közt szerepeljenek mobil- és okoseszközökkel támogatott lehetőségek. [13] Az eszközök becsempészése az osztálytermekbe lehetőséget adnak a gamifikációra, színesebbé, élvezetesebbé teszik tanóráinkat. Tanítási módszerink tárháza bővíthető általuk.

A kvízek, melyeket a cikkünkben felsoroltunk, hasznosak a tanároknak, diákoknak egyaránt, hiszen azonnali visszajelzést adnak a diákok tudásáról, és arról is információt kaphatunk, hogy a tananyag elsajátításában hol tartunk. Az értékelés mindhárom fajtájára, diagnosztikus, formatív és szummatív értékelésre alkalmasak.

A feladatlapok elkészítése segítheti a tanárokat abban, hogy szisztematikusan átgondolják, hogy mi az, amit értékelni szeretnének, milyen nehézségi szintű feladatokat építsenek be. A különféle kérdéstípusok lehetőséget adnak arra, hogy különböző gondolkodási műveleteket használjanak a tanulók a válaszadáshoz.

Mindegyik alkalmazás esetén lehetőségünk van feladatbankok létrehozására, melyeket későbbi években fejleszthetünk, formázhatunk, és további feladatokkal bővíthetünk. Redmenta esetén a véletlenszerű feladatkiosztás miatt egyre több különböző feladatsort alkothatunk néhány kattintással.

A feladatlapok összeállítása, megszerkesztése, a feladatbankok megalkotása ugyan jelentős többlet időt jelent az eleinte, ám hosszú távon ez a befektetett idő megtérül. Úgy gondolom, hogy a digitális technológia mai helyzetében minden pedagógusnak meg kellene ismerni, és módszertani repertoárjába be kellene építeni a fentiekben tárgyalt alkalmazások valamelyikét.

## Hivatkozások


1. Csapó, B. (1998). *Az iskolai tudás felszíni rétegei:mit tükröznek az osztályzatok?* Budapest, Pest, Magyarország.
2. Hunline..*Hungarian Online University.* http://okt.ektf.hu/data/szlahorek/file/hunline_pedpszi/16_pedagogus_mesterseg/416_az_rtkels_formi.html(2020.11.29.)
3. Dahlstorm Eden, C. D. (2012). *ECAR Study of Undergraduate Students and Information Technology*. EDUCASE Center for Applied Research, Amerika.
4. Terrion Jenepher Lenox, A. V. (2012). *Perceptions of the effects of clicker technology on student learning* Research in Learning Technology, 20.
5. Parázsó, L. T. (2013). *On-line értékelési módszerek I.* Eger, Magyarország.
6. Komposzt. (2018. 07 23). *K.O.M.P.O.SZ.T.* https://komposzt.wordpress.com/2018/07/23/a-kahoot-jelenseg/(2020.11.15.)
7. Fehér, P. (2020. 8. évf., 2. szám). *„Húsz év múlva" – A digitális oktatás helyzete, eszközei,trendjei világszerte.* Gyermeknevelés Tudományos Folyóirat, 350-372.
8. Michael, C. (2011. 04 01). *Students' experiences of active engagement through cooperative learning activities in lectures.* Sage Journals, Ausztrália.
9. Havassy, A. (2019. 10 26). *A 21. századi tanulásszervezés.* https://havassy.wordpress.com/tag/socrative/(2020.11.30.)
10. Gehringer, E. F. (2016. 01 11). *Daily course evaluation with Google forms.* Amerika, North Carolina.

11. Mireille Djenno, G. M. (2015. 06 01). *From paper to pixels: using Google Forms for collaboration and assesment.* ISSN: 0741-9058.

12. Gaál, G. D. (2015). *Tervezés és értékelés.* http://okt.ektf.hu/data/szlahorek/file/kezek/02_terv_ert/524az_rtkels_funkcii.html/(2020.12. 01.)

13. Wendy Hsin-Yuan Huang, D. S. (2013 decembere 2013.12.10.). *A Practitioner's Guide To Gamification of Education.* Toronto, Canada.

14. Balassa, E. *Tanulási Zóna.* https://tanulasizona.hu/kahoot-az-interaktiv-kvizkeszito-alkalmazas/(2020.12.02.)

# Az informatikai gondolkodás fejlesztése szakköri és tábori tevékenységeken keresztül

Solymos Dóra

solymos.dora@inf.elte.hu
Eötvös Loránd Tudományegyetem, Informatikai Kar

**Absztrakt.** Az informatikai gondolkodás egy fontos kulcskompetencia modern korunk diákjai számára. Egyes vélemények szerint ezen kompetenciával mindenkinek rendelkeznie kellene, és oly módon lenne szükség elsajátítani, mint az írást, olvasást vagy számolást. Ezt az elvárást érzik a szülők és a gyerekek is, hiszen egyre nagyobb az érdeklődés a programozás tanulás és a robotika iránt. Számos ilyen témájú szakkör és tábor működik a hazai kis- és nagyvárosokban, azonban az még tisztázatlan, hogy melyik típusú foglalkozáson sajátítanak el többet a programozás alapjairól a gyerekek: A táborokban vagy a szakkörökben? Kutatásunkban erre kerestük a választ az Eötvös Loránd Tudományegyetem Informatikai Karán szervezett szakkörök és táborok során. A felmérés során anonim kérdőíveket töltöttek ki a résztvevő gyerekek, melyben többek között a programozás három definíciójáról (ciklus, elágazás, változó) kérdeztük őket. A kutatás folyamatáról és az eredményeiről cikkünkben számolunk be.

**Kulcsszavak:** programozás tanítás, informatikai gondolkodás, kérdőíves kutatás, szakkör, tábor

## 1. Az informatikai gondolkodás (Computational thinking)

Az informatikai gondolkodás (angolul: Computational thinking, CT) kifejezést az 1950-es évek óta használjuk, azonban a kifejezés 2006 után robbant be a köztudatba, miután Wing újra definiálta azt [1]. Ekkor ő olyan folyamatként határozta meg a CT-t, amely magába foglalja a problémamegoldást, a rendszerszintű tervezést és az emberi viselkedés megértését, mindezt olyan formában, hogy a folyamat során az informatikai alapfogalmak felhasználhatóak legyenek. Ezt a definíciót széles körben elfogadták általános jellege miatt, azonban felmerült az igény egy specifikusabb definíció meghatározására is, amelyet már az oktatás során is használni lehet. [2] [3] [4] [5] [6]

Az International Society for Technology in Education az alábbi meghatározást adta meg az informatikai gondolkodásról: Az informatikai gondolkodás egy olyan problémamegoldó folyamat, amely magába foglalja a probléma meghatározását olyan formában, hogy az tartalmazza a megoldáshoz szükséges (informatikai) eszközöket. Az adatokat logikailag rendezi és elemzi, majd modellek és szimulációk segítségével reprezentálja őket. A probléma feltérképezése után, megkeresi a leghatékonyabb algoritmust a megoldás megadásához, majd általánosítja a problémát és a megoldást is [7].

## 2. Szakkörök és táborok

Az Eötvös Loránd Tudományegyetem Informatikai Karának T@T Kuckójában 2018-ban rendeztünk először programozói tábort 10–14 éves diákok számára. A táborozókkal a programozás alapjait ismertettük meg a LEGO© Mindstorms EV3 robot segítségével. A táborokat 2019 tavaszán szakkörök formájában folytattuk, amikor a LEGO© Mindstorms EV3 roboton kívül a BBC Micro:bit eszközzel is megismerkedhettek a gyerekek egy erről szóló foglalkozássorozaton. A szakkörök és táborok kialakulásáról és eddigi tapasztalatainkról a [8] cikkben számoltunk be részletesen.

A táborok napközisek voltak, ami azt jelenti, hogy a gyerekek reggel 8 órától 17 óráig tartózkodtak a Kuckóban. A délelőtti idősávban (9–12 óra között) programozással foglalkoztak a táborozók, míg délután, planetáriumi látogatáson, drónfoglalkozáson vagy más informatikai témájú előadáson vettek részt. A tavaszi és őszi szakköri foglalkozásokat az iskolai programokhoz igazítottuk, így a jelentkezők egy 10 hetes, heti másfél órás foglalkozássorozaton ismerkedhettek a programozással.

Mind a táborokban, mind a szakkörökön ugyanazt a tananyagot sajátíthatták el a gyerekek, csupán az ismeretek időbeni felosztása különbözött. A gyerekek kis létszámú, legfeljebb 10 fős csoportokban tanultak programozni lelkes tanár szakos hallgatóktól.

# 3. A kérdőíves kutatás bemutatása

A Kuckóban tartott szakkörök és táborok kapcsán vetődött fel néhány kérdés, melyek a következők voltak: Mit jegyeznek meg a foglalkozásokon a gyerekek? Mely típusú foglalkozásokon való részvétel során szereznek több ismeretet a programozás alapjairól a gyerekek? Mikor mondhatjuk azt, hogy fejlődött a programozás tudása annak a diáknak, aki eddig nem tudott programozni? Mit mondhatunk arról a diákról, aki már tudott programozni? Kutatásunk célja, ezeknek a kérdéseknek a megválaszolása.

## 3.1. A kutatás bemutatása

A kutatás három fő szakaszra osztható. Az első foglalkozáson, a programozás oktatás előtt, megkértük a diákokat, hogy töltsenek ki egy anonim kérdőívet. Fontosnak tartottuk kihangsúlyozni, hogy nem tesztről van szó, hanem kérdőívről, melynél, ha nem tudják a választ egy kérdésre, akkor annak nem lesz semmilyen következménye.

A gyerekek személyiségi jogainak védelme érdekében döntöttünk az anonim kérdőívek mellett, még annak ellenére is, hogy így számos kérdést nem tudtunk vizsgálni. A kérdőívben az anonimitást kiterjesztettük a nem kérdésére is, ugyanis, elég gyakori, hogy csak egy–két lány vesz részt a foglalkozásokon, így az ő válaszaik könnyen beazonosíthatóak lettek volna.

A kutatás második szakasza az oktatási tevékenységet tartalmazta, amely során a résztvevők megismerkedtek a ciklus, az elágazás és a változó fogalmakkal. A szakkörökön a változó fogalmának a bevezetésére nem szokott sor kerülni, ugyanis (néha) olyan leterhelten és fáradtan érkeznek a gyerekek a délutáni szakköre, hogy 90 perc alatt sem tudunk annyi érdemi munkát végezni velük, amivel jelentős előrelépést érhetünk el akár az eszköz megismerésében, akár a programozás alapjainak elsajátításában. Azonban arra különösen figyelünk, hogy a szakkör végére a választott eszközt, ill. a ciklus és az elágazás szerkezeteket magabiztosan tudják használni.

A táborokban alkalmazott $2 \times 90$ perces időkeret és az energikusabb gyerekanyag lehetővé tették, hogy be tudjuk vezetni a változó fogalmát elméleti és gyakorlati szinten is. Továbbá, a 2019-es táborokban lehetővé tettük a gyerekeknek, hogy egy olyan haladó szintű foglalkozásra jelentkezzenek, ahol a fő téma a változókról tanultak elmélyítése és gyakorlása volt. Ebbe a csoportba egy előzetes teszt kitöltése után kerülhettek be a gyerekek. A tesztben felmértük, hogy milyen szinten tudják használni a LEGO© Mindstorms EV3 eszközt, ill. meg tudják-e különböztetni a ciklus és az elágazás szerkezeti elemeket gyakorlati példák megoldásában.

A kutatás harmadik szakasza egy újabb kérdőív kitöltését jelentette. Ez nagyon hasonló volt az előkérdőívhez, azonban itt a demográfiai kérdések helyett inkább a programozási alapfogalmak ismeretére koncentráltunk.

## 3.2. A kérdőív bemutatása

Összesen négy kérdőívvel dolgoztunk, volt két előkérdőív és két utókérdőív. A két típus közül az egyik a szakkörön résztvevők számára készült, a másik pedig a táborozók számára. A kérdőívek kérdései az 1. táblázatban láthatóak, teljes egészében pedig online[1] megtekinthetőek.

| Sorszám | Kérdés | Kérdés típusa |
|---|---|---|
| 1. | Melyik táborban veszel részt? | Egyszeres választás |
| 2. | Milyen suliba jársz? | Egyszeres választás |
| 3. | Tanulsz/tanultál valamilyen idegen nyelven? | Egyszeres választás |
| 4. | Ha tanultál, akkor milyen nyelv(ek)en? | Többszörös választás |
| 5. | Tanultál valamilyen nyelven programozni? | Egyszeres választás |
| 6. | Ha tanultál, akkor milyen nyelven? | Többszörös választás |
| 7. | Mi az a változó? | Nyitott szöveges kérdés |
| 8. | Mit nevezünk ciklusnak? | Nyitott szöveges kérdés |
| 9. | Mit nevezünk elágazásnak? | Nyitott szöveges kérdés |
| 10. | Programozni szeretnél megtanulni vagy a robotot használni? | Egyszeres választás |
| 11. | Mi leszel, ha nagy leszel? | Nyitott szöveges kérdés |

**1. táblázat:** A kérdőívek kérdéseit tartalmazó táblázat.

A foglalkozás elején és végén feltett kérdéssorok között annyi különbség volt, hogy az utókérdőívben nem kérdezünk rá még egyszer az iskola típusára, az idegen nyelv ismeretére és jövőbeni céljaikra. Azért nem tartottuk fontosnak ezeket újra megkérdezni, mert egyéni összehasonlításra az anonimitás miatt nem volt lehetőségünk, csupán csoportok szintjén tudtuk volna elemezni az adatokat. Ezen kívül a szakköri kérdőív tartalmazott egy plusz kérdést arra vonatkozóan, hogy hogyan érezték magukat a gyerekek a foglalkozások során.

### 3.2.1. A kérdésekre adható válaszok

Mind a kérdések, mind a válaszok megfogalmazásánál arra törekedtünk, hogy a gyerekek számára könnyen érthető és ne teszt jellegű legyen a kérdőív. Ezen törekvésünk eredményeként lett informális jellegű a kérdéssor és részben emiatt kerültek bele a 10. és 11. kérdések.

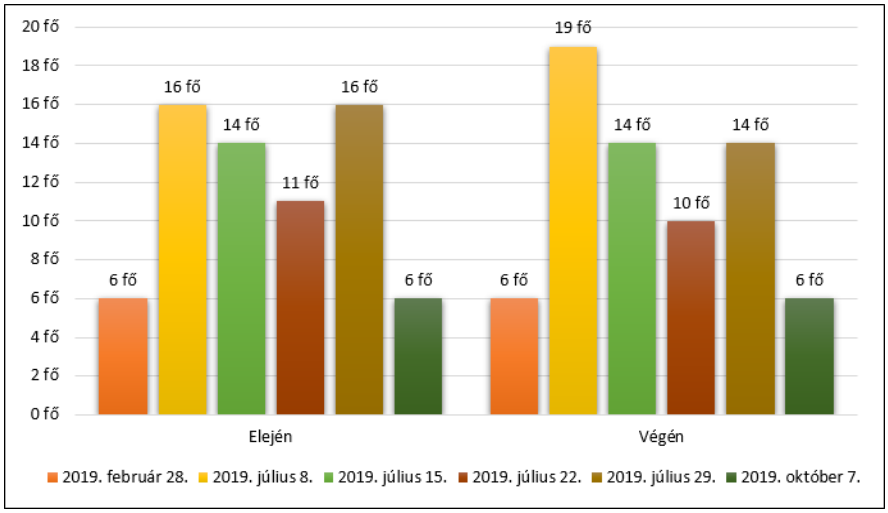
Az 1. táblázat második kérdése (*Milyen suliba jársz?*) az iskola típusára kérdez rá, így a válaszlehetőségek a következők: *általános iskola, 4, 6 és 8 osztályos gimnázium*.

A kitöltés gyorsításának érdekében a legtöbb kérdésnél adtunk meg válaszlehetőségeket és minimalizáltuk a kifejtős kérdések számát. A válaszoknál a közoktatásban előforduló lehetőségeket adtunk meg, de a kitöltőknek volt lehetőségük egyéni/egyéb válaszok megadására is.

[1] Szakkör (előkérdőív): https://forms.gle/VzgQU4bGNqh2nH846
Szakkör (utókérdőív): https://forms.gle/vzhFLaRnyfugKTh78
Tábor (előkérdőív): https://forms.gle/TKFhhZjcEUf1SuwZ7
Tábor (utókérdőív): https://forms.gle/ovgxbGesee9WehMAA

# 4. Eredmények

A kérdőívet 69 diák töltötte ki, melyből 12-t szakkörökön és 57-t táborokban válaszoltak meg a gyerekek. Az elő– és utókérdőíveket kitöltők létszáma megegyezik, de csoportonkénti eloszlása már eltér. (1. ábra) Ez a létszámingadozás a táborokban jellemző, ahol néha előfordul, hogy hétfő reggel elkésnek vagy péntek délután hamarabb elmennek a foglalkozásokról a gyerekek.




**1. ábra:** A kérdőívet kitöltők létszáma látható a diagramon a foglalkozás kezdete szerinti időrendi sorrendben.

Az időpontok a foglalkozások kezdő időpontjait jelölik, melyek közül az első (2019. február 28.) és az utolsó (2019. október 7.) szakköri alkalmak, míg a többi időpont egy-egy tábori turnust jelöl.

Nemre irányuló kérdés ugyan nem szerepelhetett a kérdőívben, de a névsorok alapján a két szakkörön összesen 4, míg a táborokban 12 lány vett részt.

A kutatás célcsoportja a 10–14 éves korosztály volt, azonban 9 és 15 éves diákok is voltak a kitöltők között. Ez azzal magyarázható, hogy a táborokba való jelentkezés során több szülői megkeresés is érkezett a korosztály bővítésére egy-egy kistestvér vagy legjobb barát kedvéért. A kérésnek eleget tettünk, így ők is részt vettek a kutatásban.

## 4.1. Az eredmények leíró statisztikai elemzése

A gyerekek különböző előzetes tudással, iskolai és családi háttérrel érkeztek hozzánk, melyről egyrészt a válaszaikból, másrészt a velük eltöltött időből értesültünk. A gyerekek, válaszaik alapján, két csoportra oszthatók. A nagyobbik csoport (71%) az általános iskolákban tanulók, a másik pedig a gimnazisták csoportja.

Az Oktatási Hivatal 2018-ban készíttetett *Vizsgálat a köznevelésben folyó idegennyelv-oktatás kereteiről és hatékonyságáról* című tanulmányában a 7. és 11. osztályos diákok nyelvtudását ismerteti [9]. A felmérésből kiderül, hogy ebben a korosztályban a legtöbb gyerek csupán egy idegen nyelvet ismer. Ez a nyelv a megkérdezettek 69%-nál az angol, 31%-nál a német, azonban a francia és a kínai nyelvek egyáltalán nem jelennek meg, annak ellenére, hogy ezek is választhatóak első idegen nyelvként [10]. A kutatásban részt vevő hetedikesek 4%-a, míg a tizenegyedikesek 35%-a tanul második idegen nyelvet. Harmadik idegen nyelv ismeretét csak a 11. osztályosoknál vizsgálták, ahol a résztvevők közül csupán egy diák ismert három idegen nyelvet.

| | Az idegen nyelvet beszélők száma |
|---|---|
| angol | 57 fő |
| német | 17 fő |
| bolgár | 1 fő |
| francia | 4 fő |
| japán | 3 fő |
| olasz | 4 fő |
| török | 1 fő |

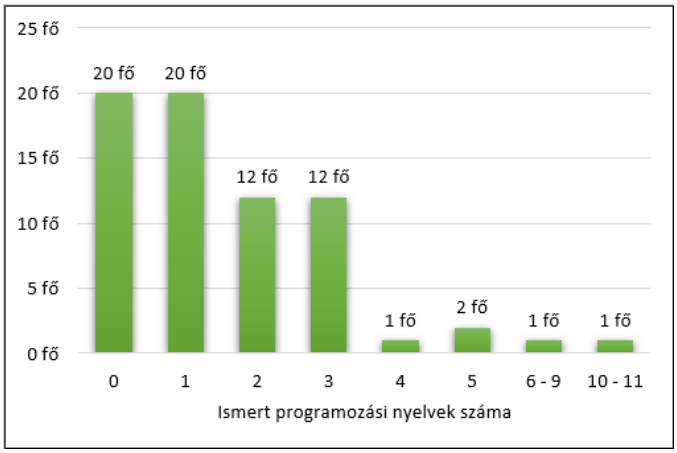
**2. táblázat:** A gyerekek által ismert idegen nyelvek és azok eloszlás.

A 2. táblázatban látható, hogy a diákok többsége, a kitöltők 83%-a, az angol nyelvet tanulja első vagy második idegen nyelvként. Ezt követi, hatalmas különbséggel, a német nyelv, amelyet a kitöltők 25%-a tanul. A válaszok között számos másik nyelv is szerepelt, mint a francia, japán, olasz, orosz és spanyol nyelvek. Ezek közül az oroszt és a spanyolt senki nem tanulta vagy beszélte a megkérdezettek közül, míg a franciát, a japánt és az olaszt 3–4 fő is igyekezett elsajátítani.

Az általunk megkérdezett diákok többsége (83%) az angol nyelvet tanulta első vagy második idegen nyelvként. Ezt követi a német nyelv, amelyet a kitöltők 25%-a tanult. 68%-uk csupán egy idegen nyelvet ismert, 25%-uk kettőt és 3%-uk hármat. Az Oktatási Hivatal eredményei [9] alapján ebben a korosztályban ez a teljesítmény kiemelkedőnek számít.

Az általunk használt LEGO© Mindstorms EV3 eszköznek jelenleg csak angol nyelvű programozói felülete érhető el, azonban ez az angolul nem tudó diákoknak nem szokott problémát okozni. A foglalkozásokon igyekszünk magyar szakkifejezéseket használni, hogy mindenki megértse és könnyebben el tudja sajátítani a szerkezeti elemek használatát, azonban elég gyakran a diákok azok, akik az angol megfelelőket használják.

Megkérdeztük a gyerekeket, hogy tanultak-e programozni mielőtt eljöttek a foglalkozásra. (Az eredmények az 2. ábrán láthatóak.) A résztvevők 29%-a nem tanult eddig semmilyen nyelven programozni.

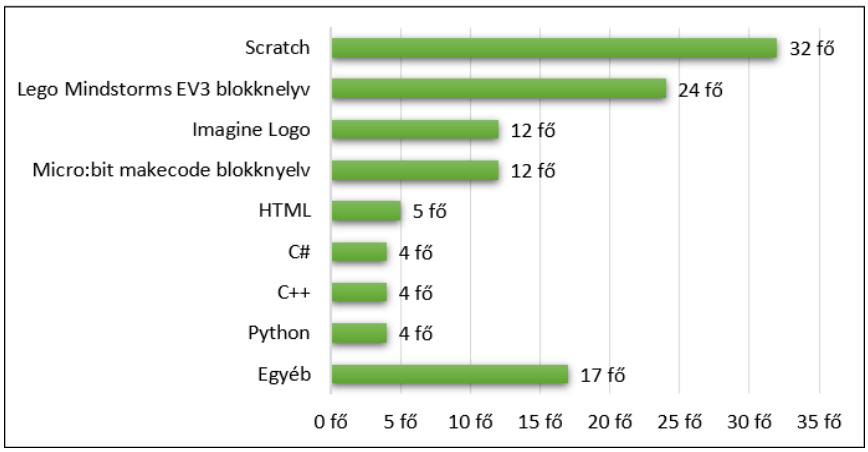



**2. ábra:** A diagramon a gyerekek által tanult programozási nyelvek száma látható.

A gyerekeket ugyan nem kérdeztük arról, hogy milyen keretek között és milyen részletesen foglalkoztak a programozással az általuk ismert nyelveken, azonban ebben az életkorban már az is figyelemre méltó teljesítmény, hogy legalább két programozási nyelvet tanultak. Ez az általunk megkérdezettek 42%-ról elmondható.

Két diákot szeretnék kiemelni, akik saját elmondásuk szerint elég sok programozási nyelvet ismertek, számszerűen az egyikőjük 9-et, a másik pedig 11-et. A kérdőív ugyan anonim volt, de a tábor során nagy érdeklődést mutatott mindkét diák a programozás iránt, akikkel beszélgetve megtudtuk, hogy ők nagyrészt autodidakta módon tanultak meg programozni. Mindketten az Imagine Logo nyelven keresztül ismerték meg a programozás világát és ötleteiket megvalósítva ismerkedtek meg a komolyabb programozási környezetekkel. A diákok az általános iskola végén jártak és határozott elképzeléseik voltak a jövőt illetően, mindketten informatikával szerettek volna foglalkozni.

A kérdőívben a gyerekeknek meg kellett adniuk, hogy ha tanultak programozni, akkor milyen programozási nyelveken. A válaszban több lehetőséget is megjelölhettek, de ha esetleg nem szerepelt valami a listán, akkor kibővíthették. Az általunk megadott nyelvek a következők voltak: Micro:bit makecode blokknyelv, LEGO© WeDo blokknyelv, LEGO© Mindstorms EV3 blokknyelv, Imagine Logo, Scratch, C#, C++, C, Java, JavaScript és HTML. A felsorolásból látható, hogy igyekeztünk minél több területét lefedni a programozásnak, azonban így is maradt néhány részterület, ami kimaradt, de ezt a gyerekek pótolták. Ők a következő nyelvekkel egészítették ki a listánkat: mBlock makecode blokknyelv, PHP, Python, Ruby on rails, Scoolcode, SQL.




**3. ábra:** A diagramon a gyerekek által ismert programozási nyelvek és az adott nyelvet ismerők száma látható.

A 3. ábrán jól látható, hogy a gyerekek által ismert nyelvek közül kiemelkedik négy: Scratch (46%), LEGO© Mindstorms EV3 blokknyelv (35%), Imagine Logo (17%), Micro:bit makecode blokknyelv (17%).

Az egyéb kategóriában azokat a nyelveket csoportosítottuk, amelyeket csupán 1–3 diák tanult. Ezek a nyelvek a következők: Java, JavaScript, LEGO© WeDo blokknyelv, Scoolcode, C, mBlock makecode blokknyelv, PHP, Ruby on rails, SQL.

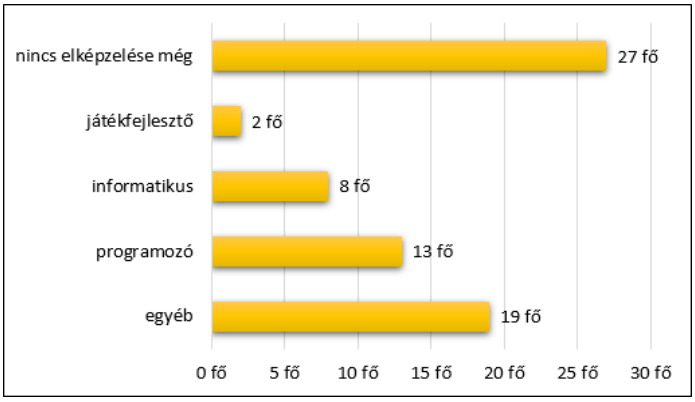
Azok a diákok, akik eddig csak egy programozói nyelvet ismernek (29%), a következő nyelvek valamelyikét használták eddig: C#, Imagine Logo, Java, LEGO© Mindstorms EV3 blokknyelv, LEGO© WeDo, Micro:bit makecode blokknyelv, Scoolcode, Scratch. Ezek közül a legtöbben a EV3 eszköz nyelvét ismerik (12%), amit a Scoolcode és a Scratch követ 4–4%-kal. A felsorolt nyelvek közül a C# és a Java lóg ki, abból a szempontból, hogy ezek nem kimondottan gyerekek számára készült nyelvek és mostanában, már nem is túl gyakoriak első programozási nyelvként. Ez abból

is látszik, hogy az általunk megkérdezettek közül, az egy nyelvet ismerők között csupán 1–1 diák tanulja/tanulta ezeket első programozási nyelvként.

A foglalkozások megkezdése előtt érdemes tisztázni, hogy kinek milyen célja van a foglalkozáson való részvétellel. Ezáltal nemcsak a diákok céljait, motivációit ismerhetjük meg, de számunkra is világossá válik, hogy mire kell nagyobb figyelmet fordítani az egyes foglalkozásokon. Ugyanis, ha többségében az eszközt szeretnék megismerni a gyerekek, mert mondjuk már van egy otthon vagy most szeretnének vásárolni egyet, akkor érdemesebb nagyobb hangsúlyt fektetni az eszközzel kapcsolatos érdekességek kiemelésére és több időt hagyni az eszközzel való önálló kísérletezésre. Azonban, ha a programozás alapjainak az elsajátítása a cél, akkor nagyobb hangsúlyt kell fektetni az elmélet elsajátítására és a robotra, inkább mint, a tanulási folyamatba bevont eszközre kell tekinteni.

Van egy harmadik csoportja is a diákoknak, akik szülői késztetés hatására azért járnak a foglalkozásra, hogy elsajátítsák a számítógép-használat alapjait. Ők többnyire a 9–10 éves korosztályba sorolhatók, akik eddig még okostelefont/tabletet sem használtak és olyan ismeretekkel sem rendelkeznek, mint például, hogyan indíthatnak el egy programot a gépen, melyik a bal és melyik a jobb egérgomb, hogyan kell egy weboldalt megnyitni. Ez alapvetően nem probléma, hiszen valamikor el kell sajátítaniuk ezeket is, viszont erre nem egy programozói szakkör/tábor a legjobb opció. Ugyanis, ha ilyen gyerekek vannak többségben a csoportban, akkor az a programozás tanulásának rovására megy, amire tulajdonképpen beíratták a szülők a gyerekeiket.

A résztvevő gyerekek mindössze 6%-nak az eszköz megismerése volt a célja és 14%-nak a programozás alapjainak elsajátítása. A többi kitöltő, vagyis a gyerekek 80%-a, mindkettőt el szerette volna sajátítani.




**4. ábra:** A gyerekek jövőjére irányuló kérdésre kapott válaszok egy csoportosítása látható a diagramon.

A jövőjüket illető kérdésre elég változatos válaszokat kaptunk, amelyek izgalmas jövőképet mutatrak. A válaszok egy csoportosításáról készült diagram látható az 4. ábrán. A megkérdezett diákok 39%-nak még nem volt konkrét elképzelése a jövőjéről, viszont a többieknek elég konkrét céljuk volt. A gyerekek 33%-a valamilyen informatikai területen szeretne elhelyezkedni, melyet válaszaik alapján három „szakmára" redukálhatunk, melyek a következők: játékfejlesztő, informatikus és programozó. A válaszaikból, az nem derül ki, hogy ismereteik alapján van-e különbség és ha van, akkor milyen különbség van egy informatikus és egy programozó között. A leendő IT-s szakemberek között csupán hárman vannak azok, akik más lehetőséget is felvázolnak munkahelyként, melyek a következők: feltaláló, latin táncos és biológus.

Azok, akiknek volt valamilyen elképzelésük a jövőjüket illetően, de az nem informatikához kötődő tevékenység volt, a 4. ábrán az egyéb kategóriába került. Ők, a következő hivatások közül

választanának: mérnök, kutató, diplomata, vendéglátós, hollywoodi színész, pilóta, fizikus, atomfizikus, asztrofizikus, író, edző, kosaras, kertész, séf, feltaláló.

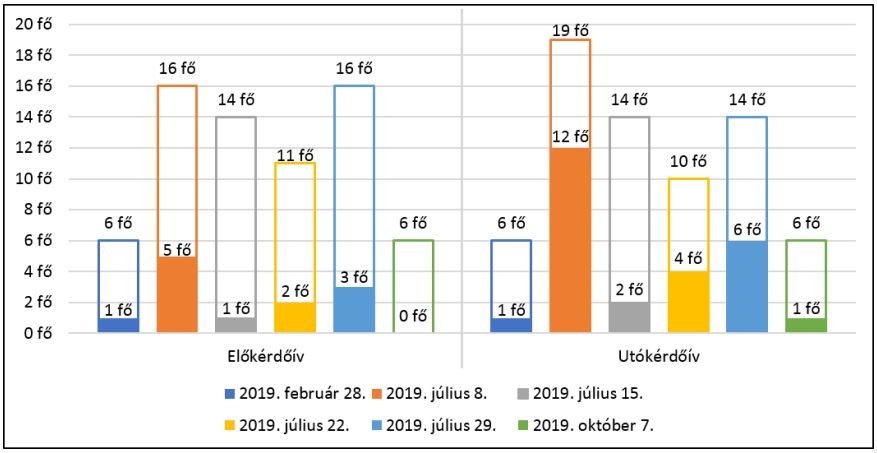
## 4.2. Programnyelvi alapok ismerete

A gyerekek válaszainak kiértékelése előtt utánanéztünk, hogy a tankönyvírók és a tananyagfejlesztők szerint, milyen definíciókat kellene tudnia egy 10–14 éves diáknak a programozás és algoritmizálás témakörein belül. Az alábbi kötetek ehhez kapcsolódó részeit tanulmányoztuk:

- Oktatáskutató és Fejlesztő Intézet által 2019-ben kiadott 6. [11], 7. [12] és 8. [13] osztályosoknak szóló tankönyvei,
- Mozaik Kiadó 2018-ban kiadott 5. [14], 6. [15] és 8. [16] osztályosoknak szóló informatika tankönyvei,
- Pedellus Tankönyvkiadó Kft. Ötödikes informatika c. könyvének 2020-as kiadása [17],
- Oktatási Hivatal Digitális kultúra 5. c. 2020-as megjelenésű informatika tankönyve. [18]

### 4.2.1. A gyerekek fogalmi képe a változóról

A tankönyvi definíciók elég változatos képet adnak arról, hogyan kellene a diákoknak meghatározni a változót. Ugyanis, van olyan kötet, melyben csupán 7 szóban fogalmazzák meg a definíciót [11], de találunk olyat is, amelyben a szerzők több mondatban fejtik ki a definíciót és tulajdonságait [18]. Ezen kívül, találtunk olyan tankönyvet is melyben nemcsak általánosan határozzák meg a változót, hanem különbséget tesznek a változó és globális változó között [15].




**5. ábra:** A foglalkozás elején és végén *Mi az a változó?* kérdésre adott helyes válaszok száma a csoportok létszámához viszonyítva, a foglalkozás kezdete szerinti időrendi sorrendben.

Az előkérdőívben a diákok 15%-a adott elfogadható választ, míg az utókérdőívben a helyesen válaszolók aránya már 32%-ra nőtt. Azt várnánk, hogy a változás egyenletes a csoportokban, de az 5. ábrán láthatjuk, hogy ez nem így van, a július 8-i csoport eredményei több mint duplájára nőttek az utókérdőív válaszainál. Ez a csoport az év első tábori turnusa, amelyben a gyerekek nagy része visszatérő táborozó volt, ami azt jelenti, hogy ők már egy hetet biztosan foglalkoztak a LEGO© Mindstorms EV3 robottal és annak programozásával. Ezen kívül, a diákok átlagéletkora (12 év) is magasabb volt a többi csoporthoz képest, ami olykor hatalmas különbségeket eredményez.[2]

[2] Az életkort nem kérdeztük a kérdőívben, azonban a résztvevők névsora alapján, meg tudtuk határozni.

A foglalkozások típusa szerinti eredmény a következő: a szakköri előkérdőívben 8%, míg az utókérdőívben 17% válaszolt helyesen, a táboroknál ez 19% volt az elején és 42% a végén. Tehát mindkét esetben duplájára nőtt a helyes választ adók száma.

| | **Teljes minta (69 fő)** | | **Szakkörök (12 fő)** | | **Táborok (57 fő)** | |
|---|---|---|---|---|---|---|
| | **Elején** | **Végén** | **Elején** | **Végén** | **Elején** | **Végén** |
| **Átlag** | 15% | 32% | 8% | 17% | 19% | 42% |
| **Szórás** | 10% | 18% | 6% | 0% | 9% | 17% |

**3. táblázat:** A változóról adott helyes definíciók aránya és százalékos eloszlása.

A 3. táblázatot elemezve, az adatok pozitív irányú változását figyelhetjük meg, azonban ez a változás kis mértékű, így arra következtethetünk, hogy a definíció meghatározása még nehézséget okoz a diákoknak. A változó fogalma egy nehezen emészthető fogalom (a gyerekeknek), mert kevésbé hétköznapi kifejezés, mint a ciklus vagy az elágazás. Ezen kívül, legfeljebb matematika órán találkozhatnak a kifejezéssel, ami (sajnos) a gyerekek által kevésbé kedvelt tantárgyak között szerepel [19]. Továbbá, figyelembe kell vennünk az értékelésnél a gyerekek életkorát, ami a teljes mintát tekintve átlagosan 12 év (pontosan 11,7). Matematika órán az ismeretlent, a változót, és ezáltal az egyenleteket (jobb esetben) 6. osztály végén tanulják, de többnyire inkább 7. osztályban, ami azt jelenti, hogy 12–13 évesen. Ilyenkor is nehézséget okoz az ismeretlen értelmezése, viszont 9, 10 vagy 11 évesen, még nehezebb.

A következőkben néhány, a diákok által írt és elfogadott definíciót közlünk.

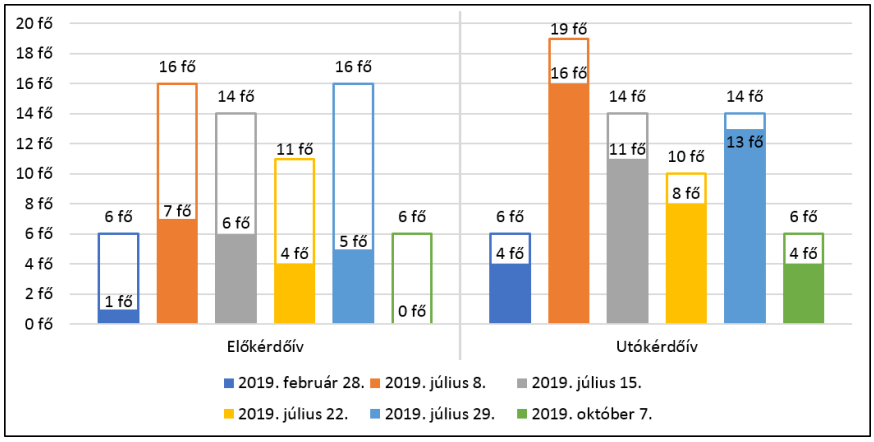
- „Egy memóriaszelet, ahol adatokat tárolunk és névvel azonosítjuk."
- „Változónak nevezünk a programozási nyelvekben egy olyan azonosítót (szót), amivel valamilyen adatra, vagy objektumra hivatkozunk."
- „Egy adattároló egység."
- „Egy érték, amit a program elején lehet megadni, így mondjuk a programmal létrehozott alakzat méreteit lehet változtatni."
- „Egy tárolóhely, amely megőrzi az információt, így azt egy programban többször is felhasználhatjuk."
- „Egy dolog, aminek a neve állandó, de az értéke változó."
- „Egy szám, szöveg vagy objektum, aminek neve van és értéke változtatható."
- „Csak a példára emlékszem: van egy doboz, ami örökké van, de a benne lévő dolgok cserélődnek."
- „Egy olyan adat, amelynek változik az értéke."

### 4.2.2. A gyerekek fogalmi képe a ciklusról

A ciklus elég sok tankönyvben megjelenik, azonban van, ahol ismétlésként definiálják [14] [15] [16] [17] és a ciklus kifejezést kevésbé említik, és van olyan is, ahol külön definiálják a hátultesztelő, számlálós és végtelen ciklusokat [18].

A tankönyvek szerint a 10-14 éves gyerekeknek ismernie kellene az ismétlés/ismétlési szerkezet, ciklus, hátultesztelő ciklus, számláló ciklus, végtelen ciklus vagy feltételes ciklus definíciók valamelyikét. Az idősebbeknek, pedig az összeset meg kellene tudniuk különböztetni. Ennek ellenére a gyerekek többsége a ciklusról az ismétlésre asszociált és ezt próbálta saját szavaival megfogalmazni. A ciklus különböző típusainak említése pedig keveseknek fordult meg a fejében. A dolog hátterében feltételezéseink szerint az áll, hogy több programozási nyelvben ismétlésként van definiálva a ciklus és a gyakorlati feladatmegoldás során az rögzül, hogy ismétlést használunk, és nem az, hogy ciklust használunk. Ilyen programozási nyelv a Scratch, a Micro:bit Makecode blokknyelv és az Imagine

Logo is. Az előkérdőívben 33% adott elfogadható leírást a ciklusra (6. ábra), melyből 83% az ismétlést írta le valamilyen formában. Ebből 52% a korábban említett három nyelv valamelyikét (vagy mindegyikét) már ismerte. (Egy diák csupán az Imagine Logot ismerte, rajta kívül, viszont mindenki programozott már Scartch-ben.) Tehát a helyes választ adó diákok 52%-a olyan nyelvben programozott eddig, melyben az ismétlés vagy ismételd nevű parancsot használták ciklusként.




**6. ábra:** A gyerekek foglalkozás elején és végén a *Mit nevezünk ciklusnak?* kérdésre adott helyes válaszainak száma a csoportok létszámához viszonyítva, a foglalkozás kezdete szerinti időrendi sorrendben.

Az előkérdőívben a diákok 28%-ának, míg az utókérdőívben már 78%-ának tudtuk elfogadni a válaszát, ami hatalmas változást jelent. Az utókérdőívben csupán két választ nem tudtunk elfogadni, a többi diák semmilyen választ nem adott. A 6. ábrán is jól látható, hogy a helyes válaszok számának növekedése egyenletesebb, nincs olyan kiugró érték, mint a változó esetében.

A csoportok típusát tekintve a következő eredmények születtek: a szakkörök elején a gyerekek mindössze 8%-a adott helyes választ, míg a foglalkozás végi kérdőívben már 67%-uk. A táborok elején a táborozók 39%-a tudta definiálni a ciklust, míg a tábor végén már 84%-uk. Ebből arra következtethetünk, hogy a ciklus egy könnyen megérthető fogalom, amit megerősít az 4. táblázatban lévő eredmény is. A gyerekek átlagos helyes válaszainak száma a ciklusnál majdnem duplája a változó eredményeinek. Ez azzal is magyarázható, hogy a ciklust szinte minden foglalkozáson használtuk, a változót viszont legfeljebb az utolsókon. Annak ellenére, hogy az utókérdőív kitöltéséhez közelebbi időpontban definiáltuk a változót, nem tudott annyira jól rögzülni, mint a ciklus.

| | **Teljes minta (69 fő)** | | **Szakkörök (12 fő)** | | **Táborok (57 fő)** | |
|---|---|---|---|---|---|---|
| | **Elején** | **Végén** | **Elején** | **Végén** | **Elején** | **Végén** |
| **Átlag** | 28% | 78% | 8% | 67% | 39% | 84% |
| **Szórás** | 16% | 9% | 8% | 0% | 5% | 6% |

**4. táblázat:** A ciklusról adott helyes definíciók számának aránya és százalékos szórása.

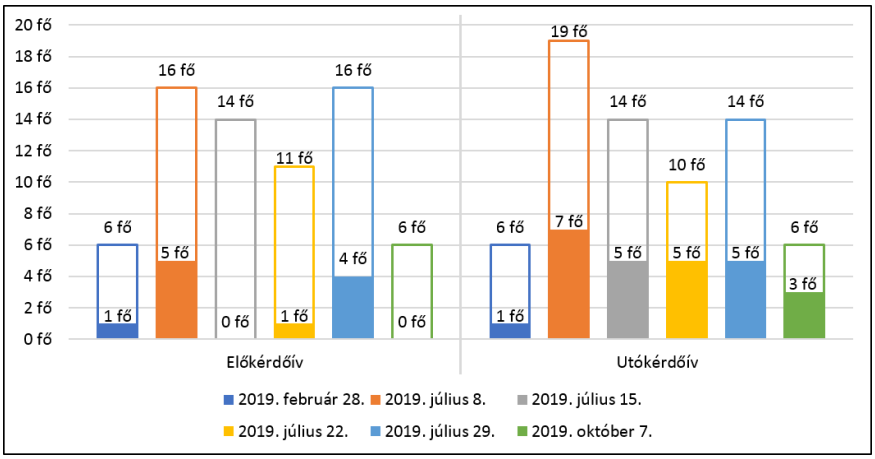
A gyerekek a következőképpen definiálták a ciklust.

- „Egy programrész, ami folyton ismétlődik."
- „Ismétlődő függvény."
- „A robot egy "cselekvésének" a többszöri elismétlése."
- „Egy folyamat, ami ismétlődik."

- „A ciklus vagy egyszer vagy többször meg tud ismételni egy vagy több utasítást.”
- „Akár rövidíteni is tudjuk a programoknak annak a részét, ahol ismételni kell valamit. Be is lehet állítani, hogy hányszor ismételje (akár végtelenszer) a kívánt eredmény elérése érdekében.”
- „Olyan parancs, amely a benne lévő dolgokat annyiszor ismétli meg, amennyiszer azt mi szeretnénk.”
- „A robot egy cselekvésének valahányszori megismétlésére szolgáló "eszköz".”
- „Ismétlődő folyamat.”
- „A loopot.”

### 4.2.3. A gyerekek fogalmi képe az elágazásról

A definíciót sokféleképpen meg lehet fogalmazni, azonban az biztosan szerepel benne, hogy legalább kétféleképpen folytatódhat a programunk, ill. az, hogy a folytatás kiválasztásának módja egy feltételhez van kötve. A gyerekek válaszainak értékelésekor mi is ezeket vettük figyelembe, és ha a korábbi megfogalmazásunk valamelyik részét tartalmazta a gyerekek leírása, akkor elfogadtuk a válaszaikat. Az előkérdőívben nem volt túl sok értékelhető válasz, amiket leírtak a gyerekek az sem felelt meg az általunk felállított kritériumoknak. Amint a 7. ábrán is látható, az előkérdőívben nem születtek túl jó eredmények, ekkor a gyerekek 14%-a adott elfogadható választ. Az utókérdőívben már a megkérdezettek 37%-a tudta megfogalmazni saját szavaival azt, hogy mi az elágazás, ami javulást jelent az előkérdőívhez képest.




**7. ábra:** A gyerekek foglalkozás elején és végén a *Mit nevezünk elágazásnak?* kérdésre adott helyes válaszainak száma a csoportok létszámához viszonyítva, a foglalkozás kezdete szerinti időrendi sorrendben.

Az elágazás használatát a táborozók a 4. napon sajátítják el, míg a szakkörön résztvevők az 5. foglalkozáson. Érdekesebb eredmény a szakköröknél született, a 2019.február 28-i csoportban nem változott a helyes választ adók száma, azonban a 2019.október 7-i csoportnál jelentős változást értünk el. Az eredmények alapján azt feltételeznénk, hogy az októberi csoportban több idősebb diák volt és emiatt lett jobb az eredmény, azonban ez a feltételezés nem helytálló. A februári csoport átlagéletkora 12, míg az októberié 11 év. A névsorok alapján tudjuk, hogy az első csoportban 3:3, míg az utolsóban 1:5 volt a lányok és fiúk aránya, ami a sztereotípiák szerint magyarázat lenne az eredményekre, azonban csak női oktatókkal a hátunk mögött, itt mi nem állhatunk meg. Emlékeink szerint a legnagyobb különbség a gyerekek motiváltságában volt. Míg az első csoportban egy–két diák volt, akit igazán érdekelt a programozás és a robotika, addig a másik csoportban legfeljebb egy diák volt kevésbé motivált. Ez az, ami a nagy különbséget eredményezi, hiszen nem mindegy, hogy a

diákok azért jönnek, hogy új barátokat szerezzenek, jókat beszélgessenek vagy azért, hogy az új barátokkal érdekes eszmecserét folytassanak a saját fejlesztésű Scratch játékaikról.

A csoportok típusa szerinti bontásban az eredmények a következők (5. táblázat): a szakkörön résztvevők 8%-a válaszolt helyesen az előkérdőívben, míg a táborokban a résztvevők 18%-a. Az utókérdőívben a szakköröknél már 33%-uk, a táboroknál 39%-uk válasza volt elfogadható. A fejlődés hasonló mértékű mindkét csoportban, a szakkörnél 25%, míg a táborban 21%. A gyerekek átlagos helyes válaszainak aránya elég kevés, a korábban is említett változó eredményeihez hasonló.

| | **Teljes minta (69 fő)** | | **Szakkörök (12 fő)** | | **Táborok (57 fő)** | |
|---|---|---|---|---|---|---|
| | **Elején** | **Végén** | **Elején** | **Végén** | **Elején** | **Végén** |
| **Átlag** | 14% | 37% | 8% | 33% | 18% | 39% |
| **Szórás** | 12% | 11% | 8% | 17% | 12% | 6% |

**5. táblázat:** Az elágazásról adott helyes definíciók számának aránya és százalékos szórása.

A gyerekek válaszainak elemzésekor olyan megfogalmazásokkal is találkoztunk, melyben a gyerekek megmagyarázni akarták az elágazás kifejezését (útkereszteződésként) és nem definiálni a programozás során használt elágazás szerkezetet. Többen is ezzel próbálták megmagyarázni az elágazást, ami részben helyes magyarázat, hiszen a hétköznapi beszélgetések során az útkereszteződésekre használjuk az elágazás kifejezést, azonban itt ezeket nem tudtuk elfogadni. Ehhez hasonló okok miatt el kellett utasítanunk a következő válaszokat is:

- „Amikor elágazik a program."
- „Egy olyan parancs, amely után a program több irányba válik szét."
- „Amikor a program több részre lesz bontva."
- „Amikor egyszerre kettő program van egy ciklusban."

A korábban tárgyalt definíciók alapján elfogadásra kerültek a következő meghatározások.

- „Ahol a program két vagy több különböző irányban tud folytatódni."
- „A program azt a részét, amely egy feltételtől [if else] függően fog folytatódni. True False"
- „Egy programnak azt a részét, amelyben a program attól függően fog folytatódni, hogy adott feltétel(ek) érvényesül(nek)-e."
- „Két program közül az egyiket futtatja."
- „Egy, kettő, vagy több lehetőségből álló eldöntendő kérdést."
- „Ha... tejesül akkor robotunk.... de ha nem akkor..."
- „Amikor a program folytatása más–más feltételektől függ, így különbözhet a két végkimenetel."
- „Egy olyan pont a programban, ahol van egy feltétel és annak megfelelően folytatódik."
- „Ha egy feltétel igaz, akkor ezt csinálja, ha meg nem, akkor mást."

## 4.3. Az eredmények összegzése

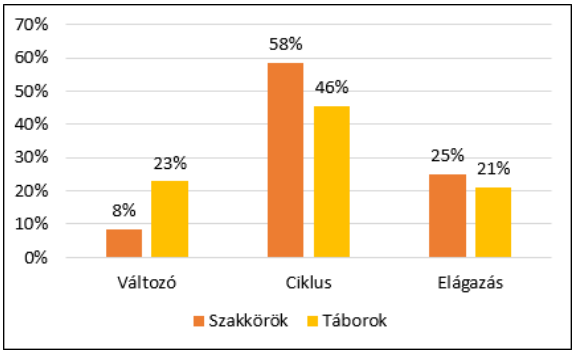
A megkérdezett diákok 71%-a érkezett úgy a foglalkozásokra, hogy már programozott valamilyen nyelven. Az általuk adott helyes válaszok százalékos aránya látható a 6. táblázatban. Megfigyelhető, hogy a legnagyobb változást náluk is a ciklus definiálásánál értük el, ami 40%-os növekedést jelent. A legkevesebb változást pedig az elágazás definiálásánál és nem a változónál. Azonban, ha a teljes mintát nézzük, akkor más eredményeket kapunk. Az egész csoportot tekintve szintén a ciklus meghatározásánál értük el a legnagyobb fejlődést, ami 48%-os növekedést jelentett, azonban a változó és elágazás fogalmaknál, ennek a növekedésnek csupán a felét értük el.

| | Helyes válaszok számának aránya az előzetes tudással rendelkezők lét-számára nézve | | | Helyes válaszok számának aránya az egész csoport létszámára nézve | | |
|---|---|---|---|---|---|---|
| | **Változó** | **Ciklus** | **Elágazás** | **Változó** | **Ciklus** | **Elágazás** |
| **Előkérdőív** | 22% | 42% | 20% | 17% | 33% | 16% |
| **Utókérdőív** | 45% | 82% | 35% | 38% | 81% | 38% |

**6. táblázat:** A táblázatban a helyes válaszok aránya látható az előzetes tudással rendelkezők ill. az összes kitöltő létszámára nézve.

A tananyag jelenlegi feldolgozási sorrendje alapján, azt feltételeztük, hogy a változó fog az utolsó helyen végezni minden szempontból, ugyanis azt ismerik meg a legkésőbb a gyerekek és emiatt kevés idejük van használni, elmélyíteni a megtanultakat. Ennek a feltételezésnek egy másik oka a diákok közötti nagy korkülönbség, ami a változó definiálásakor szokott problémát okozni, ugyanis a kisebbek azt mutatják, hogy értik a magyarázatot, azonban az önálló feladatmegoldás során nem boldogulnak egyedül. Emiatt a kicsikkel több példát kell megnézni vagy más kontextusba kell helyezni számukra a definíciót, amivel ők tanulnak, azonban az idősebbek fejlesztésére már kevesebb idő jut.

Ha a foglalkozások típusa szerinti bontásban nézzük az eredményeket, akkor azt vehetjük észre, hogy a táborokban résztvevő gyerekek több jó megoldást adtak, mint a szakkörön résztvevők. (8. ábra)




**8. ábra:** A diagramon a helyes választ adók számának változása látható.

A rendelkezésünkre álló adatok alapján megállapítható, hogy a fejlődés mértéke nagyobb a szakkörön az általunk vizsgált három definíció közül kettőnél (ciklus, elágazás). Ezzel szemben, a változó definiálásánál, lévén hogy a szakköri foglalkozásokon csak érintőlegesen volt szó erről, a fejlődés a vártnak megfelelően alacsonyabb.

Az eredmények egyértelműen azt mutatják, hogy a ciklus használatát könnyen elsajátítják a gyerekek és ezáltal definiálni is jobban tudják, mint az elágazást vagy a változót. Viszont, a foglalkozásokon az a célunk, hogy a gyerekek a programozás alapjait sajátítsák el, amibe beletartozik az utóbbi két definíció ismerete és használata is, így érdemes átgondolnunk a tananyag újrafelosztásának lehetőségét. Ezen kívül egy másik opció lehet az, ha növeljük a foglalkozások számát vagy hosszát, amivel biztosítani tudjuk, hogy több tapasztalatot szerezzenek a gyerekek az elágazások és változók használatában.

A definíciók ismeretében nagyobb fejlődést érhetünk el akkor is, ha korban egymáshoz közel álló gyerekeket teszünk egy csoportba. Ezzel, a gyerekek tudásukban és befogadóképességükben is hasonló szinten lesznek, ami kulcsa lehet a gyorsabb, hatékonyabb ismeretátadásnak.

## 5. Konklúzió

A kérdőíves kutatásban a gyerekek teljesítményét vizsgáltuk programozás tanulás során, szakköri és tábori közegben. A rendelkezésünkre álló adatok alapján, arra a megállapításra jutottunk, hogy a szakköri foglalkozásokon nagyobb arányú fejlődés érhető el, mint a táborokban. Azonban szükségesnek tartjuk a szakkörök idejének növelését ahhoz, hogy mindhárom definícióra megfelelő mennyiségű idő jusson.

A kutatás megválaszolta néhány kérdésünket, azonban nem kaptunk átfogó képet a foglalkozáson résztvevő gyerekek fejlődéséről. Ehhez, egy olyan kutatásra lenne szükség, melyben azonosítjuk az egyéneket és megvizsgáljuk kor, nem és előzetes tudás alapján is a gyerekek fejlődését. Ezzel megvizsgálhatnánk az előzetes tudással rendelkezők fejlődését is, valamint az életkoruk segítségével meghatározhatnánk azt is, hogy rendelkeznek-e a Nemzeti Alaptantervben elvárt ismeretekkel.

A vizsgálat további szempontja lehet, hogy befolyásolja-e a gyerekek programozás tanulási hatékonyságát a modern, de barátságos és befogadó környezet. Az általunk vizsgált foglalkozásokon az oktatók nem ragaszkodtak az iskola merev szabályaihoz, inkább tutori szerepet töltöttek be és az oda–vissza irányú tegeződést preferálták. A foglalkozások helyszínéül az élményszerű légkört biztosító T@T Kuckó szolgált. Ezzel szemben az iskolai szakkörökön, ugyanúgy ragaszkodnak a merev tanár–diák szerepekhez, mint a délelőtti tanórák alatt és a légkör megteremtésében fontos tényező az intézmények anyagi körülményei és szabályai.

## Irodalom


1. J. M. Wing, „Computational thinking," *Communications of the ACM,* pp. 33-35, 2006.
2. Angeli, Charoula and Giannakos, Michail, „Computational thinking education: Issues and challenges", *Elsevier*, 2020., DOI: 10.1016/j.chb.2019.106185
3. J. M. Wing, „Computational Thinking: What and Why?," 2010.
4. L. Werner, J. Denner, S. Campe, D. C. Kawamoto, „The fairy performance assessment: measuring computational thinking in middle school," in *Proceedings of the 43rd ACM technical symposium on Computer Science Education*, 2012, pp. 215-220.
5. Cynthia. C. Selby, „Relationships: computational thinking, pedagogy of programming, and Bloom's Taxonomy," in *Proceedings of the workshop in primary and secondary computing education*, 2015, pp. 80-87.
6. Pérez-Marín, D., Hijón-Neira, R., Bacelo, A., & Pizarro, C., „Can computational thinking be improved by using a methodology based on metaphors and scratch to teach computer programming to children?," *Computers in Human Behavior,* 2020., DOI: 10.1016/j.chb.2018.12.027
7. International Society for Technology in Education (ISTE), „Operational Definition of Computational Thinking for K-12 Education," 2011., https://id.iste.org/docs/ct-documents/computational-thinking-operational-definition-flyer.pdf. (utoljára megtekintve: 2020.10.26.)
8. Solymos Dóra, „LEGO robotok felhasználási lehetőségei az oktatásban," *InfoDidact 2019,* pp. 265-275, 2019.

9. dr. Öveges Enikő és dr. Csizér Kata, „Vizsgálat a köznevelésben folyó idegennyelv-oktatás kereteiről és hatékonyságáról," 2018., (utoljára megtekintve: 2020.10.26.) https://www.oktatas.hu/pub_bin/dload/sajtoszoba/nyelvoktatas_kutatasi_jelentes_2018.pdf.
10. „Magyar Közlöny," 2014, (utoljára megtekintve: 2020.10.26.), http://www.kozlonyok.hu/nkonline/MKPDF/hiteles/MK14004.pdf
11. Regele György, Ridzi Gizella, Rajk Ágnes, Lakosné Makár Erika, Informatika 6., Oktatáskutató és Fejlesztő Intézet (OFI), 2019, (utoljára megtekintve: 2020.10.26.) https://www.tankonyvkatalogus.hu/pdf/NT-11682__teljes.pdf
12. Regele György, Ridzi Gizella, Rajk Ágnes, Lakosné Makár Erika, Informatika 7., Oktatáskutató és Fejlesztő Intézet (OFI), 2019. https://www.tankonyvkatalogus.hu/pdf/NT-11782__teljes.pdf, (utoljára megtekintve: 2020.10.26.)
13. Regele György, Ridzi Gizella, Rajk Ágnes, Lakosné Makár Erika, Informatika 8., Oktatáskutató és Fejlesztő Intézet (OFI), 2019., https://www.tankonyvkatalogus.hu/pdf/NT-11882__teljes.pdf (utoljára megtekintve: 2020.10.26.)
14. Dr. Kokas Károly, Rozgonyi-Borus Ferenc, Informatika 5., Szeged: Mozaik Kiadó, 2018.
15. Rozgonyi-Borus Ferenc, Informatika 6., Szeged: Mozaik Kiadó, 2018.
16. Dr. Kokas Károly, Rozgonyi-Borus Ferenc, Informatika 8., Szeged: Mozaik Kiadó, 2018.
17. Fenyősné Kircsi Amália, Fenyős Zoltán, Ötödikes informatika, Debrecen: Pedellus Tankönyvkiadó Kft., 2020.
18. Lénárd András, Abonyi-Tóth Andor, Turzó-Sovák Nikolett, Varga Péter, Digitális kultúra 5., Oktatási Hivatal, 2020, https://www.tankonyvkatalogus.hu/pdf/OH-DIG05TA__teljes.pdf (utoljára megtekintve: 2020.10.26.)
19. Csapó Benő, „A tantárgyakkal kapcsolatos attitűdök összefüggései," *Magyar Pedagógia,* pp. 343-366, 2000.

# A Tinkercad Arduino-szimulátor alkalmazása az online programozásoktatásban

Somogyi Anikó[1], Makan Gergely[2], Kelemen András[3], Mingesz Róbert[4]

{[1]somogyia, [2]makan, [4]mingesz}@inf.u-szeged.hu;
[1,2,4] SZTE TTIK Műszaki Informatika Tanszék
[3]kelemen.andras.felix@szte.hu
[3] SZTE JGYPK Informatika Alkalmazásai Tanszék
[1,3] Szegedi Radnóti Miklós Kísérleti Gimnázium

**Absztrakt.** A COVID-19 világjárvány miatti digitális átállás az informatikaoktatást is közvetlenül érintette. Az online munkaforma a műszaki informatikai jellegű tárgyak oktatásában különleges kihívást jelent, hiszen az intelligens elektronikai eszközök programozásának tanítása alapvetően laboratóriumi környezetben valósul meg. Azonban a rendelkezésre álló szimulátorprogramok lehetőséget nyújtanak, hogy a képzés során a megkövetelt elektronikai és informatikai ismereteket a hallgató vagy diák elsajátítsa akár online oktatás során is. A tanulmányban bemutatjuk a regisztráció után ingyenesen használható online szimulációs programcsomagot, a Tinkercadet. A felületen elektronikai áramkörök valósághű modellezése, a virtuális Arduino UNO-hoz áramköri elemek, szenzorok, aktuátorok csatlakoztatása lehetséges, és az eszköz szöveges kódolással (C/C++) vagy blokk alapú környezetben is programozható böngészőben. A platform a frontális oktatás alternatívájaként lehetővé teszi az egyéni és kiscsoportos tanulói aktivitást is a köz- és felsőoktatásban egyaránt. Munkánkban megosztjuk oktatási tapasztalatainkat a programozók és informatikatanárok képzésében, továbbá ismertetünk néhány konkrét gyakorlatot, amelyet a hallgatóink megvalósítottak a digitális oktatás során.

**Kulcsszavak:** műszaki informatika, Arduino, online oktatás

## 1. Bevezetés

A Szegedi Tudományegyetem Természettudományi és Informatikai Karának Műszaki Informatika Tanszékén több olyan kurzust is oktatunk, amelyen intelligens elektronikai eszközök programozása témakörben az Arduino-programozás hangsúlyos szerepet kap. Tanárszakos hallgatóink számára akkreditáltunk egy kurzust, Az informatika műszaki alkalmazásai címmel, melynek keretén belül a hallgatók többek között interdiszciplináris laborgyakorlatokat hajtanak végre. Korábban publikáltuk a gyakorlatokon előforduló feladatok tartalmát, céljait és tapasztalatainkat [1,2]. Az utóbbi években több alapozó kurzusunk tananyagában (Műszaki alapismeretek, Elektronika alapjai programozóknak) is szerepelnek az Arduino platformmal kapcsolatos alapismeretek, sőt villamosmérnök hallgatóink tehetséggondozó óráin is hangsúlyos szerepet kapnak.

A 2019/20-as és 2020/21-es tanévek új kihívásokat támasztottak a tárgy oktatói elé. A SARS-COV19 vírus terjedése miatti korlátozások a megszokott oktatási formákat is felülírták. Mivel ezek a gyakorlatok alapvetően laboratóriumi környezetben, mérőeszközökkel, folyamatos tanári konzultációval zajlottak korábban, így nem kis kihívást jelentett a műszaki tartalmú gyakorlati anyag áthelyezése az online térbe. A távoktatás az Arduino esetében otthon elvégzett gyakorlatokkal és párhuzamos online konzultációval egészen jól megoldható, ha a hallgató beszerzi a szükséges eszközkészletet, ugyanakkor erre nem kötelezhető [3]. A gyakorlatainkon ezért a fizikai eszközöket virtuális műszerekre cseréltük, és a laborfeladatokat átalakítottuk szimulációs feladatokká. A különböző szimulációs programok programozásoktatásban való hasznosítására jó példákat találhatunk a szakirodalomban is [4], sőt a pandémia miatti lezárások alatt más felsőoktatási intézmények is alkalmazták ezeket műszaki tartalmú gyakorlataik oktatása során [5]. Fontos, hogy a szimulációs feladatok segítségével

ugyanazokat a tudáselemeket lehessen elsajátítani, hiszen a hallgatónak később ezeket valódi eszközökön is kell tudnia alkalmazni. Hasznos továbbá, ha a használt környezet valósághű, tartalmazza a gyakorlatok elvégzéséhez szükséges szenzorok jó részét, vagy legalábbis alternatív eszközökkel megoldhatók.

Az Arduino népszerűségének és széles körben való elterjedésének köszönhetően különböző szintű Arduino-szimulátorokból szerencsére nincs hiány [6,7]. A szimulátorok jó része nem oktatási célra, hanem prototípuskészítésre, tesztelésre készültek. A mi választásunk az Autodesk nagyvállalat Circuits nevű alkalmazására esett, amely Tinkercad honlapon lévő szoftvercsomag egyik alkalmazása. Ezt a programot komoly oktatási portálok [8,9] és az Arduinót népszerűsítő honlapok is ajánlják, bár említik a korlátait is [10,11].

A szoftver oktatásban való alkalmazásával kapcsolatban a tanulmány szerzőinek is volt korábbi tapasztalata, hiszen a MTA-SZTE Műszaki Informatika Szakmódszertani Kutatócsoport (MISZAK) által fejlesztett oktatóanyagok, példaalkalmazások és videóleckék egy része ezzel a programmal készültek [12,13].

# 2. A Tinkercad Circuits szimulátor

## 2.1. A Tinkercad platform általános bemutatása

A Tinkercad [14] platform egy 3D-modellezésre alkalmas, egyszerűen kezelhető, ingyenes, online CAD-szoftver 2010 óta érhető el a felhasználók számára. 2013-ban a későbbi tulajdonos, az Autodesk bővítette a Circuits ('áramkörök') alkalmazással, míg napjainkban a CodeBlocks nevű, programozható, LEGO-elemeket is tartalmazó 3D-tervezőprogrammal [15]. A magyar nyelven is elérhető szoftverek alkalmasak iskolai konstruktív geometria (pl. STEM projektekben 3D-nyomtatáshoz exportálható tervek készítése) elektronika és programozás oktatásban történő felhasználásra [16].

## 2.2. A Tinkercad biztosította lehetőségek az oktatásban

A Tinkercad fejlesztői számos, kifejezetten oktatási célú eszközt építettek be a platformba. A kidolgozott adatvédelmi protokollal rendelkező honlapon elkülönülnek a tanár/oktató és a diák/hallgató felhasználók. A diákokat osztályokként (Classroom) fejleszthetik a kollégák, melyhez kész, jól kidolgozott óraterveket biztosít a honlap. A Tinkercad Circuits jelen van az Arduino-projektekben is bővelkedő Instructables honlapon, YouTube-csatornájukon rendszeresen tesznek közzé oktatóvideókat, de jól reprezentálják magukat a közösségi média felületein is, ahol folyamatosan értesülhetnek a felhasználók az elérhető újdonságokról [17,18].

## 2.3. Fájlkezelés a Tinkercadben

A Tinkercad Circuits alkalmazásra kattintva automatikusan létrejön és mentésre kerül egy új fájl (ún. terv) egy fantázianévvel, és bekerül a saját gyűjteményünkbe. A főoldalra visszalépve csempeszerű elrendezésben látjuk valamennyi korábbi fájlunkat, a legfrissebbel kezdve időrendi sorrendben. A csempe jobb felső sarkára kattintva beállíthatók bizonyos tulajdonságok, mint például a terv neve, láthatósága (privát vagy nyilvános), de itt lehet megkettőzéssel klónt készíteni a tervről.

Lehetőség van a fájlok projektekbe rendezésére is. Egy projekt létrehozása után visszatérve a Circuits gyűjteményünkbe, a terv beállításai között található Ugrás projektre címszónál hozzárendelhetjük a kívánt projekthez. Hasznos, hogy a 3D tervek és a Kódblokkokban készült tervek is közös projektbe rendezhetők a Circuits-tervekkel, így egy helyen kezelhetők összetett, nagyobb oktatási projektek tervei.

A létrehozott fájlok alapértelmezésben mindig privát típusúak, viszont a fájl megnyitása után a Megosztás opciót választva van lehetőség Emberek meghívására: a link ismeretében más is megnyithatja, módosíthatja a tervet. Van lehetőség a terv nyilvánossá tételére is, ez esetben egy a Tinkercad-közösség számára elérhető, kereshető fájl jön létre. Az ilyen publikus tervek remixelhetők, tovább-

fejleszthetők mások által, és természetesen mi is felhasználhatjuk más felhasználók nyilvános munkáit.

## 2.4. Tinkercad által támogatott oktatási formák

Az itt leírtak alapján többféle munkaformában alkalmazhatók ezek a Tinkercad-tervek az oktatásban. Egyrészt lehetséges frontális (akár online) előadásokon „élőben" szimulálni egy-egy kísérletet sokkal egyszerűbben, mint fizikai eszközt kamerázva. Előre elkészített online tananyagokhoz pedig egy képernyőrögzítésre alkalmas programmal meglehetősen élethű, szép szimulációk készíthetők.

Egy másik természetesen adódó munkaforma az önálló, egyéni munka, amelyre online gyakorlati órákat is lehet felépíteni. Itt a hallgatónak adhatunk olyan feladatot, amelyben teljes mértékben neki kell elhelyezni a szükséges eszközöket, áramköri elemeket a tervben, majd összeállítani az áramkört és azt később programozni. Az is lehetséges, hogy előre elkészített áramköri részeket, esetleg kódrészletet hozunk létre, és ezt nyilvánossá téve a hallgató remixelheti a tervet, és azt saját terveként továbbfejlesztheti.

Egy harmadik, az online oktatás során szintén hasznos munkaforma, hogy a hallgatók tudnak távolról ugyanazon a terven dolgozni, így kooperatív, kiscsoportban végzett feladatmegoldásra is van lehetőség.

## 2.5. A Tinkercad áramkörmodellező felülete

A Tinkercad áramkörépítő grafikus felületén „fogd és vidd" (drag & drop) típusú szerkesztési lehetőséget kap a felhasználó: az eszköztárból kiválasztott absztrakt objektumot (áramköri elem) a számítógép egere segítségével behúzhatunk a grafikus felület megfelelő helyére. Az áramköri elemeket (pl. ellenállások, diódák, tranzisztorok, áramforrások) szintén az egér segítségével köthetjük össze virtuális vezetékekkel így hozhatunk létre zárt áramköröket. Szerelőlap néven implementálták a próbapanelt (breadboard), amely segítségével a terveink átláthatóan összeállíthatók, vezetékelhetők.

Az eszközök között találhatunk Arduino UNO modellt, és hozzá számos szenzort, kiejelzőket, motorokat, továbbá jó néhány integrált áramkört, drivert is implementáltak. Bizonyos alap elektronikai laboreszközök (multiméter, labortápegység, oszcilloszkóp, tesztgenerátor) modelljei is elérhetők.

## 2.6. A Tinkercad fejlesztőkörnyezete

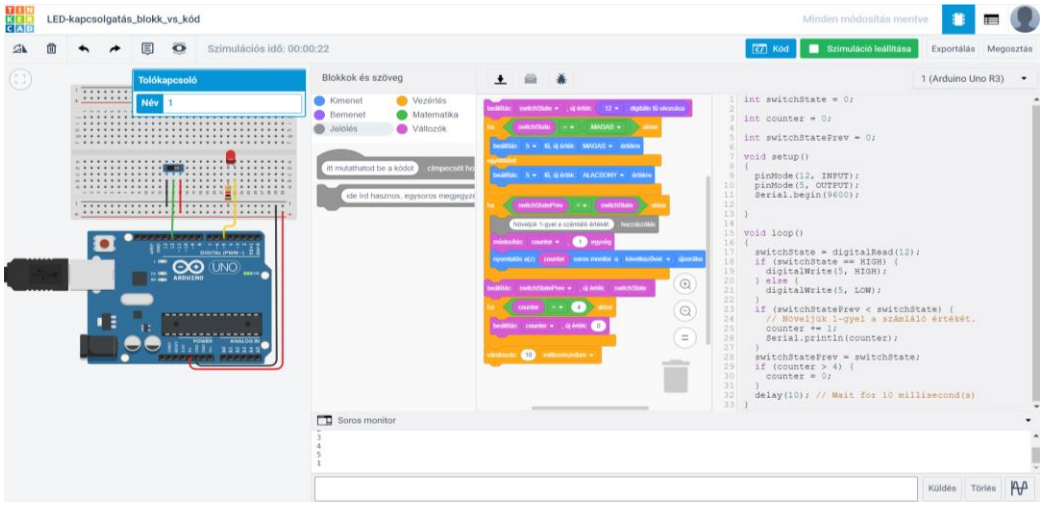
A Tinkercad jelenlegi verziójában több programozható eszköz található, az Arduino UNO R3 modellje, az ATtiny és a BBC micro:bit. Miután az eszközt elhelyezi a felhasználó a CAD-felületen, elérhetővé válik számára a Kód felület, ahol megkezdhető az eszköz programozása. Háromféle módban történhet a kódolás: a kezdő programozók számára a Scratchre [19] emlékeztető blokk alapú programozási felület nyújthat sikerélményt, míg a haladók a C/C++ alapú Arduino nyelven szöveges kódot írhatnak. A kettő közötti átmenetet és a tanulási folyamatot segíti a „Blokkok és szöveg" lehetőség, amelynél a szöveges kód felülete inaktív és nem szerkeszthető, viszont a blokkok elhelyezésekor automatikusan kiírja a program a blokkoknak megfelelő szöveges forráskódot. A BBC micro:bit esetén egyelőre csak a blokkprogramozás elérhető.

Az 1. ábrán látható áramkörben a Tinkercad felületen egy Arduino UNO 12. számú digitális bemenetére egy kapcsolót, míg az 5. számú digitális kimenetére egy LED-et csatlakoztattunk egy áramkorlátozó ellenálláson keresztül. A példa szimulációnkban megvalósítjuk, hogy amikor a kapcsolót jobbra vagy balra csúsztatjuk, a LED rendre be- és kikapcsol. A bekapcsolásokat számláljuk, de 5-nél visszaállítjuk a számlálót a kezdőértékre.

A megvalósításhoz három integer típusú változót hoztunk létre (magenta), elágazások programozásához és várakoztatáshoz vezérlési struktúrákat használtunk (narancssárga), melyek logikai feltételeinél a Matematika blokkjait építettük be (zöld). A kapcsoló állapotát a bemenet értékéből olvassuk ki (lila), míg a LED állapotát, illetve a virtuálisan a számítógépre „nyomtatott" számlálóér-

téket a Kimenet (kék) blokkjaival írattuk ki. Kommentek beiktatására is van lehetőség (szürke). Megjegyezzük, hogy a setup() és a loop() függvények nem szerepelnek a blokkok között. A megfelelő blokk elhelyezésekor és beállításakor a program automatikusan beírja a beállításokat a setup()-ba, míg az egymás után fűzött blokkokkal valójában a loop() ciklusmagját kódoljuk.

Az ablak jobb szélső területén megjelenő szöveges kód másolással kiexportálható, és a fejlesztőkörnyezetben feltölthető a valós eszközre. A nemzetközi szakirodalom is egyre gyakrabban ajánlja ezt a lehetőséget általános iskolás korúakat tanító kollégák figyelmébe, bevezető szintű programozás tanításához [20,21].




**1. ábra:** Blokk alapú és szöveges programozás a Tinkercadben

A szöveges programozást is több hasznos funkció segíti. A Tinkercadben is elérhető néhány olyan függvénykönyvtár, amely nagyban megkönnyíti az Arduinóhoz csatlakoztatott aktív eszközök programozását (pl. Servo a szervomotorokhoz, NeoPixel a LED-szalagokhoz stb.) Ezek listája és dokumentációja a kódoló felületen Elemtárak címszónál érhető el.

Vannak olyan összetett eszközök (Indítók), amelyeknél a programozható eszközhöz már eleve csatlakoztattak bizonyos alkatrészeket, és ezeknek a felületre húzásával a vezérlő vagy mérést végző kód is bekerül a szövegszerkesztőbe. Több programozható eszköz is elhelyezhető a grafikus felületen, ez esetben külön-külön programozhatók az eszközök.

Egy további funkció segíti a kódolást: a hibakereső lehetővé teszi a kód szakaszos futtatását, sőt a szünetekben a változók aktuális értékét megjeleníti a program, ha a kurzort a változó fölé visszük.

## 3. A Tinkercad alkalmazása a műszaki informatika online oktatásában

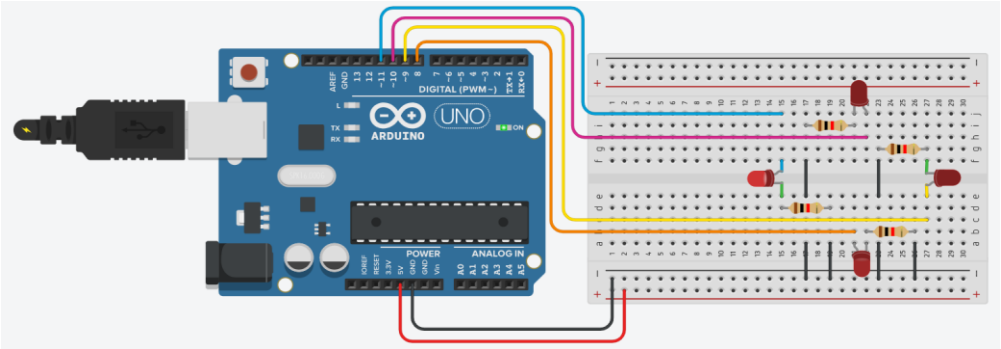
A következőkben bemutatunk néhány Tinkercadben készült Arduino-szimulációt, amelyeket különböző kurzusainkon a hallgatóink is elkészítettek. Nem titkolt célunk ugyanakkor az Arduino informatikatanárok körében történő népszerűsítése is: az itt bemutatott alkalmazásokat úgy igyekeztünk kiválasztani, hogy elsajátításukkal egy fokozatosan felépülő tudásanyagot nyújthassanak az Arduino-programozás kezdőinek, ugyanakkor a haladók is találjanak benne kihívást, tanulságos megoldásokat. A feladatok megoldásait és a forráskódokat tartalmazó Tinkercad-szimulációkat elérhetővé

tettük a honlapunkon [22]. Bár ezeket a gyakorlatokat itt szimulációként tárgyaljuk, hangsúlyozzuk, hogy az itt leírt kísérletek egy az egyben megvalósíthatók valódi eszközökkel is.

## 3.1. Digitális kimenet alkalmazása: futófény

Az Arduino-világban a LED-villogtatás programozása számít a „Hello world"-szintű bevezető programnak. Itt egyetlen LED-re kapcsolunk logikai magas (5V), illetve logikai alacsony (0V) jelszintet egyetlen digitális kimeneten keresztül. Ha az áramkörben elhelyezünk 4 db LED-et, amelyeket különböző digitális kimeneteken keresztül vezérelhetünk, ciklikus, illetve oszcilláló futófényeket is programozhatunk egyszerű vezérlési szerkezetek (pl. for ciklus) segítségével.

Egy másik alkalmazás a léptetőmotor (pl. 28BYJ-48) driveren (pl. ULN2003A) keresztül történő vezérlése [23], amely szerepel a tananyagban, viszont a Tinkercad nem tartalmazza ezt az aktuátort. Az online szimuláció [24] alapján jól látszik, hogy a ciklikus futófény programozása valójában – a sebességtől eltekintve - a léptetőmotor egy irányban történő mozgatásához szükséges logikai jelszintek beállításával egyenértékű. Így az online oktatás alatt a hallgatóknak a 2. ábrán szereplő áramkört kellett összeállítani, majd szimulálni 5 periódusnyi, óramutató járásával ellentétes irányú, majd rövid szünet után 5 periódusnyi óramutató járásával megegyező irányú ciklikus futófényt. Az ismétlésszámok növelésével és a lépések közötti időtartam csökkentésével a léptetőmotor oda-vissza mozgatása pl. kapunyitáshoz is megvalósítható, így a hallgatók által készült programokat később lehet léptetőmotoron is tesztelni [25].




**2. ábra:** Áramkör a léptető motort modellező futófény szimulációjához

## 3.2. Digitális bemenet alkalmazása: kapuvezérlő rendszer modellje

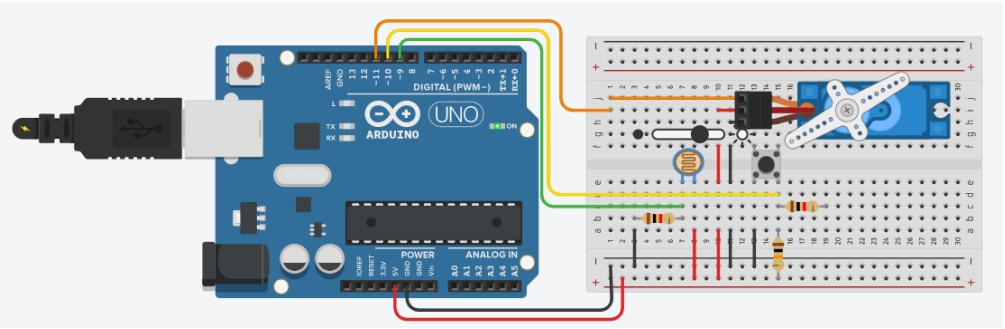
A digitális bemenetek alkalmazására fontos példák a két logikai szint váltására alkalmas kapcsoló és nyomógomb aktuális jelszintjének beolvasása. Ugyanakkor analóg szenzorokat is kapcsolhatunk digitális bemenetre: Az Arduino UNO esetében a digitális bemenetre kapcsolt 0 és 5V közötti jelek esetében 3V fölötti érték esetén a műszer logikai magas, míg alatta logikai alacsony jelet mér. (Az Arduino UNO-ban alkalmazott ATmega328 adatlapja alapján a bemeneti I/O portokon egy Schmitt trigger, valamint egy szinkronizáló áramkör is gondoskodik arról, hogy a folytonosan változó jeleket is helyesen tudja feldolgozni a mikrovezérlő.)

Feladatként kitűzhető, hogy valósítsa meg a hallgató vagy diák egy kapuvezérlő rendszer modelljét a Tinkercadben. A kapu forgatását végző mechanikai szerkezetet ezúttal egy szervomotor reprezentálja. A szervomotor működtetéséhez elegendő egyetlen digitális kimenet alkalmazása, és programozásához elérhető a Servo függvénykönyvtár Tinkercadben is. A modellben a szervomotort 90°-kal egyik irányba elforgatjuk, majd rövid szünet után az eredeti helyzetbe visszaforgatjuk, ezzel reprezentálva a kapu nyitását és zárását.

A kapu forgásának elindítását a felhasználó vezérli, egy nyomógomb segítségével, melynek jelét az Arduino UNO digitális bemenetén olvassuk be. A nyomógomb lenyomása a szimuláció elindítása után egérrel kattintással lehetséges.

Ha a kapu útjába kerül egy akadály, amelyet valamilyen optoszenzor-rendszer (fotokapu) érzékel, vészleállításra van szükség. Ennek egy lehetséges megoldása, hogy a kapu zárt állásának vonalában helyeznek el egy fotokaput, illetve a kapu előtt 1-2 méterre egy másikat. Az akadályt figyelő mozgásérzékelőt a tényleges kísérletben és ebben a szimulációs feladatban is egyetlen fotoellenállás (3. ábra) segítségével egyszerűsítettük le, amelyet a környezeti fény világít meg, de ha kezünkkel eltakarjuk, akkor megváltozik a mért logikai szint. (Ha a hallgatók két fényérzékelőt használnak, a modell még jobban közelíthető egy tényleges fotokapu-rendszerhez.) A fotoellenállás analóg jelét ezúttal egy digitális bemenetre kapcsoltuk. Ezt egy valós tantermi kísérletben megtehetjük, hiszen a motor mozgását leállító akadályt a hallgató keze jelenti, amely kitakarja a terem környezeti fényét. Mivel a választott szenzort és a vele sorba kapcsolt előtétellenállást az Arduino UNO digitális bemenetére kapcsoltuk, megfelelő kísérleti körülmények esetén az tapasztalható, hogy a jel tulajdonképpen bináris és megfelelő, hogy változására a kapu megálljon. Ennek oka, hogy a kísérletben a fotorezisztor ellenállása oly mértékben változik, hogy az analóg jel egyértelműen olvasható digitális bemeneten. [1]. A kísérlet szimulációjában a fényviszony-változás megfelelő szimulálására úgy van lehetőség, hogy indítás után a szenzorra kattintás hatására megjelenő csúszkán manuálisan változtatjuk a „megvilágító fény erősségét”.

A hallgatók feladata, hogy olyan szimulációt hozzanak létre, melyben a kapunyitás (motormozgás) nem indul el addig, amíg be nem kapcsolják (felhasználó lenyomja a nyomógombot), vagy a mozgásérzékelő (fotoellenállás) valamit érzékel a fényútban. Indítás után, ha a fényútba kerül valami (felhasználó a sötétedés irányába mozgatja a csúszkát), akkor a veszély elhárításáig megáll a kapu mozgása, ami az akadály eltávolítása után automatikusan újraindul. A kapu bezáródása után újabb engedélyre (nyomógomb lenyomására) vár a program [26].

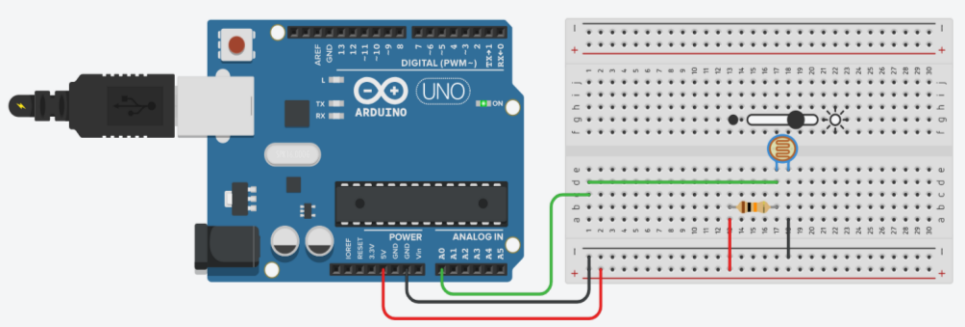



**3. ábra:** Áramkör a szervomotorral megvalósított kapuvezérlőrendszer-modell szimulációjához

## 3.3. Analóg bemenet és soros monitor alkalmazása: ellenállásmérés

Az Arduino analóg bemenetét feszültség- és ellenállásmérésre is alkalmazhatjuk [1]. Míg a laborgyakorlaton egy ismeretlen értékű ellenállás értékének a meghatározása a cél, addig a Tinkercadben a furatszerelt ellenállások ohmban mért ellenállása ismert a hallgató számára. Ha azonban a változtatható értékű fotoellenállást kapcsoljuk a feszültségosztóba, az ellenállásméréssel a félvezető eszköz elektromost tulajdonságainak vizsgálatára is lehetőség van. A szimuláció indítása után a csúszka mozgatása mellett megmérhető, hogy az erősebb megvilágítás esetén a szenzor ellenállása drasztikusan lecsökken. Itt egy említésre méltó önellenőrzési lehetőséget is ad a felhasználó számára a szoftver: található benne beépített multiméter is, amellyel szintén megvizsgálható a fényérzékelő ellenállása az áramkörből kivéve.

A Tinkercadben implementálták a számítógép és az Arduino közötti USB-porton történő soros kommunikációt is. A Soros monitor felnyitása után a grafikon ikonra kattintva a hagyományos fejlesztőkörnyezet (Arduino IDE) Soros plotteréhez hasonló diagramon lehet az idő függvényében megfigyelni az ellenállás értékét. [27]



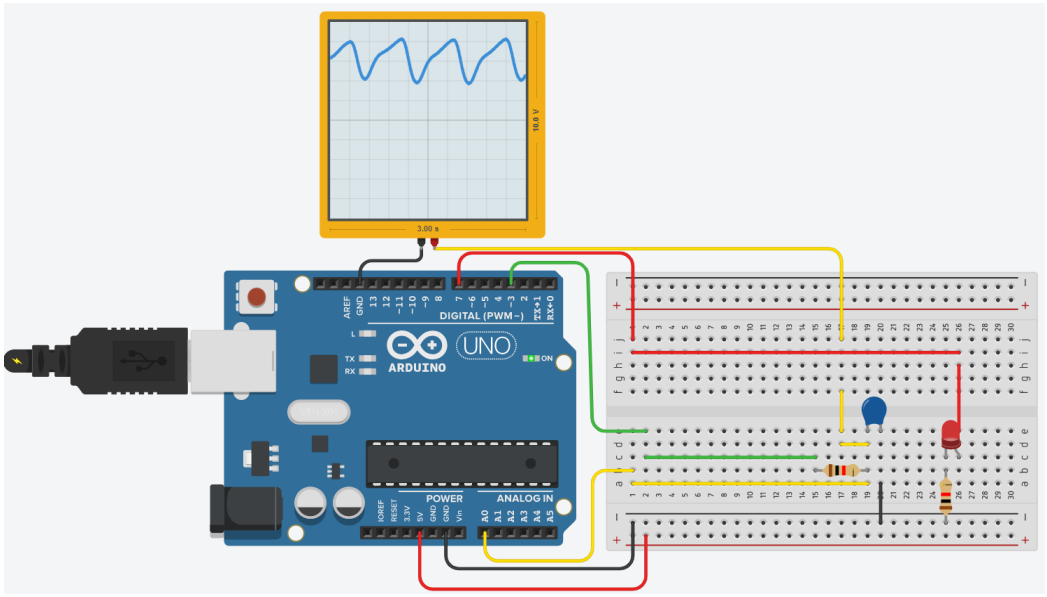

**4. ábra:** Áramkör a fotoellenállás különböző fényviszonyok között történő ellenállásmérésének szimulációjához

## 3.4. PWM jelgenerálás alkalmazása: fotopletizmográfiai mérésen alapuló pulzusmérés

Az Arduinóval végzendő fotopletizmográfiai méréshez a kutatócsoport által fejlesztett EDAQuino szenzor interfész használatát javasoljuk [28]. Ha azonban online alternatívát keresünk, nincs egyszerű dolgunk, ugyanis az ujjbegyben történő nyomásváltozás szimulálására nem alkalmas a Tinkercad.

Egyszerű periodikus jelek (négyszög-, szinusz-, háromszögjel) generálására ugyan lehetőséget nyújt a beépített Tesztgenerátor eszköze, viszont ezen jelek frekvenciáját a felhasználó állíthatja be, így a jel frekvenciájának mérése kevéssé tűnt megfelelő alternatívának, noha magának a szintmetszés algoritmusának, és a periódusidő, frekvencia meghatározásának programozását kiválóan el lehet sajátítani egy leegyszerűsített jelalakkal pl. háromszögjellel.

Egy megoldást szeretnénk arra mutatni, hogy hogy lehet mérésen alapuló jelalakot megvalósítani a szimulátorban. Az EDAQuino segítségével valós mérésként felvettük egy személy fotopletizmogram-jelét 100 Hz-es mintavételi frekvenciával. A mért jel néhány periódusnyi hosszát kivágtuk úgy, hogy azt ismételgetve folytonos jelet lehessen szimulálni, és eltároltuk egy tömbben. Mivel az Arduino UNO-n nem található analóg kimenet, ezért pulzusszélesség-modulációt alkalmaztunk: a PWM-típusú digitális jel kitöltési tényezőjét ciklikusan olvastuk ki a tömbből 10 ms-onként, majd egy aluláteresztő RC szűrő (passzív integrátor) segítségével átalakítottuk analóg jellé. Megjegyezzük, hogy mind PWM-jel, mind pedig az analóg jel alakjának grafikus megjelenítése is megoldható az Oszcilloszkóp eszköz segítségével az 5. ábrán látható módon. Ezt a jelet visszamértük az egyik analóg bemeneten.



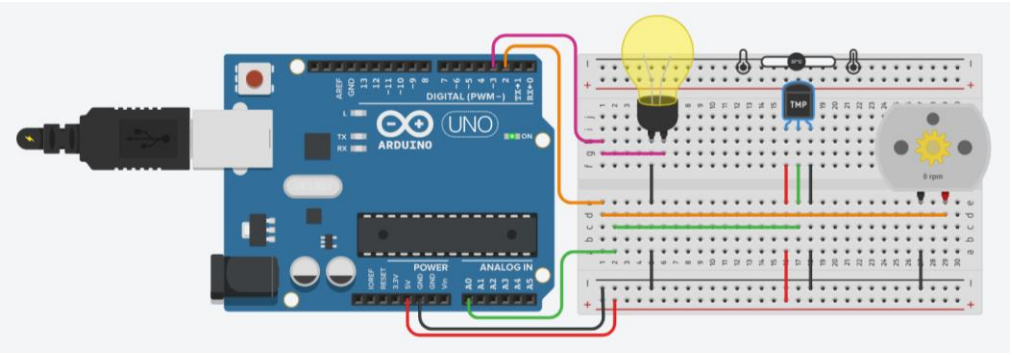

**5. ábra:** Áramkör a mért adatokból generált fotopletizmogramjel alapján történő pulzusmérés szimulációjához

A mért jelen egy lineáris értéktranszformáció (tükrözés) is végrehajtható szoftveresen, hogy a fotopletizmogramra emlékeztető jelet megkaphassuk, és a Tinkercad Soros plotterén megfigyelhessük. Ezután lehetővé válik az így kapott jel felszálló ágain történő szintmetszések időpontjai között eltelt időtartamok (periódusidő) mérése, illetve a reciprok értékét véve a pulzus meghatározása [29].

## 3.5. Hőmérsékletszabályozás

A hallgatók a gyakorlat során megismerkednek az on-off szabályzás elvével: feladatuk egy kétállásos hőmérsékletszabályozó rendszer modellezése. A hőmérséklet folyamatos monitorozása mellett, ha a hőmérséklet lecsökken egy bizonyos 1. számú szint alá, akkor bekapcsol a fűtés, amelynek hatására a hőmérséklet értelemszerűen megnövekszik, és egy 2. szint elérésénél a fűtés kikapcsol. Azonban, ha a rendszer túlmelegszik azaz a 2. számú szint fölé emelkedik, egy ventilátor is bekapcsol, és addig működik, amíg a hőmérséklet újra el nem éri a 1. számú szintet.

A Tinkercadben „látványos” fűtőellenállás nincs implementálva, így fűtőelemként egy izzót használunk. A hőmérséklet növekedésére nemlineáris módon csökkenő ellenállású félvezető analóg szenzor, a termisztor sajnos nem szerepel az elérhető eszközök között, viszont a TMP36 szenzor igen, így a mért analóg feszültség hőmérsékletté történő konverziója konverzió könnyen megvalósítható. A Tinkercad itt is ad egy ellenőrzési lehetőséget, hiszen a szimuláció futtatása közben megjeleníthető szabályozó csúszkán a program kiírja a beállított (és mérendő) hőmérséklet értéket, ami összevethető a mért értékkel. A ventilátort egy egyszerű egyenáramú motorral szemléltetjük. A fűtő-, illetve hűtő elemek hatása a hőmérsékletre sajnos nem érhető tetten a szimulációban, viszont még így, manuálisan „reagálva” és változtatva a hőmérséklet értékét is jól szimulálható és megérthető a szabályozórendszer működése [30].



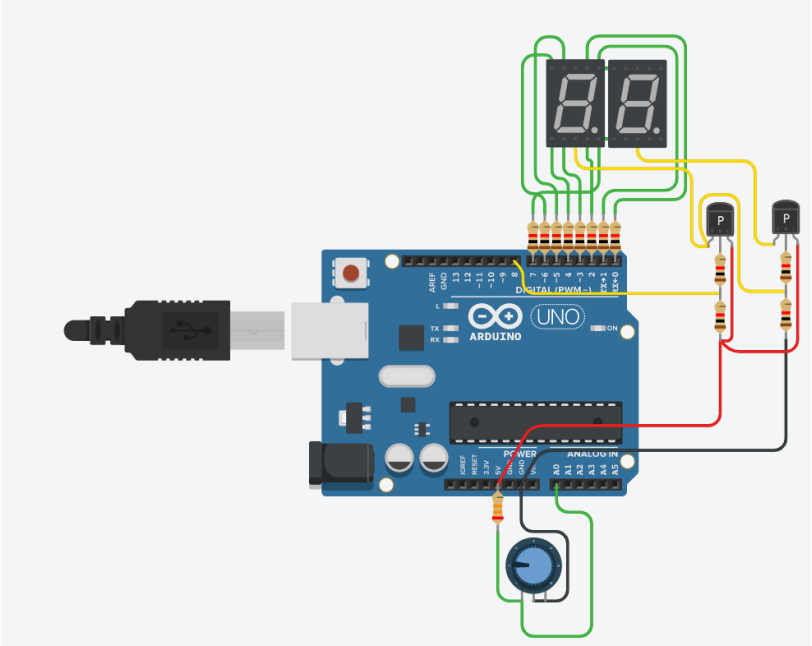

**6. ábra:** Áramkör a hőmérsékletszabályozó rendszer szimulációjához

## 3.6. Ellenállásmérés, 7-szegmenses kijelző vezérlése

Az ellenállásmérési feladatot kiegészíthetjük azzal, hogy egy a potenciométer egyik végpontja és a csúszóérintkezője közötti ellenállását két 7-szegmenses kijelzőn jelenítjük meg. A potenciométert egy referencia ellenállással sorba kötve megkapjuk a potenciométer ellenállását, ha a rajta eső feszültséget mérve az ADC kódot feszültségbe konvertáljuk és a feszültségosztó egyenletéből kifejezzük a mérendő ellenállást.

A 7-szegmenses kijelző vezérlését nagyban megkönnyíti, hogy a teljes portot, azaz mind a 8 bitet egy utasítással, a *PORTD* makróval tudjuk írni. Teljes port csak egy érhető el az Arduino UNO-n, a D port. Ennél a portnál viszont arra kell figyelni, hogy a 0 és az 1-es kivezetések a soros kommunikációval is osztoznak, így ha szükségünk van ezekre a bitekre, akkor nem ajánlott soros kommunikációt is használni mellette. A kivezetések módját digitális kimenetnek konfigurálhatjuk a DDRD makró segítségével, így nem kell egyesével beállítani a kimeneteket a pinMode() függvénnyel [31]. Mivel két kijelző van, de csak egy teljes port, ezért mindkét kijelző ugyanarra a portra van kötve és két tranzisztor segítségével van megoldva, hogy egyszerre csak az egyik legyen bekapcsolva. A két tranzisztort egy digitális kimenettel tudjuk vezérelni és amikor az egyik kinyit, akkor a másik zár. Ezt a kijelzőválasztó bitet viszonylag nagy frekvenciával (100 Hz) negáljuk, így a villogás nem látszik, de mindkét kijelző egyedileg is használható lesz. A Tinkercadben elérhető kijelzővezérlő IC is, de a hallgatók feladata lehet ennek a funkciónak a megvalósítása is.

Az ellenállást 0,1 kΩ-os felbontással tudjuk kiírni a kijelzőkre. Amikor a kijelzőválasztó bit logikai magas, akkor az első digittel, amikor logikai alacsony, akkor a második digittel kell kiindexelni karaktereket tartalmazó konstans tömböt. Ennek a módszernek az előnye, hogy nem kell a számkarakterek kezeléséhez egy hosszú, 10 esetet tartalmazó switch-case struktúrát használni, amiben minden számjegynek megfelelne egy-egy eset [32].




**7. ábra:** Áramkör a potenciométer ellenállásmérésén alapuló 7-szegmenseskijelző-vezérlés szimulációjához

# 4. A Tinkercad előnyei és hátrányai

Sorra vesszük a Tinkercad – oktatásban és tananyagfejlesztésben történő - használata során összegyűlt tapasztalatainkat két csoportra bontva: először bemutatjuk a program előnyeit és a benne rejlő lehetőségeket, majd a hátrányokat és a szimulátor alkalmazhatóságának korlátait is [11].

## 4.1. Előnyök

A távoktatás során a legfőbb előnye a programnak, hogy a laborgyakorlatok esetén legalább a gyakorlati jellegét meg tudtuk őrizni a kurzusoknak, és továbbra is megvalósítható volt a hallgatók aktív tanulása, a gyakorlás a frontálisan közölt vagy kiadott tananyagok elsajátítása helyett.

A tantárgyak közötti kapcsolatot lehet építeni komolyabb eszközbeszerzés nélkül. Az informatikaoktatás során be lehet vonni a fizika (elektronika) tudáselemeit, ugyanakkor az áramkörépítő alkalmazás a közoktatás szintjén alkalmas az egyszerű elektronikai mérések szimulálására is [33]. Az áramkörök breadboardon való megvalósítása nagyon szépen megtervezhető a szimulátorban: a vezetékek derékszögben hajlíthatók, így áttekinthetővé tehetők a „kísérleti elrendezések”.

Ha még hozzávesszük a lehetőségekhez a 3D geometriai tervező programokat, akkor könnyen belátható, hogy meglehetősen komplex STEAM-projektek támogatására is alkalmas a programcsomag.[1]

Internetelérés esetén bárhol, bármikor folytatható egy projekt, ami a programozás időszakát nagyon kényelmessé teszi. Később, amikor az Arduino és a hozzá csatlakoztatott egyéb eszközök

[1] A mozaikszó ’Science, Technology, Engineering, Art & Math’, azaz természettudományok, technológia, mérnöktudományok, művészet és matematikatudományterületek ötvözésére utal.

rendelkezésre állnak, ki lehet próbálni a programot, de a kísérletet így már megelőzi egy hosszabb tervezés és szimulátorral segített tesztfolyamat, ami a tényleges kísérlet sikerét szinte garantálja. Osztálytermi környezetben, ha nincs elegendő Arduino Kit, akkor is mindenki haladhat a programozással, folyamatos visszajelzés mellett, később váltva kipróbálhatják a kész programot valós eszközökön. Nem csupán a tanulási folyamatban, de saját projektek készítésekor a tesztelés során elkerülhetetlen a hibák lokalizálása, melyhez a korábban részletezett hibakereső nagy segítséget nyújt.

## 4.2. Hátrányok

A következőkben ismertetjük a Tinkercad jelenlegi verziójának általunk felismert korlátait is.

Jó lenne, ha mobil eszközökön is elérhető lenne az applikáció. Ugyan létezik már iOS-változat (az Android-verzió jelenleg nem elérhető) a szoftverhez, viszont jelenleg a Circuits csak megnyitást enged, sem a tervek szerkesztése, sem pedig a szimuláció nem megengedett. [34]

Oktatási szempontból mindenképp hangsúlyozni kell, hogy nagyon meggyorsítja a tesztelési, hibakeresési folyamatot egy előzetes szimuláció készítése. Ugyanakkor – online oktatás során - nem fog olyan hibákat produkálni, mint a valóság, ahol hibás lehet egy csatlakozás, vezeték vagy komponens. Ez előny is, kevesebb hibakeresésre van szükség, ugyanakkor fontos készségeket nem sajátítanak el.

Elektronikai szempontból elmondható, hogy bizonyos alkatrészek fizikai tulajdonságai nem, vagy nem helyesen vannak megvalósítva. A LED-eknek a nyitófeszültsége a LED színével van összefüggésben, viszont itt színtől függetlenül ugyanannál a feszültségértéknél nyitnak ki a világító diódák. Bizonyos eszközöknél a multifunkcionalitást hiányoljuk: a multiméter és a függvénygenerátor csupán a legalapvetőbb funkcióval rendelkezik. A szoftvert viszonylag könnyű úgy terhelni (pl. túl gyakori mintavételezéssel), hogy lelassuljon a szimuláció közben, és ne valós időben mutassa a kísérletet.

A hobbielektronika világa olyan rohamosan fejlődik, ezzel a Tinkercad nem veszi fel a versenyt. Viszont eléggé alapvetőnek számító alkatrészek is hiányoznak (pl. termisztor, stepper motor). Amit még hiányolunk, hogy nincsen lehetőség saját eszköz tervezésére, bevitelére. Jó lenne, ha további programozható eszközök (pl. Arduino Nano vagy Arduino Mega) elérhetők lennének. Korábban elérhető volt az ESP8266 wifi modul, de sajnos biztonsági okokra hivatkozva már nem elérhető.

Ami a programozási környezetet illeti, jó lenne, ha bővítenék a blokkprogramozásban elérhető blokkokat, továbbá, ha a felhasználó tudná bővíteni a saját függvénykönyvtárát, esetleg saját blokkokat létrehozni. Ezen kívül nem túl felhasználóbarát a szövegszerkesztő (nincs vonalzó, automatikus behúzások, automatikus zárójelbezárás vagy épp gépelési hiba jelzése (kisbetű, nagybetű), javítása. Természetesen külső szövegszerkesztőben elért külalakot megőrzi a Tinkercad is.

# 5. Összefoglalás

Tanulmányunkban oktatási tapasztalataink alapján bemutattuk a Tinkercad Circuits ingyenes, online Arduino-szimulátort, melyet az online programozásoktatásban alkalmaztunk. Bemutattuk a Tinkercad oktatást támogató funkcióit, az áramkörmodellezést lehetővé tévő grafikus objektumait és az intelligens elektronikai eszközök (Arduino UNO, ATtiny, BBC micro:bit) programozását lehetővé tévő fejlesztőkörnyezetet. Ezt követően ismertettünk konkrét szimulációkat, amelyeket az eredetileg laborgyakorlatnak meghirdetett műszaki informatikai kurzusaink online oktatásra történő átállásakor alkalmaztunk a gyakorlati feladataink alternatívájaként. Végül sorra vettük, milyen lehetőségeket rejt magában, ugyanakkor milyen korlátokkal rendelkezik a Tinkercad.

Bízunk benne, hogy az általunk bemutatott platform felkelti az informatikatanárok, oktatók érdeklődését, és saját oktatói munkájuk során is hasznosnak találják majd az általunk bemutatott ötleteket, megoldásokat.

## Köszönetnyilvánítás

A tanulmány elkészítését a Magyar Tudományos Akadémia Tantárgypedagógiai Kutatási Programja támogatta.

## Irodalom

1. Makan G., Dóra A., Mingesz R., Gingl Z., Kopasz K., Mellár J.Z., Vadai G., *Interdiszciplináris műszaki gyakorlatok az informatikatanár szakon*. (2018). https://doi.org/10.6084/m9.figshare.7359203.v1
2. G. Makan, D. Antal, R. Mingesz, Z. Gingl, J. Mellár, G. Vadai, K. Kopasz, *Interdisciplinary Technical Exercises for Informatics Teacher Students*. Central-European Journal of New Technologies in Research, Education and Practice, (2019) 1. 21–34. https://doi.org/10.36427/CEJNTREP.1.1.382
3. *Arduino Remote Learning*. https://www.arduino.cc/remotelearning (Utoljára megtekintve: 2020. 10. 28.)
4. M.G. Jamil, S.O. Isiaq, *Teaching technology with technology: approaches to bridging learning and teaching gaps in simulation-based programming education*. International Journal of Educational Technology in Higher Education, (2019) 16. 25. https://doi.org/10.1186/s41239-019-0159-9
5. P.L. Rocca, F. Riggi, C. Pinto, *Remotely teaching Arduino by means of an online simulator*. Physics Education, (2020) 55. 063003. https://doi.org/10.1088/1361-6552/abaa21
6. *A Beágyazott rendszerek alapjai tantárgyhoz: taneszközkészlet, ajánlható irodalom, szoftverek. | Villamos Csoport*. http://vill.elmki.hu/taneszkozkeszlet-a-beagyazott-rendszerek-alapjai-tantargyhoz/#more-2270 (Utoljára megtekintve: 2020. 11. 2.)
7. *Top 10 Best Simulators for Arduino*. Projectiot123 Technology Information Website Worldwide, (2019). https://projectiot123.com/2019/03/15/top-10-best-simulators-for-arduino/ (Utoljára megtekintve: 2020. 10. 28.)
8. *Teacher Resources for Computer Engineering*. TryEngineering.Org Powered by IEEE,. https://tryengineering.org/category/teacher-resources/ (Utoljára megtekintve: 2020. 10. 28.)
9. *Electronics Online - FUSE - Department of Education & Training*. https://fuse.education.vic.gov.au/Resource/LandingPage?objectId=2c138337-0fcf-4425-92d3-a8e39e9015d4 (Utoljára megtekintve: 2020. 10. 28.)
10. *The Arduino Simulator you've been looking for!* Programming Electronics Academy, (2019). https://www.programmingelectronics.com/arduino-simulator-tinkercad/ (Utoljára megtekintve: 2020. 11. 5.)
11. *Arduino Simulator Q & A*. Programming Electronics Academy, (2019). https://www.programmingelectronics.com/tinkercad-part-2/ (Utoljára megtekintve: 2020. 10. 28.)
12. *Arduino alkalmazása a fizika és a digitális kultúra oktatásában | MISZAK*. http://www.inf.u-szeged.hu/miszak/arduino-alkalmazasa-a-fizika-es-az-informatika-oktatasaban/ (Utoljára megtekintve: 2020. 11. 2.)

13. *miszak-mta-szte | A MISZAK YouTube-oldala*.
https://www.youtube.com/channel/UCak16GKllLDG1w1ZspKd3Ww (Utoljára megtekintve: 2020. 11. 2.)
14. *Tinkercad | Create 3D digital designs with online CAD*.
https://www.tinkercad.com (Utoljára megtekintve: 2020. 10. 28.)
15. *Tinkercad*. Wikipedia, (2020).
https://en.wikipedia.org/w/index.php?title=Tinkercad&oldid=985779734 (Utoljára megtekintve: 2020. 10. 28.)
16. *Autodesk | 3D Design, Engineering & Construction Software*.
https://www.autodesk.com/ (Utoljára megtekintve: 2020. 10. 28.)
17. *Tinkercad tanárok számára*.
https://www.tinkercad.com/teach (Utoljára megtekintve: 2020. 10. 28.)
18. *Instructables - Tinkercad circuits*.
https://www.instructables.com/member/circuits/ (Utoljára megtekintve: 2020. 10. 28.)
19. M. Resnick, J. Maloney, A. Monroy-Hernández, N. Rusk, E. Eastmond, K. Brennan, A. Millner, E. Rosenbaum, J. Silver, B. Silverman, Y. Kafai, *Scratch: programming for all*. Communications of the ACM, (2009) 52. 60–67.
https://doi.org/10.1145/1592761.1592779
20. B.N. Mohapatra, R.K. Mohapatra, J. Joshi, S. Zagade, *Easy Performance Based Learning Of Arduino And Sensors through Tinkercad*. International Journal of Open Information Technologies, (2020) 8. 4.
21. C. Vidal-Silva, J. Serrano-Malebran, F. Pereira, *Scratch and Arduino for Effectively Developing Programming and Computing-Electronic Competences in Primary School Children*, in: 2019 38th International Conference of the Chilean Computer Science Society (SCCC), 2019 1–7.
https://doi.org/10.1109/SCCC49216.2019.8966401
22. *A Tinkercad Arduino-szimulátor alkalmazása az online programozásoktatásban | MISZAK*.
http://www.inf.u-szeged.hu/miszak/infodidact2020/ (Utoljára megtekintve: 2020. 11. 5.)
23. A. Somogyi, A. Kelemen, R. Mingesz, *Motion tracking by an Arduino Due and Excel*, in: Proceedings of XXXIII. DidMatTech 2020 Conference. New Methods and Technologies in Education, Research and Practice, Eötvös Loránd University in Budapest, Trnava University in Trnava
24. *Stepper motor*. Wikipedia, (2020).
https://en.wikipedia.org/w/index.php?title=Stepper_motor&oldid=986069649 (Utoljára megtekintve: 2020. 11. 2.)
25. *Ismételt futófény számlálóval*.
https://www.tinkercad.com/things/5Bq0GhUsLDt (Utoljára megtekintve: 2020. 11. 1.)
26. *Kapuvezérlő rendszer szervomotorral*.
https://www.tinkercad.com/things/bitdgKX3BAe (Utoljára megtekintve: 2020. 11. 1.)
27. *Ellenállásmérés*. https://www.tinkercad.com/things/cysTcfu2OOT (Utoljára megtekintve: 2020. 11. 1.)
28. Z. Gingl, J. Mellár, T. Szepe, G. Makan, R. Mingesz, G. Vadai, K. Kopasz, *Universal Arduino-based experimenting system to support teaching of natural sciences*. Journal of Physics: Conference Series, (2019) 1287. 012052.
https://doi.org/10.1088/1742-6596/1287/1/012052
29. *Fotopletizmográfia pulzusszélesség-modulációval - pulzusmérés*.
https://www.tinkercad.com/things/0YSWThiW8rJ (Utoljára megtekintve: 2020. 11. 1.)

30. *Hőmérsékletszabályozás*.
https://www.tinkercad.com/things/lCLayymNgrW (Utoljára megtekintve: 2020. 11. 2.)
31. *Arduino - PortManipulation*.
https://www.arduino.cc/en/Reference/PortManipulation (Utoljára megtekintve: 2020. 11. 3.)
32. *Ellenállásmérés, 7-szegmenses kijelző vezérlése*.
https://www.tinkercad.com/things/gMqp1WRamdd (Utoljára megtekintve: 2020. 11. 3.)
33. *Elektromosság KIT*.
https://sites.google.com/view/elektromossag-kit (Utoljára megtekintve: 2020. 11. 3.)
34. *Tinkercad for iOS*.
https://apps.apple.com/us/app/tinkercad/id1469440830 (Utoljára megtekintve: 2020. 10. 28.)

# Tanárjelöltek tanítási gyakorlatának jelenlegi kérdései, a tanárképzés digitális átalakítása

Stoffová Veronika[1], Czakóová Krisztina[2]

[1]veronika.stoffova@truni.sk, [2]czakoovak@ujs.sk
[1]Trnava University in Trnava, Faculty of Education, Department of Informatics and Computer Science
[2]J. Selye University in Komárno, Faculty of Economics and Informatics, Department of Informatics

**Absztrakt.** A tanító- és tanárképző programokat megvalósító egyetemi karok hosszú kitartó munkával kiépítették a gyakorló iskolák hálóját, kidolgozták és kifejlesztették a tanító- és tanárjelöltek pedagógia gyakorlatának rendszerét. Ez mind méretre szabottan a nappali képzésre irányult. A COVID-19 pandémia idejében a nappali képzés egyik napról a másikra távutas képzésre váltott nemcsak az egyetemeken, hanem a kötelező iskolalátogatást biztosító intézményekben is. Ennek kapcsán sok új probléma merült fel nem csak az egyetemi, hanem az általános iskolai és középiskolai oktatásban egyaránt. Legfőképpen ez a laboratóriumi munkát megkövetelő tantárgyak tanításában okoz jelentős gondot. Specifikus gondok jelentek meg a leendő tanítók és tanárok pedagógia gyakorlatának lebonyolítása során is. A szerzők célja megfogalmazni azon nehézségeket, melyekkel a megnevezett két szlovákiai egyetem (tanárképző programok) hallgatói a Selye János Egyetemen és a Nagyszombati Egyetemen szembesülnek.

**Kulcsszavak:** távoktatás, pedagógia gyakorlat, e-learning, on-line képzés

## 1. Bevezető

A járvány helyzet egyre nagyobb hatással volt és van a kontaktkapcsolatokon alapuló tantermi tanítási gyakorlat megvalósítására is. A tanítási gyakorlat hagyományosan az általános és középiskolákban a tantermi tanításra összpontosított. Senki sem számított egy olyan helyzettel, hogy menet közben egyik napról a másikra távoktatási formát vesz fel a kötelező iskolalátogatás. A szlovákiai iskolák nem voltak erre felkészülve, hiszen távoktatás addig a gyakorlatban csak kivételes esetben került megvalósításra. A módszertani eszközöket, amelyek a nappali képzésben beváltak, csak módosítás után lehetett a távoktatásba bevinni. Azzal, hogy hogyan lehet a digitális oktatási technológiákat, az online technológiákat és az interaktív oktatási eszközöket és az oktatáshoz használt szoftvereket az általános és középiskolákban a távoktáshoz szabni, nem foglalkozott senki. A digitális technológiák hatékony használatának előfeltétele az iskolákban, hogy a tanároknak legyen hozzáférésük a szükséges technikai eszközökhöz és technológiákhoz, valamint elegendő digitális műveltséggel és kompetenciával rendelkezzenek azok használatához (Czakóová, 2019; Záhorec et al., 2020). Ugyanezt feltételezik és elvárják a tanulótól is. Az oktató és tanuló technikai és technológiai felszereltsége, valamint felhasználói kompetenciái és képességei a digitális technológiák alkalmazásában jelentős szerepet játszik a távoktatás sikeres lebonyolításában Stoffová, 2018; Stoffová – Gabaľová, 2019). A cikk arról is beszámol, hogy Szlovákiában az oktatók miként kezelték a távoktatást a COVID-19 koronavírus-járvány idején.

## 2. A helyzet rövid jellemzése

A digitális technológiák hatékony alkalmazása kétségtelenül fontos feltétele mind a minőségi oktatás megvalósításának, mind az iskola zavartalan működésének, akár az adminisztratív munkában, akár a nyilvánossággal, különösen a szülőkkel folytatott kommunikáció során. A fejlődés egyértelműen azt mutatja, hogy a technológiák hatása a tanítási folyamatra jelentős (Czakóová. 2016). A képzési folyamat szervezésében, irányításában és adminisztrálásában sikerült megvalósítani az átállást, amit

egyértelműen bizonyít a képzési folyamat működése a koronavírus-járvány idején is (Hyksová – Stoffová.2020).

Az iskolák Szlovákiában 2020 márciusának elején teljesen bezárták kapuikat, és a tanítás átállt online üzem módba. A 2019/2020 iskolaév végéig – kisebb nagyobb kivétellel – az iskolák távoktatással működtek. Az oktatók és a tanulók szokatlan helyzetbe kerültek a COVID-19 járvány idején. Az összes iskolának egyszerre kellett megoldania a különböző helyzetekből adódó mindennapi problémákat, amelyek operatív kezelésre szorultak és komoly döntéseket eredményeztek. Sok kérdés merült fel, amelyeket meg kellett válaszolni. Az egyik legnagyobb problémát a tanulók és tanárok eszközellátottsága okozta. A tanulóknak/diákoknak és a tanároknak egyaránt saját digitális eszközeiket kellett használniuk – számítógépeket, laptopokat, táblagépeket és okostelefonokat, ill. mobiltelefonokat. Sajnos az iskolák nem tudták, a nehéz szociális helyzetű diákjaikat kellő mennyiségű és minőségű eszközzel ellátni, így ezek hátrányos helyzetbe kerültek. Ugyanakkor a tanárok anyagi helyzete sem feltétlenül engedte meg a hiányzó eszközök pótlását.

Az oktatás gyakorlati megvalósítása során az alábbi kérdések, problémák merültek fel:

- Hogyan lehet megoldani a tanítás folytatását? Távoktatás, kombinált forma vagy egészen más forma a „jó" megoldás?
- Milyen eszközöket lehet és érdemes használni a távoktatásra?
- Milyen tanulási környezetet lehet és érdemes használni?
- Hogyan lehet a képzésbe aktívan bevonni a tanulókat?
- Megfelel-e a távoktatásra a diákok otthoni technikai és technológiai felszereltsége?
- Megfelel-e a távoktatásra a tanárok otthoni technikai és technológiai felszereltsége?
- Megfelel-e a távoktatásra az iskola technikai és technológiai felszereltsége?
- Vajon az online oktatás mobiltelefonokon, táblagépeken, számítógépeken vagy laptopokon fog-e zökkenőmentesen működni?

Az iskolák központi támogatást és irányítást vártak, de a központi rendeletek és betartásuk, csak nagyobb felelősséget és terhet hozott a z iskoláknak és vezetőinek. A konkrét segítség főleg a tanítók és tanárok együttműködéséből született. Elektronikusan feldolgozott tananyagokat, tudásmérő eszközöket, elektronikus és online teszteket, elektronikus tankönyveket és egyéb más online tanításhoz alkalmas taneszközt bocsájtottak a kollégák és diákjaik rendelkezésére (Stoffová – Czakóová, 2016; Czakóová, 2017; Pšenáková, 2016). A szlovák televízió STV2 csatornája meghosszabbította a mindennapi oktatást támogató műsorát (szlovák nyelvű iskolák számára). Több oktatásra irányuló vagy projekt eredményeit/kimeneteit tartalmazó portál kínálta a pedagógusoknak szolgáltatásait, és ingyenesen elérhetővé tette az elektronikus tananyagait és tankönyveit.

A 2019/2020 iskolaév, a problémák ellenére, kicsit késve, de sikeresen lezárult. A diákok és az érettségizők bizonyítványt kaptak. Mindenki abban reménykedett, hogy a következő iskolaév már hagyományos formában fog lezajlani.

## 2.1. 2020/2021 tanév rendelkezései

A 2020/2021 tanév szeptemberében az iskolák megnyitották kapuikat, és a tanítás szabályosan hagyományos módon elindult. A tanítók és tanárok igyekeztek behozni a lemaradást, pótolni a hiányosságokat az előbbi tanévből és egyidejűleg tanítani az aktuális tananyagot. De ez nem tartott sokáig.

2020. október10-től a középiskolák újból távoktatásra tértek át, majd október 26-tól az általános iskolák 2. szintjének tanulói nappali oktatása további rendelkezésig megszakadt, és a diákokat távoktatás formájában tanítják 2020. november 16-tól az általános iskolák 2. osztályának szociálisan hát-

rányos helyzetű tanulói visszatérhettek az iskolákba, de csak kis csoportokban, 5 diák + 1 tanár felállásban tanulhatnak.

## 2.2. Távoktatás szervezése

A távoktatás kétféleképpen zajlik – az iskolaügyi miniszter rendeletes szerint a diákok módosított órarenddel rendelkeznek – meghatározott óraszámú online oktatással. Ez elsősorban az oktatási terület fő tantárgyaira vonatkozik**.** Például az informatika tantárgy esetében a fő tantárgy a matematika, mivel a Matematika és az informatika tantárgycsoportba tartozik. Az informatikát, mint kiegészítő tantárgyat ezen a területen, többnyire távoktatással, munka végzéssel, feladatok megoldásával, projektmunkával online órákkal kombinálva oktatják.

Az online oktatás az általános iskolákban főleg az EduPage rendszer szolgáltatásainak kihasználásával történik. Az EduPage komplex iskolai információs rendszert a legtöbb szlovákiai iskola már az online oktatás előtt bevezette és lehetőségeinek nagyobb részét kihasználja. A tanár elkészítheti és beállíthatja az online órákat a Zoom, a Teams, a GoogleMeet vagy a JitsiMeet, illetve a Skype vagy a Messenger segítségével. Visszacsatolásra egyrészt tesztek, másrészt házi feladatokat készíthet. A bezkriedy.sk portált általában azon iskolákban használják, amelyek a zborovna.sk tanárportál támogatásával dolgoznak. Ugyanis a „Kréta nélkül" (bezkriedy) portál a Tanácsterem (zborovňa) portál szerves része (Stoffová, 2020).

Az online oktatási platformot az iskola típusától és helyétől függően választják meg, függetlenül attól, hogy a gyermekek rendelkeznek-e megfelelő technikai eszközökkel és kellően „erős" internetkapcsolattal otthon. Ez a feltétel nem mindig teljesül és az órák alatt is gyakran megszűnik a hálózati kapcsolat, egyiknek nincs kamerája, másnak mikrofonja stb., a diákok kiesnek az óráról. Nem elhanyagolható az a tény sem, hogy miképp oldják meg a technikai eszközök megosztását azok a szülők, akik több alap- vagy középoskolást nevelnek otthon, illetve ők is home-office-ban dolgoznak.

Néhány iskolában a tanárok is technikai eszközök hiányával küzdenek. Saját IKT-eszközt, régi laptopokat vagy számítógépeket használnak otthon, sok esetben bizonytalan internetkapcsolattal, mivel az internetszolgálatók semmiféle felelőséget nem vállalnak. További gondot jelent a tanárok számítógépes gondolkodásának, digitális kompetenciájának alacsony szintje, ami tovább csökkenti az online oktatás hatékonyságát (Stoffová V. – Stoffová M. – Némethová, 2020).

# 3. A tanítási gyakorlat jelenlegi problémái és megoldásai

A fentebb összegzett körülmények mellett lehetetlenné vált a tanítási gyakorlat hagyományosan szervezett lebonyolítása. Új feltételek mellett, komplikált és nehéz helyzetben kellett és kell a tanárképző karok hallgatóinak a kötelező tanítási gyakorlatot az általános és középiskolákon megvalósítani. Az utolsó két szemeszterben a tanítási gyakorlat menetét és formáját nagy részben megbélyegezte a pandémia. Mindkét szemeszter elején, amikor a tanítás hagyományos formában folyt, sikerült néhány hallgatónak a tanítási gyakorlatot befejezni. A hallgatók zöme úgy a Nagyszombati Egyetemen, mint a Komáromi Selye János Egyetemen a tanítási gyakorlat felében állt, de akadtak olyanok is, akik még el sem kezdték. A tanítási gyakorlatok sikeres befejezése nemcsak a tanárképző egyetemi karoknak okozott nagy gondot, hanem a gyakorlóiskoláknak is. A tanítási gyakorlat kivitelezésének idomulni kellett a gyakorlóiskolákon futó távoktatáshoz. A kialakult, nem épp pozitív helyzetre a gyakorlatvezető tanárok kreativitása és a gyakorló iskolák gyakorló tanárainak rugalmassága jelenthetett/jelentett megoldást. Az egyetemi tanárképzési program hallgatóinak alkalmazkodni kellett a gyakorlóiskola teljes vagy részben távoktatási üzemmódjához és a gyakorló tanár/tanító által megválasztott tanítási formához. Számtalan hagyományos tantermi tanári aktivitás más online vagy offline aktivitással volt helyettesítve, kibővítve, mint például tananyagot feldolgozó prezentációk, videofelvételek készítése, a tanulóknak/diákoknak online mentorálása adott tantárgyakból stb.

A tanítási gyakorlat sikeres befejezéséhez figyelembe kellett venni bizonyos tényeket és alkalmazkodni az iskolákban kialakult aktuális helyzethez és ennek gyors változásához. A jelenleg futó szemeszter – 2020/2021-es tanév 1. félév – esetében ezek a következők.

- Az egyetemekkel szerződött gyakorló iskolákban is átállt az oktatás online formára.
- Kivételt az alsó tagozatos tanulók képezik, akik bizonyos időszakban még korlátozott létszámban megjelenhetnek az iskolapadokban, kontaktórák formájában tanulhattak, betartva a szigorú közegészségügyi szabályokat (maszkviselés, kézfertőtlenítés stb.)
- Egy további kivétel a 2. szinten tanuló szociálisan hátrányos helyzetű tanulók. Ezeknek a tanulóknak a tanítása engedélyezett kis csoportban (5 tanuló+1 tanító), kontaktórák formájában.
- Ezen szabályok nem minden esetben teszik lehetővé a hallgatók részvételét az intézményekben tartott tanítási órákon.
- Az oktatás maga sem a megszokott, órarend szerinti módon zajlik. Ennek egyik magyarázata és akadálya, hogy a tanítók sem voltak jelen teljes létszámban (betegség, fertőzöttség, karantén miatt).

# 4. Megfigyelések, kutatások eredményei

## 4.1. Összegzés, következtetés

A távoktatás alatt számos tény került felszínre, amely továbbgondolásra ad okot. A pandémiának sok negatív hatása volt az oktatásra és annak minőségére, de mindenképpen van néhány pozitív hozadéka is. A megfigyeléseink és kérdőíves kutatásaink eredményei ezeket a feltételezéseket megerősítették és alátámasztották.

A modern online oktatási technológiák használata a távoktatásban a járvány idején megerősítette azok fontosságát és hasznosságát.

Számos digitális technológia, információs és kommunikációs rendszerek és platformok, amelyek a tanárok rendelkezésére álltak, nem voltak addig kellő mértékben használva.

Sok eddig (a nappali tanításban) kis mértékben használt eszköz életre kelt, és alkalmazásuk a járvány időszak távoktatása alatt nélkülözhetetlenné vált.

Azok a tanárok, akik a nappali tanításban probléma nélkül használták a modern digitális taneszközöket és technológiákat, probléma nélkül alkalmazták a távoktatásban is. A távoktatási formát nem tartották megterhelőnek, kisebb módásításokkal gond nélkül tudták folytatni a tanítást az eddig is használt eszközökkel.

Azok, akik eddig figyelmen kívül hagyták a modern digitális taneszközöket és technológiákat, most kénytelenek voltak használni őket. Menet közben – használva, sokszor próba szerencse eljárással – tanulták az eszközök kezelését. Ezek hatékonysága azonban erősen megkérdőjelezhető. A megkérdezettek, azonban a távoktatásra, a tanórákra való felkészülést időigényesnek tartották, különösen az új digitális tananyagok fejlesztését, az online feladatok, tesztek, vizsgák elkészítését és ezek kiértékelését.

A tapasztalt és megfelelő szintű digitális műveltséggel rendelkező informatikusokszakos tanárok, akik korábban is használták ezeket a modern digitális lehetőségeket és folyamatosan saját didaktikai alkalmazásokat hoztak létre, rendelkeztek digitálisan feldolgozott interaktív tananyaggal. Arra a kérdésre, hogy: *Kihívást jelentett-e Önnek ezek a (online) tanítási módszerek?* Leginkább azt válaszolták, hogy nem, mert eddig is használták azokat.

Az iskolák, a tanárok és a tanulók nem voltak felkészülve az ilyen jellegű tanításra. A tanárok (a felmérés válaszadói) személyes tapasztalatai alapján elmondható, hogy hiányzott a szervezettség és a

fegyelem az online oktatásból. A tanárok jellemzően túlzásba vitték a házi a feladatok kiosztását, elvárták, hogy a diákok folyamatosan készüljenek, becsületesen és önállóan dolgozzanak és a számonkérésnél ne csaljanak.

Ezzel szemben, sok diák pedig azt hitte, hogy „járvány szünet" van. Nógatni kellett őket, hogy rendszeresen (együtt) dolgozzanak és látogassák az online tanítási órákat.

Az ideális megoldás az lenne, ha az iskola nemcsak tervet készítene egy ilyen helyzetben való továbblépésről, hanem el is látná a tanítókat és diákokat szükséges digitális eszközökkel és technológiákkal, melyeket felhasználva növelni lehet a tanulási/tanítási folyamat hatékonyságát. Abban az időszakban azonban, amikor a COVID-19 vírus gyors terjedése meglepett mindenkit, senkinek nem volt ilyen megoldása. A keletkezett problémákat mindenki a maga módján próbálta megoldani.

A kellő információs műveltség (digitális írástudás) és a szükséges digitális kompetenciák mindkét (a tanárok/tanítók és a diákok/tanulók) oldalon hiányoztak.

Egy iskolán belül több különböző távoktatási platformot is használtak, így megtörténhetett az, hogy a tanuló/diák minden tanítási órán más és más környezetben dolgozott.

Sok esetben a tanítók, legfőképp az általános iskolák alsó tagozatán a szülők (nagyszülők) segítségére támaszkodtak, akik fel voltak szólítva az együttműködésre. A szülők azonban nem rendelkeztek ideális feltételekkel és lehetőségekkel (nagy részük munkaviszonya nem tette lehetővé az otthonmaradást/otthoni munkavégzést) és nem rendelkeztek sem a tartalmi, sem a digitális kompetenciákkal

Az iskolákon különféle információs (és irányító) rendszereket használtak. Ezek lehetőségei és szolgáltatásaik csak kis mértékben voltak kihasználva.

Mivel az oktatás tervezése nem volt központilag irányítva, megtörtént az is, hogy egy diáknak ugyanabban az időben több online tanítási órán kellett volna részt vennie.

Főleg a Microsoft, de más szoftverfejlesztő és kivitelező cég is megoldást kínált. További funkcióval, lehetőséggel bővítették termékeiket, keretrendszereiket és platformokat, amelyek főleg a távoktatás megkönnyítésére irányultak.

Számos szabadon elérhető intenzív tanfolyam indult a tanítók és tanárok számára a felkínált eszköz lehetőségeiről és kezeléséről.

Az előbbiekből látható, hogy a távoktatás kulcsszemélye elsősorban az oktató. Természetesen, megfelelő technikai felszereltség nélkül nem lehet ~~a~~ csodát tenni. Ha az oktató jártas az egyes eszközök használatában, meg van győződve hasznosságukról és jelentőségükről, a tanításban képes erre rászoktatni diákjait is. Fontos, hogy az eszköz használata ne okozzon stresszt és ne csökkentse sem az oktató sem a diák teljesítményét.

## 4.2. Következmények

A kényszerhelyzetben megvalósított tanítási gyakorlat feltárta annak hiányosságait és megmutatta, hogy hogyan érdemes a tanárképzést átalakítani. Ez jelenleg rendkívül aktuális kérdés, mivel a szlovákiai egyetemek ez időben már intenzíven készülnek a módosított tanulmányi programok akkreditálására. A távoktatás alatt felmerült problémákból és a feltárt hiányosságokból le kell vonni a tanulságot. Mindenképpen orvosolni kell a feltárt hiányosságokat, amihez alaposan meg kell fontolni, mit érdemes megváltoztatni a tanárképzésben. Fontos megőrizni a bevált hagyományos képzési alapot, a kiépített értékeket, viszont frissíteni és pontosítani kell a tanítási célokat, a tantárgyak küldetését és tartalmát, figyelembe a véve a TPCK (Technological Pedagogical Content Knowledge) (Mishra – Koehler. 2006) valamennyi aspektusát, a tanárok és tanárjelöltek számítógépes gondolkodásának fejlesztését (Wing, 2006), a számítógépes problémamegoldás hatékonyságának emelését. (Polya, 1954; Csernoch, 2017) Méretre szabottan kell definiálni a tanító és tanárképző tanulmányi programok végzőseinek profilját.

**Mit kell a tanárképzésben megváltoztatni?**

Számítani kell a távoktatással és ennek érdekében

- új tantárgyakat kell bevezetni;
- a meglevő bevált tantárgyak tartalmát változtatni/módosítani/pontosítani kell – időszerűvé kell tenni, beépíteni a számítógépes gondolkodás és a számítógépes problémamegoldási módszerek és stratégiák fejlesztését.

**Milyen témákat, témaköröket kell beiktatni?**

Az, amit feltétlenül be kell vonni a tanító és tanárképző programok közös részébe az eddigi bevált és hagyományos tartalom mellett a következők:

- iskolai információs rendszerek és szolgáltatásaik,
- távoktatás és eszközei,
- modern (mobil) technológiák az oktatásban,
- számítógépes problémamegoldás,
- tananyag feldolgozás online tanításhoz és távoktatáshoz.

## 5. Befejezés

A modern online oktatási technológiák használata a távoktatásban a járvány idején megerősítette azok fontosságát és hasznosságát a válsághelyzetkezelés szempontjából.

A távoktatás, amelyet az iskolák kénytelenek voltak végrehajtani a COVID-19 járvány első hulláma alatt, sok problémát és negatívumot tárt fel, ugyanakkor megmutatta oktatási rendszerünk és maga a szlovák oktatási rendszer pozitívumait is. Ezek közül a legjelentősebbek az alábbiak.

Az iskolák, a pedagógusok és a tanulók nem voltak felkészülve a teljes távoktatásra. Sok iskolának máig sincs megfelelő technikai és technológiai eszköze a távoktatáshoz. A pedagógusoknak, a diákoknak és a tanulóknak saját eszközeiket kellett használniuk – számítógépeket, laptopokat, táblagépeket és mobiltelefonokat. Sokan közülük nem rendelkeztek a szükséges felszereléssel vagy elég gyors internetkapcsolattal.

A szükséghelyzet ellenére a központilag kínált lehetőségeket és a rendelkezésre bocsátott e-learning felületeket nem használták fel kellő mértékben. Az IKT és a modern oktatási technológiák használata és integrálása a tanításba új lehetőségeket kínál a távoktatás területén is.

A távoktatás pozitív előnye, hogy az egyetemi hallgatók új készségeket és képességeket szereztek, és olyan tevékenységeket fejlesztettek ki, amelyek során növelték digitális írástudásukat, és megtanulták kritikusan értékelni a különféle információforrások relevanciáját, valamint az alkalmazott eszközök és technológiák alkalmasságát.

A pedagógusok kénytelenek voltak használni az iskolai információs rendszereket, elektronikusan kommunikálni a tanulókkal/diákokkal (offline és online módon), különféle oktatási rendszereket használni, digitális tananyagokat létrehozni, és általa növelni információs műveltségüket (digitális írástudásukat) és digitális kultúrájukat.

A szülők, és talán a felülrendelt irányító szervek is beismerték, hogy a tanítás nem egyszerű dolog, mivel megkövetel bizonyos mesterséget, hiszen a tanító és tanári státusz nem beosztás, hanem hivatás.

## Javaslatok

Annak érdekében, hogy a tanárképző programokban nemrég végzett hallgatók az iskolai információs rendszerek, a modern didaktikai és oktatási technológiák, valamint a távoktatásra alkalmas szoft-

veralkalmazások és videokonferencia-rendszerek használatára kellően felkészültek legyenek, úgy elméletileg, mint gyakorlatilag, tanulmányi programjaikba új tematikus egységeket kell bevonni, és így kiküszöbölni a feljebb említett hiányosságokat. A jelenlegi tanárképzési programokban a leendő tanároknak már az egyetemi tanulmányok és a kötelező tanári gyakorlat során el kell sajátítaniuk a hiányzó készségeket, képességeket és némi tapasztalatot kell szerezzenek a számítógépes problémamegoldásban. Ezt az akut társadalmi megrendelést az új tanulmányi programok előkészítésekor teljesíteni kell. A szlovák egyetemek összes tanulmányi programját – beleértve a tanárképzési programokat is – a közeljövőben új akkreditációs folyamat várja.

Az iskoláknak át kell gondolniuk azt is, hogy miként biztosítsák a szükséges hardvereszközöket megfelelő szoftverrel azon tanulók/diákok számára, akik nem rendelkeznek semmilyen IKT vagy mobil eszközzel.

Az iskoláknak koordinálniuk és módszeresen kezelniük kell a távoktatást, és helyesen kell megválasztaniuk a távoktatásnál alkalmazott szoftvert. Ügyelni kell arra is, hogy a tanulóknak/diákoknak ne kelljen egyidejűleg több rendszert használniuk. Így egyszerűbbé válhat elsajátítani a kezelését és megismerni a rendszer lehetőségeit és funkcióit.

Csökkenteni kell az általános és középiskolákban, valamint az oktatási intézményekben használt iskolai információs rendszer-típusok számát. Javasoljuk csak azokat használni, amelyek kompatibilisek és összekapcsolhatók az országos tanfelügyeleti információs rendszerrel.

Ugyanúgy csökkenteni kell a távoktatás támogatására és megszervezésére használt szoftvertípusok számát. Egy iskolán belül legfeljebb kettőt használjanak, ha ez indokolt.

Központilag irányítani kell az elektronikus (digitális) tankönyvek és tananyagok gyártását és terjesztését, hogy legalább a legfontosabb tematikus egységeket lefedjék, és ne hozzanak létre nagy átfedéssel (redundanciával) rendelkező tananyagokat.

Szükséges a nemzeti (klasszikus) tankönyveket elektronikus formában rendelkezésre bocsájtani az iskolák, pedagógusok és a tanulók számára is.

**A tanulmányt a KEGA 012TTU-4/2018: Interactive animation and simulation modells in education (Interaktív animációs-szimulációs modellek az oktatásban) és a KEGA 015TTU-4/2018: Interactivity in electronic didactic applications (Interaktivitás az elektronikus didaktikai alkalmazásokban) projektek támogatták.**

## Irodalom

1. Hyksová, H. – Stoffová, V.: Softwarové prostředky na podporu on-line vzdělávání (Software resources to support on-line education". In: *XXXIII DIDMATTECH 2020. Budapest:* Eötvös Loránd University (ELTE), Faculty of Informatics, 2020, s. 98 – 109. ISBN 978-963-489-244-1
2. Csernoch, M.: Thinking Fast and Slow in Computer Problem Solving, Journal of Software Engineering and Applications, Vol.10 No. 01, Article ID:73749, (2017) pp. 30 10.4236/jsea.2017.101002.
3. Mishra P, Koehler MJ. Technological pedagogical content knowledge: A framework for teacher knowledge. Teachers College Record, 108(6), (2006) 1017–1054.
4. Polya, G.: How To Solve It. A New Aspect of Mathematical Method. (2nd Edition 1957), Princeton University Press, Princeton, New Jersey. 1954
5. Pšenáková, I.: Interactive applications in the work of teacher. In: *XXIXth DidMatTech 2016*. Budapest : Eötvös Loránd University in Budapest, Faculty of Informatics, 2016, s. 92 – 100. ISBN 978-963-284-800-6. https://www.mii.lt/informatics_in_education/htm/infedu.2017.07.htm. WOS:000399818000007
6. Stoffová, V.: Využitie IKT v učiteľskej profesii – Tvorba didaktických aplikácií (Use of ICT in the teacher profession – Creation of didactic applications). In: *Sborník konference Didinfo 2018* [online]. Ed. Jindra Dráb-

ková a Jan Berki. Liberec, 2018. s. 153 – 162. ISBN 978-80-7494-424-6, ISSN: 2454-051X. Dostupné z: http://www.didinfo.net/images/DidInfo/files/Didinfo_2018.pdf

7. Stoffová, V. – Gabaľová, V.: IKT vo vyučovaní na prvom stupni základnej školy **(**ICT in first degree of primary school) In: *DidInfo 2019: 21. ročník národnej konferencie o vyučovaní informatiky*. Editori: Dana Horváthová, Alžbeta Michalíková, Jarmila Škrinárová, Patrik Voštinár. Banská Bystrica: Univerzita Mateja Bela, Fakulta prírodných vied, 2019, s. 130 – 129, ISBN 978-80-557-1533-9, ISSN 2454-051X
8. Stoffová, V. – Stoffová, M. – Némethová S.: Development of ICT and distance education skills in teacher training. In. *ICERI 2020* (in press)
9. Stoffová, V.: Školské informačné systémy v učiteľskej príprave (School information systems in teacher training) In: *UNINFOS 2020* (in press)
10. Záhorec, J. – Hašková, A. – Munk, M.: *Digitálna gramotnosť učiteľov v kontexte ich profesijnej prípravy*. Bratislava: UK, 2020 (in press)
11. STOFFOVÁ, V. – CZAKÓOVÁ, K.: Prostredie mikrosveta v práci učiteľa 1. stupňa základnej školy. In. Walat, W.(ed.): EDUKACJA
12. Czakóová, K.: Creation small educational software in the micro-world of small languages. In *Teaching Mathematics and Computer Science*. 14th volume, issue one, 2016/1, p. 117. Debrecen: University of Debrecen, 2016. ISSN 1589-7389.
13. Stoffová, V. – Czakóová, K.: How to prepare and introduce a new subject into the teacher training curriculum. In *Teaching Mathematics and Computer Science*. 14th volume, issue one, 2016/1, p. 127. Debrecen: University of Debrecen, 2016. ISSN 1589-7389.
14. Czakóová, K.: Microworld environment of small language as „living laboratory" for developing educational games and applications. In. Proceedings of the 13th International Scientific Conference „eLearning and Software for Education": Could technology support learning efficiency? Volume 1, DOI: 10.12753/2066-026X-17-042, 2017/1, p. 286-291. Bucharest: "CAROL I" National Defence University Publishing House, 2017. ISSN 2066-026X ISSN-L, 2066-026X, ISSN CD 2343 – 7669. (WoS)
15. Czakóová, K.: Digitalizáció az oktatásban és hatása a társadalomra. In. A Kárpát-medence, mint gazdasági tér: I. Szlovákiai Magyar Közgazdászok Találkozója. Komárom: Selye János Egyetem - Szlovákiai Magyar Közgazdász Társaság, 2018. s. 155-163. ISBN 978-80-8122-
16. Wing, J. M.: Computational thinking. Communications of the ACM, 49(3), 33. DOI= http://doi.org/10.1145/1118178.1118215, 2006
17. Wolfram, C.: The Math(s) Fix: An Education Blueprint for the AI Age. Wolfram Media, Inc. 2020

# A táblázatkezelés és a programozás didaktikai kapcsolata

Törley Gábor[1], Zsakó László[2]

[1]gabor.torley@inf.elte.hu; ORCID: 0000-0002-0496-936,
[2]zsako@caesar.elte.hu; ORCID: 0000-0002-4614-1509
ELTE Eötvös Loránd Tudományegyetem Informatikai Kar

**Absztrakt.** Amikor a problémamegoldás készségéről beszélünk, akkor általában a programozás jut eszünkbe, mint tevékenység, ami fejleszti az algoritmikus gondolkodást, illetve az absztrakciós készséget. Ugyanakkor a táblázatkezelésről első gondolatként a szoftveralkalmazás területe juthat eszünkbe, illetve második gondolatként eszünkbe juthat a matematika. Amikor a táblázatkezelés és a programozás egy lapon szerepel, akkor a makrók programozása kerül előtérbe, ami inkább programozás, mint táblázatkezelés. Cikkünkben azt vesszük górcső alá, hogy miként hat egymásra ez a két terület, illetve fel kívánjuk hívni a figyelmet arra, hogy a táblázatkezelés is hatékony eszköz az algoritmikus gondolkodás fejlesztéséhez, sőt, több „áthallás" van a két eszköz között. Példákon keresztül mutatjuk meg továbbá, hogy a programozás és a táblázatkezelés között két irányú kapcsolat létezik, azaz a kölcsönös egymásra építés mindkét témakör számára hasznos lehet.

**Kulcsszavak:** táblázatkezelés, programozás, problémamegoldás, algoritmikus gondolkodás, tanítási módszerek

## 1. Bevezetés

A táblázatkezelést, mint tanulási eszközt, nem szokás a problémamegoldás eszközei közé sorolni. Ennek ékes példája a NAT 2012-es kerettanterv [1], amelyben a problémamegoldás informatikai eszközökkel és módszerekkel témakör kizárólag a programozás, algoritmizálás területével foglalkozik, és a táblázatkezelés az alkalmazói ismeretek témakörben kapott helyet. E szerint a táblázatkezelés célja az információk kinyerése.

A 2020-as kerettanterv [2] ebből a szempontból árnyaltabb képet fest, ugyanis itt – helyesen –, a táblázatkezelés már a problémamegoldás fejlesztésének új témaköreként jelenik meg. Ezzel követi az ELTE Informatikai Karán sokkal korábban kidolgozott módszertant. [10, 11]

## 2. Irodalmi áttekintés

Szalayné szerint [9] a táblázat egy programnak tekinthető adatokkal és előre meghatározott algoritmusokkal. Bár a tanulók egy táblázatot/táblázatkezelőt látnak, mégis meg kell érteniük az általuk írt „programot", azaz a függvényekkel megvalósított megoldásukat. Ilyen módon táblázatkezelő alkalmazásával lehet indirekt módon programozást tanítani.

Bíró és Csernoch szerint a táblázatkezelő szoftver használható a problémamegoldás ezközeként [3], és az általuk leírt Sprego módszer alkalmas a tanulók számítógépes gondolkodásának és algoritmikus készségének fejlesztésére.

Warren szerint, ha először táblázatkezelőt használunk, és utána a programozási nyelvet, akkor gyorsabban el lehet jutni a tananyagban a bonyolultabb algoritmusokig. [7]

## 3. Változó- és típusfogalom

A táblázatkezeléssel tanítani lehet a skalár, a tömb, a mátrix, valamint az indexelés fogalmát, amelyek alapvető adatszerkezetei a programozásnak, hasonló módon az elemi típusok bevezetését is támogatja a táblázatkezelés témaköre. Haladó lehetőségeivel akár nevet is adhatunk egyes tartományainak, amire névvel hivatkozhatunk (mint egy változóra a programozási nyelvekben).

A típusfogalom kialakításában fontos szerepe van a táblázatkezelésnek, mivel bizonyos műveletek (függvények) csak bizonyos típusú adatokon értelmezhetőek, pl. SZUM, ÁTLAG függvény csak számokon, míg az ÖSSZEFŰZ, FŰZ függvények csak szövegeken értelmezhetők. Ebből a tanuló megértheti, hogy a típus nem csak egy halmazt jelöl, hanem a rajta értelmezett műveleteket is magába foglalja. Ugyanúgy éles különbség van a táblázatkezelésnél a számjegyek (mint szöveg) és a számok között (vö. ”23” és 23 közötti különbség). Az éles különbség a szöveg- és számkonstansok esetében is látszik. (Kitérő: nagy hasonlóságot lehet felfedezni a táblázatkezelők cellái formátumának megadási lehetőségei és a programozási nyelvek formázott kiírási lehetőségei között, pl. hány számjegyre jelenjen meg a szám, hány tizedesjegyre, ...)

Bár a táblázatkezelés változófogalma sajátos [6], mégis, a függvények működésének mélyebb megértése hozzájárul ahhoz, hogy a tanuló megértse, mi a különbség a skalár és a sokaság között, valamint mit jelent bejárni egy sokaságot (jelen esetben tömböt vagy mátrixot). Az indexelés megértésének egyik legjobb eszköze az INDEX fv. lehet, amely lényegében egy általunk kijelölt tartományon hajtja végre az indexelés műveletét.

## 4. Vezérlési szerkezetek, sémaalgoritmusok

A táblázatkezelés ugyan függvényszerű gondolkodást igényel (bevezetés a funkcionális programozásba – ennek kifejtése egy másik cikk témája lehet), de a klasszikus függvényei (SZUM, MAX, ...) egyben alapvető programozási algoritmus sémák (programozási tételek) is. Egyes függvények ráadásul úgy érthetők meg, ha elképzeljük a végrehajtási algoritmusukat (pl. FKERES). A függvények paraméterezése, és egymásba ágyazása segítheti a hagyományos nyelvekbeli paraméterezés, paraméterátadás megértését.

A HA függvénnyel, lényegében, az elágazás, mint vezérlési szerkezet működését lehet megérteni, egyben kiváló eszköz a logikai típus, illetve a logikai műveletek (ÉS, VAGY fv.) megértéséhez. A ciklusok fogalma „nyelvi” szinten nem kerül elő a táblázatkezelés során, viszont a tömbképletek használatával és megértésével bevezethető a számlálós ciklus fogalma. A keresési függvények mélyebb vizsgálata juttathat el a feltételes ciklusokig, mivel ezek megértésénél fel lehet tenni a kérdést, hogy végig kell-e futnunk a sokaságon ahhoz, hogy egyértelműen választ adjunk a keresési feladatra.

Haladó táblázatkezelésnél a tömbfüggvényekkel le lehet képezni az összes sémaalgoritmust (programozási tételt), illetve kapcsolatot lehet találni a tömbképletek és a programozási tételek utófeltétele között.

Az alábbi táblázatban foglaljuk össze a táblázatkezelés és a sémaalgoritmusok kapcsolatát:

| Sémaalgoritmus | Táblázatkezelés megvalósítás |
|---|---|
| Sorozatszámítás (feltételes is) | SZUM, SZUMHA, SZUMHATÖBB, ÁTLAG, ÁTLAGHA, ÁTLAGHATÖBB, AB.SZUM, AB.ÁTLAG, FŰZ, ÖSSZEFŰZ |
| Megszámolás | DARABTELI, DARABHATÖBB, AB.DARAB, AB.DARAB2 |
| Maximumkiválasztás | MAX, MIN |
| Eldöntés | HA(DARABTELI), HA(DARABHATÖBB) |
| Kiválasztás | FKERES, VKERES, XKERES, INDEX(HOL.VAN), AB.MEZŐ |
| Keresés | Eldöntés + Kiválasztás |
| Másolás | Nincs speciális függvény, valójában a cellahivatkozás másolásával alkalmazható (1. ábra) |
| Kiválogatás | Irányított szűrés |
| Feltételes maximum | MAXHA, MINHA |
| K-adik legnagyobb | NAGY, KICSI |
| Rendezés | Rendezés (bár csak a feltétel látszik, a módszer nem) |

**1. táblázat:** Táblázatkezelés és sémaalgoritmusok kapcsolata

A fenti táblázatból kiemelendő az eldöntés, a kiválasztás és a keresés algoritmusok táblázatkezelőben való megvalósítása. Egyrészt a táblázatkezelésnél bemutatható, hogy az FKERES és HOL.VAN függvények lényegében a kiválasztást valósítják meg, hiszen nem adnak értelmes választ akkor, ha nincsen meg a keresett elem. Az eldöntést át kell fogalmazni úgy, hogy létezik-e az adott vagy adott tulajdonságú elem (ez a gondolkodásmód már kapcsolatban áll az eldöntés tétel utófeltételével). Ahogy a programozásban a lineáris keresés tételét a kiválasztás és az eldöntés összeépítéséből vezetjük le, ugyanígy látszik, hogy táblázatkezelésnél is a fenti két tétel összeépítésével lehet megvalósítani a lineáris keresést: pl. HA(DARABTELI()>0;FKERES();"Nincs")

A másolás tételére nincs külön függvény, hiszen a tétel szerint ugyanazt a műveletet hajtjuk végre a sokaság összes elemén, és ez eredmény egy sokaság. A táblázatkezelő „képletmásoló" (valójában hivatkozás másoló) funkciója látványosan mutatja be, hogy a másolás során a képletet, tehát a műveletet is „másolom". Így lényegében a függvényleképezést valósítom meg. Ezt mutatja be az 1. ábra.

| | A | B | C | |
|---|---|---|---|---|
| 13 | 1 | | 2 | =2*A13 |
| 14 | 2 | | 4 | =2*A14 |
| 15 | 3 | | 6 | =2*A15 |
| 16 | 4 | | 8 | =2*A16 |
| 17 | 5 | | 10 | =2*A17 |
| 18 | 6 | | 12 | =2*A18 |
| 19 | 7 | | 14 | =2*A19 |
| 20 | 8 | | 16 | =2*A20 |
| 21 | 9 | | 18 | =2*A21 |
| 22 | 10 | | 20 | =2*A22 |

**1. ábra:** Másolás tételének megvalósítása.

A néhány sémaalgoritmus (más néven programozási tétel) utófeltételének megértésében segíthet a táblázatkezelés. Ehhez a tömbképleteket kell segítségül hívnunk. A táblázatkezelő függvényeivel megvalósított változat sok esetben adja magát (Pl. összegzés, megszámolás, feltételes összegzés, másolás, feltételes másolás). Vegyük például a megszámolás algoritmusát. Ennek az utófeltétele így néz ki:

$$Db := \sum_{\substack{i=1 \\ T(Tömb_i)}}^{N} 1$$

Ahol T jelöli a tulajdonságfüggvényt, N a sokaság (jelen példában tömb) méretét. Jelentése: Ha az adott tömbelem T tulajdonságú, akkor hozzáadok 1-et a Db-hoz. Táblázatkezelőben szó szerint meg lehet valósítani a fenti képletet tömbfüggvénnyel. A nagy szigma jelöli azt, hogy összeadok több elemet, és az alatta levő feltétel azt, hogy mely elemeknél adom hozzá az 1-et az eddigi összeghez. A nagy szigma működését a SZUM, a feltétel-kiértékelést a HA függvénnyel oldja meg a táblázatkezelő, ezért az alábbi képlet megvalósítja a megszámolás tétel utófeltételét: {=SZUM(HA(T(tömb);1;0}
Látható, hogy a SZUM egy 0-ákból és 1-esekből álló tömböt fog összeadni, illetve az is, hogy azoknál a tömbelemeknél lesz 1 a SZUM által összeadandó tömbben, amelyik T tulajdonságú volt.

Két korábbi munkánkban bizonyítottuk [5, 8], hogy az összes programozási tétel visszavezethető a sorozatszámítás tételére. Ebben leírtuk, hogy a sorozatszámításra visszavezetett eldöntés algoritmus egy logikai vektor alapján ad helyes megoldást, ahol a logikai vektor i. eleme igaz, amennyiben a tömb i. eleme T tulajdonságú. Az eldöntés algoritmusának két variánsa van: létezik T tulajdonságú elem a tömbben, illetve a tömb minden eleme T tulajdonságú. Könnyen belátható, hogy az alábbi két tömbfüggvény megvalósítja az eldöntés tételét:

- létezik T tulajdonságú elem: {=VAGY(HA(T(tömbelem);IGAZ;HAMIS))}
- minden elem T tulajdonságú: {=ÉS(HA(T(tömbelem);IGAZ;HAMIS))}

Megjegyezzük, hogy az eldöntés tételét úgy is megközelíthetnénk tömbképlettel, mint ahogy azt a „normál" függvényekkel tettük (DARABTELI, DARABHATÖBB), de ez a gondolkodásmód nem vezetne hatékony algoritmushoz, és nem is tudnánk kötni az utófeltételhez.

A tételek összeépítésének megértésénél is segítséget nyújthatnak a tömbfüggvények, erre jó példa a feltételes maximumkiválasztás, ahol az eldöntés tételének utófeltételét kombináljuk a maximumkiválasztásával, azaz, ha létezik T tulajdonságú elem a tömbben, akkor kiszámoljuk ezen tömbelemek

maximumát:
{=HA(VAGY(HA(T(tömbelem);IGAZ;HAMIS));MAX(HA(T(tömbelem);A1:A10;""));"NINCS")}

Sémaalgoritmusok utófeltétele és tömbfüggvények kapcsolatát a 2. ábra és a 2. táblázat mutatja be. A 2. ábrán a T tulajdonság jelentése: a szám páros.

| | A | B | C | D |
|---|---|---|---|---|
| 1 | 1 | Összegzés | 55 | {=SZUM(A1:A10)} |
| 2 | 2 | Megszámolás | 5 | {=SZUM(HA(MARADÉK(A1:A10;2)=0;1;0))} |
| 3 | 3 | Maximum | 10 | {=SZUM(HA(MAX(A1:A10;A1:A10)=A1:A10;A1:A10;""))} |
| 4 | 4 | Eldöntés (létezik) | IGAZ | {=VAGY(HA(MARADÉK(A1:A10;2)=0;IGAZ;HAMIS))} |
| 5 | 5 | Eldöntés (minden) | HAMIS | {=ÉS(HA(MARADÉK(A1:A10;2)=0;IGAZ;HAMIS))} |
| 6 | 6 | Feltételes összegzés | 30 | {=SZUM(HA(MARADÉK(A1:A10;2)=0;A1:A10;""))} |
| 7 | 7 | Feltételes max | 10 | {=HA(VAGY(HA(MARADÉK(A1:A10;2)=0;IGAZ;HAMIS));MAX(HA(MARADÉK(A1:A10;2)=0;A1:A10;""));"NINCS")} |
| 8 | 8 | | | |
| 9 | 9 | | | |
| 10 | 10 | | | |
| 11 | | | | |

| | E | F | G | H | I | J | K | L |
|---|---|---|---|---|---|---|---|---|
| 1 | Kiválogatás | | {=HA(MARADÉK(A1:A10;2)=0;A1:A10;"")} | Másolás | 2 | {=A1:A10*2} | | |
| 2 | | 2 | | | 4 | | | |
| 3 | | | | | 6 | | | |
| 4 | | 4 | | | 8 | | | |
| 5 | | | | | 10 | | | |
| 6 | | 6 | | | 12 | | | |
| 7 | | | | | 14 | | | |
| 8 | | 8 | | | 16 | | | |
| 9 | | | | | 18 | | | |
| 10 | | 10 | | | 20 | | | |
| 11 | | | | | | | | |
| 12 | | | | Feltételes másolás | 1 | {=HA(MARADÉK(A1:A10;2)=0;A1:A10*2;A1:A10)} | | |
| 13 | | | | | 4 | | | |
| 14 | | | | | 3 | | | |
| 15 | | | | | 8 | | | |
| 16 | | | | | 5 | | | |
| 17 | | | | | 12 | | | |
| 18 | | | | | 7 | | | |
| 19 | | | | | 16 | | | |
| 20 | | | | | 9 | | | |
| 21 | | | | | 20 | | | |

**2. ábra:** Táblázatkezelés példa a sémaalgoritmusok és az utófeltétel kapcsolatára.

Látható, hogy a kiválogatás tételénél – a táblázatkezelő sajátosságai miatt – nem lehetett a tömb folytonosságát megőrizni.

| **Sémaalgoritmus és Utófeltétel** | **Megvalósítás tömbfüggvénnyel** |
|---|---|
| Összegzés (sorozatszámítás) $\sum_{i=1}^{N} Tömb_i$ | {=SZUM(tömb)} |
| Megszámolás $\sum_{\substack{i=1 \\ T(Tömb_i)}}^{N} 1$ | {=SZUM(HA(T(tömb);1;0))} |
| Maximumkiválasztás (egy maximum esetén) $\forall i \in [1..N]: max \geq Tömb_i$ | {=SZUM(HA(MAX(tömb;tömbelem)=tömbelem;tömbelem;""))} |
| Eldöntés (létezik egy megfelelő) van := $\exists i \in [1..N]: T(Tömb_i)$ | {=VAGY(HA(T(tömbelem);IGAZ;HAMIS))} |
| Eldöntés (minden) minden := $\forall i \in [1..N]: T(Tömb_i)$ | {=ÉS(HA(T(tömbelem);IGAZ;HAMIS))} |
| Feltételes összegzés $\sum_{\substack{i=1 \\ T(Tömb_i)}}^{N} Tömb_i$ | {=SZUM(HA(T(tömbelem);tömbelem;0))} |
| Feltételes maximum van := $\exists i \in [1..N]: T(Tömb_i)$ és Van → $1 \leq MaxI \leq N$ és $T(Tömb_{MaxI})$ és $\forall i (1 \leq i \leq N): T(Tömb_i) \rightarrow Tömb_{MaxI} \geq Tömb_i$ ) | {=HA(VAGY(HA(T(tömbelem);IGAZ;HAMIS)); MAX(HA(T(tömbelem);A1:A10;""));"NINCS")} |
| Másolás $\forall i \in [1..N]: F(Tömb_i)$ | {=F(tömbelem)} |
| Feltételes másolás $\forall i \in [1..N]: T(Tömb_i)$ esetén: $F(Tömb_i)$ különben $Tömb_i$ | {=HA(T(tömbelem);F(tömbelem);tömbelem)} |
| Kiválogatás $Db := \sum_{\substack{i=1 \\ T(Tömb_i)}}^{N} 1$ és $\forall i \in [1..Db]: T(Y_i)$ | {=HA(T(tömbelem);tömbelem;""} |

**2. táblázat:** Sémaalgoritmusok utófeltétele és a tömbfüggvények kapcsolata

## 5. Összegzés

A fentiek alapján egyrészt beláthatjuk, hogy a táblázatkezelést (a táblázatformázás és a diagramformázás kivételével) miért sorolják inkább a számítástudományi ismeretekhez (az algoritmizálással, programozással együtt), mint a digitális írástudáshoz. Látható, hogy jelentős részben ugyanarról a problémamegoldási területről szól, mint az algoritmizálás.

A két terület egyes témakörei (adatok, típusalgoritmusok) között is nagy hasonlóságot találhatunk, ami előrevetíti, hogy egymás segítségére lehetnek a fogalmaik tanulásában.

Klasszikus tanítási sorrendben előbb tanulnak a diákok táblázatkezelést, mint típusalgoritmusokat (programozási tételeket), azaz a programozástanítást lehetne a táblázatkezelési ismeretekre építeni [1, 2, 9]. Ebben a cikkben azonban megfogalmaztunk olyan lehetőségeket is (ezek a tömbfüggvények), amelyek a klasszikus sorrendben sem előzik meg a hozzájuk illeszkedő programozástanulást, ha egyáltalán megjelennek a tantervekben – tömbfüggvényekkel legfeljebb a tehetséggondozásban foglalkoznak [4].

## Irodalom


1. *A 2012-es NAT-hoz illeszkedő Informatika kerettanterv* (2012) https://kerettanterv.oh.gov.hu/05_melleklet_5-12/5.2.21_informat_5-10.doc (utoljára megtekintve: 2020.11.11.)
2. *A 2020-as NAT-hoz illeszkedő tartalmi szabályozók* (2020) https://www.oktatas.hu/kozneveles/kerettantervek/2020_nat (utoljára megtekintve: 2020.11.11.)
3. Biró P., Csernoch M.: *Algoritmusok és/vagy táblázatkezelés?* In: Ujhelyi Adrienn, Lévai Dóra (ed.): VII. Oktatás-Informatikai Konferencia Tanulmánykötet 2015, Budapest (2006) pp. 97–111
4. Molnár K.: Tehetséggondozás az informatikában – Táblázatkezelés. ELTE Informatikai Kar, 2014, http://tehetseg.inf.elte.hu/tananyagok/tablazatkez/index.html (utoljára megtekintve: 2020.11.11.)
5. P. Szlávi, G. Törley, L. Zsakó: *Programming theorems have the same origin* In: Veronika Stoffová, Román Horváth (eds.) New methods and technologies in education and practice: XXX. DIDMATTECH 2017. pp. 52-58 Trnava, 2017. ISBN 978-80-568-0029-4
6. P. Szlávi, G.Törley, L. Zsakó: *The most difficult notion of programming: The variable*, In: Elzbieta Sałata; Agata Buda - Education - Technology - Computer Science in Building better future Radom, Lengyelország: Wydawnictwo Uniwersytetu Technologiczno-Humanistycznego w Radomiu, (2018) pp. 108-118., 11 p.
7. P. Warren: *Learning to program: spreadsheets, scripting and HCI*, in Proceedings of the Sixth Australasian Conference on Computing Education – vol. 30, Darlinghurst, Australia, (2004) 327–333.
8. Szlávi P., Törley G., Zsakó L.: *Az összetett programozási tételek is egy tőről fakadnak* In: Szlávi Péter, Zsakó László (szerk.) INFODIDACT 2017. Konferencia helye, ideje: Zamárdi, Magyarország, 2017.11.23-2017.11.25. Budapest: Webdidaktika Alapítvány, 2017. Paper 21. (ISBN:978-615-80608-1-3)
9. Zs. Szalayné Tahy: *How To Teach Programming Indirectly – Using Spreadsheet Application.* Acta Didactica Napocensia 9 (1), 15-22, 2016. pp. 15-22. ISSN 2065-1430
10. Zsakó L.: Informatika Nemzeti Alaptanterv 2020. In: Szlávi Péter, Zsakó László (szerk.) INFODIDACT 2015. Konferencia helye, ideje: Zamárdi, Magyarország, 2015.11.26-2015.11.27. Budapest: Webdidaktika Alapítvány, 2017. Paper 1. (ISBN: 978-963-12-3892-1)
11. Zsakó L.: Informatikai tantervelmélet? Diszciplínák tanítása – a tanítás diszciplínái 1. Tanulmányok a tudós tanárképzés műhelyeiből. ELTE Eötvös Kiadó, 2015. pp. 92-111. (ISBN 978-963-284-611-8)

# JavaScript könyvtárak programozott rajzolás alapú tanulás támogatásához

Visnovitz Márton[1], Horváth Győző[2]

{[1]visnovitz.marton, [2]horvath.gyozo}@inf.elte.hu
ELTE Eötvös Loránd Tudományegyetem, Informatikai Kar

**Absztrakt.** Több kutatás foglalkozott korábban a JavaScript programozási nyelv oktatásban való használhatóságával, annak előnyeivel, hátrányaival. Egy korábbi kutatásunkban ennek a kérdéskörnek egy szűkebb területét, a böngészőben történő programozott, Canvas API alapú grafikát, és az arra épített konstrukcionista módszertan lehetőségeit vizsgáltuk. Kutatásaink és tapasztalataink feltárták a platform és a technológia több olyan hiányosságát, melyek javításával még hatékonyabb oktatási, tanulási eszközt kaphatunk. Ennek a cikknek a célja ezen problémák, illetve az ezen problémákat orvosló JavaScript eszközök bemutatása.

**Kulcsszavak:** JavaScript, programozott grafika, animáció, szimuláció, játékkészítés, Canvas API

## 1. Bevezetés

Az internet elterjedésével egyre több tanítási környezet jelent meg webes alkalmazásként a böngészőben, amelyeknek egy része a programozásoktatást tűzte ki célul ezen a viszonylag új platformon [1]. Kezdetben adta magát, hogy ezt a célt a böngészőben natívan futó programozási nyelv, a *JavaScript* segítségével érjék el, hiszen ehhez semmilyen plusz erőforrásra nem volt szükség sem a kliens, sem a szerver oldalon. Nem kellett a felhasználónak kiegészítőket telepítenie a böngészőjében, és nem kellett komoly futtató infrastruktúrát kialakítania a fejlesztőknek a szerveroldalon. Nem véletlen tehát, hogy az eleinte megjelenő webes programozásoktató alkalmazások jelentős része JavaScripten keresztül ismertette meg az alapokat a tanulókkal.

E programozási nyelvi választást segítette az is, hogy a JavaScript tervezéséből adódóan egy könnyen használható nyelv, amely kellően rugalmas a használatát illetően, és sokféle programozási paradigmát támogat, így az oktatási anyagok készítői könnyen a saját elképzeléseikhez tudták idomítani a tanítandó koncepciókat és azok formáit. Az alap nyelvi elemek és szolgáltatások fölé viszonylag egyszerűen lehetett egy plusz absztrakciós réteget kialakítani, így egy olyan eszközkészlet definiálása, amely a feladatok megoldásához szükséges, nem kívánt nagy erőfeszítést a fejlesztők részéről.

A JavaScript programozási nyelv nemcsak a webes oktatóplatformokon van jelen mint a programozásoktatás eszköze, hanem a felsőoktatásban is megjelent a bevezető programozástanításban [2,3,4]. Léteznek már egészen fiatal korosztálynak szóló könyvek is, melyek a JavaScriptet használják a programozás bevezetésére [5], de az amerikai Stanford egyetem bevezető programozási kurzusa[22] is a JavaScript nyelvet használja, kiegészítve egy tanulási célokat támogató absztrakciós réteggel.

Az idő előrehaladtával, és a webes platform fejlődésével egyre több klasszikus környezet is meg tudott jelenni a weben [6], így ma már számos példát találunk böngészőben futó teknőcgrafikára (pl. Python with turtle[23] a Repl.it[24]-en[25]), blokk-alapú programozásra (pl. Scratch[26]), hagyományos okta-

[22] https://web.stanford.edu/class/cs101/
[23] https://docs.python.org/3/library/turtle.html
[24] https://repl.it/

tási célú és ipari felhasználású nyelvek használatára (pl. Repl.it), vizuális programozásra, eseményvezérelt felhasználói felületek kialakítására (pl. Code.org).

## 2. A grafikus programozás lehetőségei JavaScript nyelven

A mai világban a hagyományos, matematikai problémákon keresztül bevezetett programozás mellett nagyon nagy szerepet kapnak a különféle vizuális környezetek a programozás oktatásában. Egyre több programozást oktató rendszer, tananyag vezeti be a programozás fogalmait valamilyen programozott grafika, animáció vagy játékkészítés révén. Ezen programozásoktatási stratégiák hatékonyan támogatják a programozás alapfogalmainak elsajátítását, miközben a vizuális aspektus révén nagy teret engednek a kreativitásnak és magasan tartják a tanulók motivációját [7,8]. Sok környezet a JavaScript programozási nyelvet használja az ilyen jellegű grafikus programok készítéséhez [9]. A böngésző, a HTML nyelv és a JavaScript több, remek eszköztárral rendelkezik grafikák készítéséhez.

Raszteres grafikákat a böngésző Canvas API-ján keresztül készíthetünk alacsony szintű, de intuitív műveletek segítségével. A HTML nyelv `canvas` (vászon) eleme[27] egy olyan programozható rajzvásznat biztosít, melyen JavaScript segítésével tudunk úgynevezett „útvonalakat" definiálni vonalak és egyszerű alakzatok (téglalap, kör(ív), ellipszis) segítségével. Ezen útvonalaknak tudjuk adott stílusú vonallal megrajzolni a pályáját, valamint kitölthetjük őket tetszés szerinti színnel. Ez a módszer egy *imperatív* stílusú, programozott grafika készítését teszi lehetővé. Ez a fajta imperatív, programozott stílus közel áll a klasszikus, algoritmikus programozási ismeretekhez, lehetővé teszi, hogy annak ismereteit könnyen alkalmazzuk/tanítsuk grafikák készítésén keresztül.

Egy másik lehetőség a böngészőben a vektoros grafikák készítése. Ehhez a HTML nyelv beágyazott SVG támogatása ad kiváló eszközt. Amennyiben a tanuló korábban már találkozott a HTML nyelvvel a szintaxis ismerős lesz számára, csak a különböző alakzatok neveit, illetve a beállításokat szolgáló attribútumokat kell megismerni. Ezzel a *deklaratív* megközelítéssel könnyen összeköthetővé válik a böngészőben történő statikus ábrák készítése más ismeretkörökkel, például weboldalak készítésével a HTML nyelv révén vagy a (vektor)grafikai ismeretekkel az SVG formátum és a vektorgrafika elmélete révén. Ilyen formán ez a fajta megközelítés inkább az alkalmazói ismeretek oktatásához köthető.

Habár mindkét megközelítésnek vannak előnyei oktatási szempontból, mégis inkább a Canvas API alapú megoldások a népszerűek a webes oktatási platformok tananyagaiban.

### 2.1. Animációk, játékok készítése a böngészőben

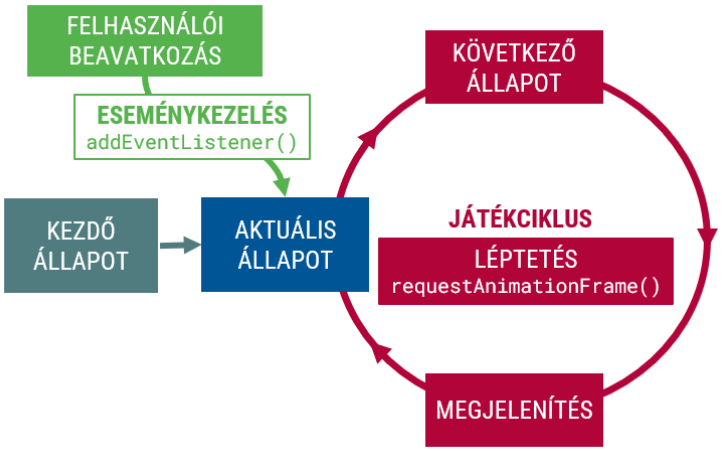
A böngésző azon túl, hogy lehetőséget biztosít statikus ábrák programozott készítésére a Canvas API-n keresztül, illetve vektorgrafikus ábrák készítésére az SVG dokumentumok beágyazása révén, rendelkezik egy olyan programozási eszköztárral is, mellyel könnyedén készíthetők animációk, szimulációk és játékok is natív eszközök segítségével [10].

Az animációk, szimulációk készítésének alapja az ábra folyamatos változása. Ez a fajta folyamatos változás megvalósítható a már meglévő elemek mozgatásával (pl. egy SVG grafika esetén) vagy az ábra gyors és folyamatos újrarajzolásával. Ez utóbbi módszert jellemzően a Canvas alapú grafika esetén használjuk. A grafika ilyen jellegű folyamatos változtatására a böngésző biztosítja a fejlesztő számára a `requestAnimationFrame` függvényt[28], mely az adott számítógép erőforrásainak megfele-

[25] https://turtleacademy.com/
[26] https://scratch.mit.edu/
[27] https://developer.mozilla.org/en-US/docs/Web/API/HTMLCanvasElement
[28] https://developer.mozilla.org/en-US/docs/Web/API/window/requestAnimationFrame

lően, lehetőség szerint a képernyő képfrissítési gyakoriságához igazítva képes egy függvény periodikus meghívására, és ebben az újrarajzolást elvégezve az ábra folyamatos változtatására.

Animációk, szimulációk és játékok készítésekor, mint minden más webes alkalmazás esetén, a megjelenés mellett a másik két fontos komponens: az állapot tárolása, illetve az események kezelése [11]. Az adatok (állapot) tárolása és kezelése a megjelenéstől (grafikától) teljesen függetlenül, alapvető nyelvi elemek (változók, függvények) segítségével megvalósítható. Az események kezelésére egyszerű megoldást biztosít a böngésző mint programozási platform az `addEventListener` függvény/metódus révén, hiszen az eseménykezelés minden böngészőben futó JavaScript programnak alapvető részét képezi. Ezen elemek segítségével könnyen felépíthető egy animációs-/játékciklus, mely működtetni képes a programunkat. Minden ciklusléptésben módosítjuk az állapotteret, ami alapján ugyanebben a lépésben megtörténik a kirajzolás. Az eseménykezelőkben lefutó logika pedig ugyancsak az állapotteret módosítja, amely hatása a kirajzolás során azonnal meg is jelenhet.




**1. ábra:** Az animációs-/játékciklus sematikus ábrája [12]

## 2.2. Canvas API használatának előnyei és hiányosságai

A böngésző és a Canvas API egy kiváló eszköz programozott grafikák, animációk és játékok készítésére. A webes platform ezen fajta sokoldalúsága lehetőséget nyújt a feladattípus-orientált programozásoktatás több stratégiájának alkalmazásához [7,8] a böngészőn belül a JavaScript programozási nyelv segítségével. A platform különböző stratégiák alkalmazása révén akár a bevezető programozástól kezdve is alkalmas lehet a programozás oktatására [2]. Ezt erősítik meg az ELTE Informatikai Karának nyári táborában szerzett tapasztalataink is. A programozás Canvas API alapú tanítására a konstrukcionista neveléselmélet elveit alkalmazva kialakítható egy projektközpontú, élményalapú tanulási környezet [10]. Ebben a tanulási környezetben nem szükséges semmilyen programozási előismeret és egy programozási nyelv segítségével lehetőségünk van az alapoktól akár a komplexebb játékok készítéséig eljutni, út közben számos programozási koncepciót megismerve.

A környezet használatának további előnye, hogy nem igényel semmilyen speciális szoftvert, csupán egy böngészőprogramra és egy szövegszerkesztőre (akár jegyzettömb) van szükség a munkához. Mivel a böngészőben natív technológiákat használunk, ezért akár webes szerkesztőkkel is könnyen integrálható a technológia. Akár teljeskörű online integrált fejlesztőkörnyezetekben (pl. Repl.it) is könnyen lehetséges a Canvas alapú programozás oktatása.

A Canvas API alapú programozás hátrányai között megemlíthetjük, hogy a natív Canvas eszközkészlet viszonylag alacsony szintű, az alapvetően elérhető formák száma kicsi. Habár sok minden

beállítható a környezetben, de ezek a beállítások sok apró műveleten keresztül érhetőek el. Animációkat és játékokat is könnyen készíthetünk ezzel a technológiával, de néhány funkció és haladóbb lehetőség csak körülményesen valósítható meg ebben a környezetben. Ezek közé tartozik például, hogy ha precízen akarjuk kezelni a mozgó elemek sebességét, akkor nekünk kell követnünk és számolnunk az animáció/játék során az eltelt időt, illetve az is, hogy a vásznak létrehozása, illetve azok rajzolási kontextusához[29] való hozzáférés megkívánja minimális mértékben a HTML kód szerkesztését, illetve a JavaScript oldalról némi, a lényeghez közvetlenül nem tartozó kód írását is.

# 3. Külső könyvtárak programozott grafika támogatására

Az egyik első online programozás-oktatási platform, a Khan Academy[30] programozásba bevezető kurzusa[31] is grafikus elemek JavaScriptben történő programozásával épül fel. Ehhez egy oktatási célú Canvas könyvtárat, a Processing.js[32]-t használja. Számos más webes oktatási platform (pl. Udacity, Udemy, Coursera, Codacademy) szintén rendelkezik a csak nyelvi elemeket oktató kurzusok mellett Canvas API alapú tananyagokkal. Ezek egy része a natív utasításokat tanítja, de vannak olyanok is, amelyek (játék)keretrendszereket használnak.

A külső könyvtárak használata felvet egy újabb kérdést is: hogyan töltsük be a külső könyvtárat a programunkba. A JavaScript nyelv erre több lehetőséget is biztosít, de a leggyakoribb megoldás a HTML kódban külön `<script>` tagben megadni a külső könyvtár elérési útját. Ez lehet egy helyi fájl is, de gyakoribb, hogy valamilyen online forrásból (pl. CDN) töltjük be a függőséget. Ezt a megoldást használja a p5.js[33] könyvtár online homokozója[34] is, mely alapértelmezetten el is rejti a HTML fájlt, melyben ez a kódrészlet található, és csak a felhasználó által írt kódot helyezi fókuszba.

## 3.1. Magas szintű absztrakciót használó könyvtárak

A technológia előző fejezetben bemutatott hiányosságinak orvoslására sok platform használ valamilyen a Canvas API-re alapuló, oktatási célú vagy vizualizációs függvénykönyvtárat, mely magasabb absztrakciós szinten megfogalmazott eszközkészletet definiál. Ilyen függvénykönyvtár például a korábban említett Processing.js vagy annak továbbfejlesztése a p5.js. Ezek a könyvtárak a legtöbb esetben egy vastagabb, magas szintű absztrakciós réteget építenek a Canvas API felé, elfedve ezzel a böngészőben elérhető natív műveleteket, teljesen új eszközkészletet definiálva.

A Processing.js (illetve az azt felváltó p5.js) oktatási célból talán a legkiemelkedőbb Canvas API-re épülő függvénykönyvtár. Többek között éppen az hívta életre, hogy a programozás grafikus élményen keresztüli bevezetéséhez egy kellően egyszerű, jól használható alapot adjon, hogy a tanuló a feladatra és ne a környezet jellegzetességeire koncentrálhasson. Ennek megfelelően számos azonnal elérhető függvényt definiál, illetve vannak olyan speciális függvényei is, amelyek külön szolgáltatásokat biztosítanak. Így például, ha implementálunk egy `draw` nevű függvényt, akkor azt a rendszer egy animációs ciklusban hívogatja meg. Az oktatási profil mellett erős hangsúlyt kap a kreatív oldal is, így kulturális projektekben, illetve adatvizualizációk során is használják ezt a függvénykönyvtárat.

2018-tól a Processing.js-t felváltotta a p5.js nevű függvénykönyvtár, ami a nagy elődnyomdokain elindulva továbbra is az élményalapú programozást vallja magáénak. A böngésző rasztergrafikai

[29] https://developer.mozilla.org/en-US/docs/Web/API/CanvasRenderingContext2D
[30] https://www.khanacademy.org/
[31] https://www.khanacademy.org/computing/computer-programming/programming
[32] https://github.com/processing-js/processing-js
[33] https://p5js.org/
[34] https://editor.p5js.org/

képességein túllépve azonban számos egyéb izgalmas elemhez is magas szintű támogatást nyújt, így például DOM programozáshoz, hang- és videóelemek programozásához, 3D-s animációkhoz. Egy sokkal szélesebb és a modern böngészők lehetőségeit sokkal jobban kihasználó palettát kínál fel a webes programozásoktatáshoz (és természetesen egyéb vizualizációkhoz).

Az általános célú, Canvas API-ra épülő függvénykönyvtárak mellett léteznek kifejezetten interaktív 2D animációk készítésére készített függvénykönyvtárak is, melyek hatékonyan használhatók az oktatásban is [13]. A magas szintű absztrakciót használó könyvtárak körében érdemes még megemlíteni az amerikai Stanford Egyetem bevezető CS101 kurzusán[35] használt „JavaScript változatot" is. Ez a könyvtár egy egészen magas szintű absztrakciót vezet be a JavaScript nyelv felett, mely gyakorlatilag a böngészőben megtalálható összes elemet elfedi. Ez a kiegészítése a JavaScriptnek kifejezetten ehhez a kurzushoz készült, célja, hogy a hallgatók ne a nyelvi, hanem az algoritmikus elemekre fókuszáljanak a programozás kezdő lépéseiben.

## 3.2. Az alacsony szintű absztrakció lehetőségei

Lehetséges azonban olyan függvénykönyvtár készítése is, mely fókuszáltan a platform és a technológia hiányosságainak kijavítását célozza meg, miközben minél nagyobb mértékben támaszkodik a platform nyújtotta *natív* lehetőségekre. Ebben az esetben nem kapunk nagyobb eszköztárat, csupán az alapból elérhető eszközök használatát tesszük kényelmesebbé, a komplexebb eszközöket magunknak kell felépíteni (pl. összetett alakzatok rajzolásához érdemes saját – paraméterezett – függvényeket létrehozni, melyek segítségével azok megrajzolása könnyebbé válik). Az eszköztár ilyen jellegű, saját magunk által történő bővítése is hozzájárulhat a tanulási folyamathoz, hasonlóan, mint Logo környezetben a saját eljárások bevezetése sorminták rajzolásához.

Kutatásunk során egy ilyen függvénykönyvtár megvalósíthatóságát vizsgáltuk, majd el is készítettük annak prototípusát. Oktatási tapasztalataink, illetve a táborainkban résztvevő diákok visszajelzése alapján meghatároztuk a két legfőbb hiányosságát a natívan elérhető eszközkészletnek, melynek javításával reményeink szerint minőségi javulás érhető el a tanulási/tanítási folyamatban.

### 3.2.1. Saját függvénykönyvtár megvalósítása

Saját függvénykönyvtárunk prototípusának megvalósításakor törekedtünk arra, hogy minél vékonyabb legyen az absztrakciós réteg a Canvas API felett amellett, hogy a legfontosabb hiányosságokat orvosoljuk. Ennek megfelelően két új absztrakció (osztály) került bevezetésre:

1. A `Canvas` osztály egyesíti, és egyetlen objektumon keresztül teszi elérhetővé a böngészőben a `<canvas>` HTML elem, illetve annak „2D" rajzolási kontextusának műveleteit. Ezen új osztály bevezetésével szükségtelenné válik a vászon (vagy vásznak) létrehozásakor a HTML kód szerkesztése, illetve a rajzolási kontextust elkérő JavaScript kód is (2. és 3. ábra).

```
<!-- HTML -->
<canvas></canvas>

// JavaScript
const canvas = document.querySelector("canvas");
const context = canvas.getContext("2d");
```

**2. ábra:** A `<canvas>` elem és annak rajzolási kontextusának elérése natív eszközök segítségével

```
// JavaScript
const canvas = new Canvas();
```

[35] https://web.stanford.edu/class/cs101/

**3. ábra:** A rajzvászon elérése a `Canvas` osztállyal

Az új osztályon keresztül lehetőség van hozzáférni a vászon elem fontosabb tulajdonságaihoz (szélesség, magasság, stílusok), illetve a rajzolási kontextus műveleteihez is. Ez a megoldás amellett, hogy kényelmesebb használtot tesz lehetővé, biztosítja, hogy a natívan rendelkezésre álló teljes fejlesztői eszköztár továbbra is rendelkezésre áll. A Canvas objektum konstruktora ezen túl lehetőséget nyújt a vászon kezdő méreteinek beállítására is.

```
// A canvas elem tulajdonságának módosítása
canvas.width = 500;

// A rajzolási kontextus műveletének meghívása
canvas.fillRect(0, 0, 100, 100);
```

**4. ábra:** A `<canvas>` elem egy tulajdonságának és a rajzolási kontextus egy műveletének elérése a Canvas objektumon keresztül

A Canvas osztály megvalósítására a JavaScript nyelvben található Proxy típust használtuk fel. Ez teszi lehetővé, hogy a `Canvas` osztály automatikusan döntse el, hogy egy adott metódus vagy tulajdonság a Canvas HTML elemhez vagy a rajzolási kontextushoz tartozik-e. Ahhoz, hogy a HTML kódot ne kelljen szerkeszteni új vászon létrehozásakor, a konstruktor automatikusan létrehozza a szükséges HTML elemet és azt be is szúrja a dokumentumba.

2. A `Timer` osztály egyesíti a böngésző `requestAnimationFrame` és `setInterval` műveleteinek lehetőségeit és interfészt biztosít egy adott függvény adott időközönként történő futtatására úgy, hogy a háttérben a hatékony `requestAnimationFrame`-et használja. Lehetőséget ad automatikus (képernyő frissítési gyakoriságnak megfelelő) időközönkénti, illetve a felhasználó által megadott adott időközönkénti futtatásra is. Az időzítő leállítható és újraindítható, illetve van lehetőség az időzítő frissítési gyakoriságát állítani az `interval` tulajdonságon keresztül. A `Timer` osztály ezen túl automatikusan átadja paraméterként az előző futás óta eltelt időt a paraméterként megadott függvénynek (4. és 5. ábra), ezáltal könnyen készíthetünk realisztikus, a ténylegesen eltelt időtől függő mozgásokat.

```
// JavaScript
function update(dt) { /*...*/ }

let lastFrameTime = performance.now();
function next() {
  const currentTime = performance.now();
  const elapsedTime();
  update(dt);
  lastFrameTime = currentTime;
  requestAnimationFrame(next);
}
```

**5. ábra:** A képfrissítések között eltelt idő manuális követése natív eszközökkel

```
// JavaScript
function update(dt) { /*...*/ }

const timer = new Timer(update);
timer.start();
```

**6. ábra:** A képfrissítések között eltelt idő követése a `Timer` osztállyal

## 4. Összefoglalás, további célok

Elmondható, hogy a böngészőben történő grafikus programozás – hiányosságai ellenére – egy jó eszköz lehet a programozás oktatásában, mint ahogy azt a számos online platform gyakorlata, illetve saját oktatási tapasztalataink mutatják. A technológia hiányosságának kiküszöbölésére a leggyakoribb módszer valamilyen absztrakciós réteg használata, mely a natív eszközök fölött definiál saját eszköztárat. Ezen absztrakciós rétegeket jellemzően valamilyen külső programkönyvtár formájában valósítják meg. Az absztrakciós réteg „vastagságától" függően beszélhetünk magas szintű (vastag), illetve alacsony szintű (vékony) absztrakciós rétegekről. Ezek tulajdonságait az 1. táblázat foglalja össze.

| **Magas szintű absztrakciós függvénykönyvtár** | **Alacsony szintű absztrakciós függvénykönyvtár** |
|---|---|
| többnyire saját eszközkészlet | többnyire a platform natív eszközkészlete |
| több koncepcionális elem elrejtése | koncepcionális elemek elrejtése minimális |
| kifejezetten oktatási célú (is lehet) | az natív eszközkészlet alapvetően nem oktatási célú, de oktatásra is jól használható, a kiegészítések az oktatási célt szolgálják |

**1. táblázat:** Magas- és alacsony szintű absztrakciót használó függvénykönyvtárak összehasonlítása

Az alacsony szintű absztrakciós réteget biztosító függvénykönyvtár-prototípus, melyet a kutatásunk keretében készítettünk igazolja, hogy lehetséges a Canvas API használatakor felmerülő legfontosabb problémákat orvosolni egy ultravékony programozási interfész segítségével. Természetesen ez a prototípus jelenlegi formájában még nem alkalmas oktatásban történő éles alkalmazásra, további fejlesztések szükségesek.

A könyvtár kódjának letisztázása és további fejlesztése mellett az oktatásban történő használathoz szükséges egy jól használható dokumentáció készítése, valamint oktatási segédanyagokra, tananyagokra is szükség van. A hosszabb távú tervek között szerepel további feladattípus-orientált tanítási stratégiák támogatásának kialakítása további alacsony szintű könyvtárak segítségével. Ilyen például az automataelvű teknőcgrafika (ld. Logo[36]) és ügynök-vezérelt szimulációk (ld. NetLogo[37]) támogatása.

## Irodalom

1. Hováth, G., L. Menyhárt, *Oktatási környezetek vizsgálata a programozás tanításához*, (2014).
2. Horváth, G., L. Menyhárt, *Teaching introductory programming with JavaScript in higher education*, in *Proceedings of the 9th International Conference on Applied Informatics*, (2015), pp. 339–350, DOI: 10.14794/icai.9.2014.1.339.
3. Horváth, G., L. Menyhárt, L. Zsakó, *Viewpoints of Programming Didactics at a Web Game Implementation*, *DIDMATTECH 2016*, (2016).
4. Mercuri, R., N. Herrmann, J. Popyack, *Using HTML and JavaScript, in Introductory Programming Courses*, Philadelphia, PA, (1998).
5. Morgan, N., *JavaScript for Kids*. No Starch Press, (2014).

[36] https://el.media.mit.edu/logo-foundation/what_is_logo/logo_programming.html
[37] https://ccl.northwestern.edu/netlogo/

6. Horváth, G., *A web-based programming environment for introductory programming courses in higher education*, *ANNALES MATHEMATICAE ET INFORMATICAE*, vol. 48, pp. 23–32, (2018).
7. Bernát, P., L. Zsakó, *Methods of Teaching Programming - Strategy*, *XXXth DIDMATTECH 2017*, pp. 40–51, (2017).
8. Bernát, P., L. Zsakó, *Programozás tanítási módszerek − stratégia a kezdetekre*, in *INFODIDACT 2019*, (2019).
9. Wu, P., *Teaching Basic Game Programming Using JavaScript*, (2009).
10. Visnovitz, M., G. Horváth, *A Constructionist Approach to Learn Coding with Programming Canvases in the Web Browser*, *CONSTRUCTIONISM 2020*, pp. 1–8, (2020).
11. Visnovitz, M., G. Horváth, *The Web - A Platform for Creation*, *CONSTRUCTIONISM 2018*. Vilnius, (2018).
12. Horváth, G., M. Visnovitz, *A böngésző mint alkalmazásfejlesztési platform*. ELTE Informatikai Kar: Budapest, (2018).
13. Végh, L., J. Udvaros, *Possibilities of creating interactive 2d animations for education using html5 canvas javascript libraries*, in *eLearning and Software for Education Conference*, (2020), pp. 269–274, DOI: 10.12753/2066-026X-20-119.