ЗАТВЕРДЖЕНО
Вченою радою ЗУІ
Протокол №2 від „28”лютого 2024 р.
Ф-КДМ-3

Міністерство освіти і науки України

Закарпатський угорський інститут ім. Ференца Ракоці ІІ

Кафедра математики та інформатики

Реєстраційний №_____

Кваліфікаційна робота

Використання комп’ютерних ігор при вивченні математики

Балог Крістіан Золтанович

Студент IV-го курсу

Освітня програма «Середня освіта (Математика)»

Спеціальність 014 «Середня освіта (Математика)»

Рівень вищої освіти: бакалавр

Тема затверджена на засіданні кафедри

Протокол № 3 / 2024

Науковий керівник: Головач Йожеф Ігнацович

доктор технічних наук, професор,

професор кафедри математики та інформатики

Завідувач кафедрою математики та інформатики: **Кучінка Каталін Йожефівна**

(к. ф.-м. н, доцент)

Робота захищена на оцінку _____, «___» _____ 202_ року

Протокол № _____ / 202_

ЗАТВЕРДЖЕНО
Вченою радою ЗУІ
Протокол №2 від „28”лютого 2024 р.
Ф-КДМ-3

Міністерство освіти і науки України
Закарпатський угорський інститут ім. Ференца Ракоці ІІ

Кафедра математики та інформатики

Кваліфікаційна робота

Використання комп’ютерних ігор при вивченні математики

Рівень вищої освіти: бакалавр

Виконавець: студент IV-го курсу

Балог Крістіан Золтанович

освітня програма «Середня освіта (Математика)»

спеціальність 014 «Середня освіта (Математика)»

Науковий керівник: Головач Йожеф Ігнацович
доктор технічних наук, професор,

професор кафедри математики та інформатики

Рецензент: Стойка Мирослав Вікторович

доцент;

кандидат фізико-математичних наук,

Берегове
2025

Зміст

ЗАТВЕРДЖЕНО
Вченою радою ЗУІ
Протокол №2 від „28”лютого 2024 р.
Ф-КДМ-3

Ukrajna Oktatási és Tudományügyi Minisztériuma
II. Rákóczi Ferenc Kárpátaljai Magyar Főiskola

Matematika és Informatika Tanszék

A SZÁMÍTÓGÉPES JÁTÉKOK ALKALMAZÁSA A MATEMATIKAI OKTATÁSBAN

Szakdolgozat

Készítette: Balog Krisztián

IV. évfolyamos matematika

szakos hallgató

Témavezető: Holovács József

műszaki tudományok doktora, professzor, Matematika és Informatika Tanszék professzora

Recenzens: Sztojka Miroszláv

docens;

fizika és matematika tudományok kandidátusa

Beregszász

2025

Tartalomjegyzék

Bevezetés

Napjainkban az oktatás területén egyre nagyobb szerepet kap a technológia és számítógép alkalmazása a tanulás folyamatában. Annak érdekében, hogy a diákok motiváltabbak és aktívabbak legyenek a tanórákon a hagyományos oktatás mellett, fontos az interaktív, akár számítógépes oktatás alkalmazása is. A matematika két fő részét, a logikus gondolkodást és a problémamegoldást egyaránt fejlesztik az interaktív számítógépes matematikai játékok.

Ezen szempontokból kifolyólag döntöttem úgy, hogy a diplomamunkámban a számítógépes játékok matematikai oktatásban betöltött szerepét fogom vizsgálni. Ennek megvalósításához pedig a Python programozási nyelvet és annak egyik legegyszerűbben átlátható crossplatformos keretrendszerét a Kivyt fogom alkalmazni.

A diplomamunkám célja, hogy rávilágítsak, hogyan is tudják a számítógépes játékok támogatni a matematikai oktatást, megtartva a diákok motivációját, fenn tartva érdeklődésüket ;fejlesztve a logikus gondolkodás és problémamegoldó képességeiket.

1. fejezet

Számítógépes játékok alkalmazása a matematikai oktatásban .

1.1. Fogalmi keret és történeti áttekintés

Ez az alfejezet két forrás, Luk'ianenko, K. [8] és a MindResearch Blog [10] tanulmányainak együttes feldolgozásán alapul.

A számítógépes didaktikai játékok a tanítás egyedülálló formáját képviselik, amelyek lehetőséget biztosítanak a tanulók érdeklődésének növelésére, valamint információs kultúrájuk kialakítására, ami a modern személyiség elengedhetetlen követelménye .A kutatók szerint a megfelelően megszervezett oktatási játék elősegíti a keresést, fejleszti a logikus gondolkodást, a képzelőerőt és a reakciósebességet .

Ebben a fejezetben definiáljuk a játék alapú tanulás és a gamifikáció fogalmait, valamint bemutatjuk a köztük lévő különbségeket. Ezt követően áttekintjük az oktatási játékok történeti fejlődését .

A játék alapú tanulás (game based learning) és a gamifikáció (gamification)sokszor összemossák egymással, pedig a tanulási élményben épített játékelemek megközelítése mind elméletileg, mind gyakorlatban különbözik. Bár a két definíció a tanulás és a játék kombiációját jelenti, különbségük abban rejlik, hogy a játékelemek, hogyan is épülnek be a tanulási élményekbe. Ez a megkülönböztetés pedig nagyobb különbséghez vezet a tanulási eredményekben, amikor a játékalapú tanulást és a gamifikációt hasonlítjuk össze.

A játék alapú tanulás definíciója

Ez az alfejezet a MindResearch Blog „Game based learning vs. gamification” [10] című bejegyzésének elemzése alapján készült, amely egy tudományos könyvrészletet dolgoz fel közérthető formában.

A játék alapú tanulás nem más, mint egyfajta aktív tanulási tapasztalat játék keretében, amely konkrét tanulási célokkal és mérhető eredményekkel rendelkezik.

Ez a tanulási tapasztalat a diákok számára világos és kihívást jelentő célokat ad egy virtuális játék keretein belül, amely magas fokú tanulói interakciót igényel, és informatív visszajelzést ad a tanulók teljesítményéről. Sok esetben a játékok úgy vannak tervezve, hogy a diákok egy valós életszituáció keretein belül tudják megérteni a tananyagot.

A gamifikáció definíciója

Ez a fejezet a MindResearch Blog „Game based learning vs. gamification” [10] című bejegyzésének elemzése alapján készült, amely egy tudományos könyvrészletet dolgoz fel közérthető formában.

Gamifikációnak azt a folyamatot nevezzük, amelynek során a játékelemeket vagy mechanikákat adunk hozzá egy tapasztalathoz, annak érdekében, hogy növeljük az elkötelezettséget vagy az élvezetet.

Ezek a játékelemek általában elkülönülnek a tényleges tanulási tartalomtól. A játékosított leckék vagy tevékenységek tartalmazhatnak olyan elemeket, mint a jelvények, ranglisták, időzített tevékenységek, jutalmak vagy pontok .

Játék alapú tanulás vs Gamifikáció

Ez az alfejezet a MindResearch Blog „Game based learning vs. gamification” [10] című bejegyzésének elemzése alapján készült, amely egy tudományos könyvrészletet dolgoz fel közérthető formában.

A játék alapú tanulás és a gamifikáció közötti legfőbb különbség abban rejlik, hogy a tanulási élmény, vagyis maga a játék hogyan támogatja a diákok belső motivációját, szemben azzal, amikor pusztán játékelemeket illesztünk egy hagyományos

oktatási környezetbe.

Az önmeghatározási elmélet alapján három alapvető pszichológiai szükséglet - a kompetencia, az autonómia és a kapcsolódás – kielégítése tartja fenn és erősíti az intrinzik azaz a belső motivációt. A játék alapú tanulási rendszerekben a játék narratívája és mechanikája van úgy kialakítva, hogy a tanulók lépésről lépésre érezzék képességeik fejlődését(kompetencia), választhassanak stratégiáik és feladataik között(autonómia), valamint együttműködhessenek vagy versenyezhessenek társaikkal(kapcsolódás), így a tanulás élménye önmagában nyújt értelmet és kihívást [10]. Ezzel szemben a gamifikációs megközelítés gyakran külső jutalmakra és visszajelzésekre épít: pontok, rangok, jelvények, vagy időre menő kihívásokat ajánl.

A kognitív értékelési elmélet (Az önmeghatározási elmélet egyik al-elmélete) szerint az ilyen külső jutalmak eleinte növelhetik a motivációt, mert a tanuló kontroll és kompetenciaérzetet kap, ám hosszabb távon az ún „Undermining effect" révén csökkenhet az intrinzik motiváció, ha a diákok elsősorban a jutalomért cselekednek, nem pedig a tanulás élményéért. Ráadásul a gamifikáció sokszor a motiváció legalacsonyabb szintjén, a külső szabályozás szintjén ragad: a tanulók a jutalom elkerülése vagy megszerzése érdekében hajtják végre a feladatokat, anélkül, hogy személyes értelemmel vagy hosszú távú elköteleződéssel viszonyulnának hozzá [10]. Most, hogy definiáltuk a játék alapú tanulás és a gamifikáció közötti motivációs különbségeket, vizsgáljuk meg, hogyan alakultak ki az oktatási játékok az elmúlt évtizedekben:

1.2. A játék alapú tanulás elmélete

Ez az alfejezet a Plass, J.L., Homer, B.D. és Kinzer, C.K. által írt „Foundations of Game Based Learning" [13] című tanulmány elemzése alapján készült.

Napjainkban egyre nagyobb népszerűségnek örvendenek a videójátékok, különösképpen a diákok körében. Bár a videójátékok javítják a szem-kéz koordinációt, mégis fölösleges időpocsékolásnak tűnnek, legfőképp az idősebb generációk számára, mivel nem tartalmaznak olyan „tartalmat", amely valódi tanulást eredményezne. Véleményem szerint ezen probléma abból fakad, hogy a tanulás, az iskoláztatás és a tudás megszerzésének módja kizárólag intellektuális vagy akadémiai területeken gyökerezik –

például a fizikában, történelemben, művészetben vagy éppen az irodalomban .

Ezen gondolatok mélyen berögzültek az emberek gondolkodásában. Az emberek úgy gondolják, hogyha az adott tevékenység szórakoztató, nem jár együtt mély tanulással és csupán csak értelmetlen játékként értelmezhetőek. Természetesen a videójátékok is ebbe a kategóriába sorolandóak.

Ennek a nézőpontnak egy formája régóta létezik a nyugati kultúrában. Példának okával Platón és Arisztotelész álláspontja szerint a tudás, a fentebb említett értelemben önmagában értékes. Ezen ismeretek gyakorlati hasznosítása, melyek nem feltétlenül járnak elmélyült tanulással vagy a tartalom mélyebb megértésével gyakran hétköznapinak és közönségesnek tűnhetnek. Fontos figyelembe venni, hogy a diákok tudása ne maradjon meg passzív tudásként, hanem képesek legyenek azt gyakorlatban is kellőképpen alkalmazni.

Ha a játékalapú tanulást kognitív szemszögből nézzük, akkor a tanulók játékkal való elköteleződésének célja a mentális modellek felépítése. Egy kognitív elmélet szerint a tanulók először kiválasztják a bemutatott információkat, majd azokat a vizuális és verbális reprezentációkká szervezik a munkamemóriában, v égül pedig összekapcsolják ezeket egymással és a már meglévő tudásukkal .

Kognitív megközelítésből a kutatók és a tervezők azt vizsgálják, hogy a játék mely elemei támogatják leginkább a tanulási tartalom kognitív feldolgozását, vagyis azt , hogy hogyan kell az adott tartalmat megjeleníteni, valamint milyen tanulási mechanizmusokat kell kialakítani annak érdekében, hogy a játék elősegítse a kívánt kognitív eredmények elérését. Ezen túl a játék tervezőjének figyelembe kell vennie a tanulók által tapasztalt kognitív terhelést is, tehát azt, hogy milyen mentális erőfeszítés szükséges a játék különböző elemeinek megértéséhez. Az oktatási célú játékok tervezésekor a következőkre kell törekedni:

1. Felesleges kognitív terhelés csökkentése
2. Lényeges információk hatékony feldolgozásának támogatása
3. A tanulók aktív mentális erőfeszítéseinek ösztönzése

A játékok többféle módon segíthetik a kognitív feldolgozást. Az alábbi kulcsterületek játszanak ebben szerepet: 1. A tanulás helyzetbe ágyazottsága

2. Tanulási transzfer
3. Lépcsőzetes támogatás és visszacsatolás

4. Dinamikus értékelés
5. Információs és interakciós tervezés
6. Gesztusok és mozgás szerepe a tanulásban

A tanulási transzfer

Ez az alfejezet a Plass, J.L., Homer, B.D. és Kinzer, C.K. által írt „Foundations of Game Based Learning" [13] című tanulmány elemzése alapján készült.

Napjaink oktatásának egyik legnagyobb kihívása, hogy olyan módon tanítsunk, amely lehetővé teszi a diákok számára, hogy tudásokat ne csak az iskolában, hanem azon kívül is tudják hasznosítani. A transzfer – azaz a megszerzett ismeretek és készségek új helyzetekbe való felhasználása, annál könnyebb, minél inkább hasonlít az új kontextus a tanulás eredeti környezetére. Ugyanakkor számos tényező befolyásolja a tudás átadását / átvihetőségét.

Perkins és Salomon 1989-ben két fő mechanizmust különböztetett meg, amelyeken keresztül a tudás új helyzetekbe átültethető:

• Alacsony szintű transzfer: Ez az automatizmusra épül, és ismétlődő gyakorlás révén teszi lehetővé egy adott készség alkalmazását új szituációkban.

• Magas szintű transzfer: Ez tudatos absztrakciót és a megszerzett tudás alkalmazását igényli eltérő helyezetekben.

A játékok mindkét transzferformát támogatják. Az ismétlődő gyakorlás l ehetőséget ad a készségek megerősítésére és automatizálására(alacsony szintű transzfer), míg a különböző, de egymással összefüggő tapasztalatok elősegítik a tudás általánosítását, így az új helyzetekben is alkalmazhatóvá válik(magas szintű transzfer).

Ezeket figyelembe véve pedig bátran kijelenthetjük, hogy a játékok nemcsak az új készségek elsajátítását segíthetik elő, hanem a meglévő tudás és képességek gyakorlását, elmélyítését és későbbi felhasználását is elősegítik.

Lépcsőzetes támogatás és visszacsatolás

Ez az alfejezet a Plass, J.L., Homer, B.D. és Kinzer, C.K. által jegyzett „Foundations of Game Based Learning" [13] című tanulmány főbb gondolatainak és szakirodalmi megállapításainak feldolgozásán alapul.

Ahogy a videójátékok és a hozzájuk kapcsolódó digitális médiumok egyre összetettebbé és tudatosan oktató jellegűvé válnak, úgy egyre inkább figyelmet fordítanak arra, hogy a játékok természetes módon kialakuló folyamatait, digitális környezetbe ültessék át annak érdekében, hogy a tanulást elősegítsék.

Ennek a fogalmát Wood, Bruner és Ross vezette be 1976-ban, hogy leírják azt a folyamatot, amely során egy szakértő vagy tapasztalt személy egy kevésbé jártas egyént egy probléma megoldásában vagy egy feladat elvégzésében. Lépcsőzetes támogatás során egy szakértő átveszi a tanuló számára még túl nehéz feladatok bizonyos aspektusait, így a tanuló képes elvégezni egy olyan feladatot, amelyet önállóan nem tudna teljesíteni. Noha Wood és munkatársai nem kapcsolták közvetlenül a scaffoldingot Vygotsky proximális fejlődési zónájának elméletéhez, világos, hogy a hatékony támogatás feltétele, hogy a tanuló által megoldandó feladat éppen a fejlődési zónáján belül helyezkedjen e.

A modern pedagógiában a scaffolding fogalma annyira elterjedtté vált, hogy veszélybe került a pontos jelentése. A scaffolding két kulcseleme:

1. Folyamatosan alkalmazkodó támogatás, amely a tanuló fejlődésének folyamatos értékelésére épül.

2. Fokozatos visszavonás, amely során a támogatás csökken, ahogy a tanuló egyre önállóbbá válik. Ennek ellenére sok oktatási technológiában alkalmazott scaffolding valójában nem ad megfelelő támogatást, mivel az adott segédletek nem vonhatók vissza, így inkább a tudás megosztott feldolgozásához, nem pedig az önálló tanulás fejlődéséhez vezetnek.

A modern szórakoztató videójátékok kiemelkedően hatékonyak a játékosok támogatásában. Gyakran egy oktató(tutorial)szinttel kezdődnek, ahol a játék figyeli a játékos tevékenységeit, és szükség esetén visszajelzést és támogatást ad azokon a területeken, ahol a játékos elakad. Ha a játékos sikeresen teljesíti az oktató szintet, a támogatás fokozatosan visszavonásra kerül, így a játékos önállóan folytathatja a játékot. Viszont ezen módszer az oktatási célú játékokban sokkal nehezebben alkalmazható,mivel ezekben sokkal bonyolultabb a tanuló fejlődésének dinamikus értékelése és megfelelő támogatása.

A hatékony lépcsőzetes támogatás (scaffolding)

Ez az alfejezet a Plass, J.L., Homer, B.D. és Kinzer, C.K. által írt „Foundations of Game Based Learning" [13] című tanulmány elemzése alapján készült.

Ez alapvető feltétele a tanulók tudásának és készségeinek folyamatos és pontos értékelése. Az értékelésnek pontosan meg kell határoznia, hogy mely támogatási elemek lesznek a leghatékonyabbak, és dinamikusan kell alkalmazkodnia ahhoz, hogy mikor kell módosítani vagy csökkenteni ezeket a támogatásokat.

A különböző adaptív oktatási rendszerek szintén dinamikus értékelést igényelnek. Például, ha egy játék tanulási folyamata alkalmazkodik a tanuló aktuális tudásszintjéhez, akkor a feladatok sikerességének folyamatos mérése határozza meg, hogy a következő lépés milyen nehézségű feladat lesz, vagy hogy a tanuló továbbhaladhat-e egy új témakörre.

A dinamikus értékelés első lépése, hogy egyértelműen meghatározzuk azokat a tényezőket, amelyeket értékelni kívánunk. Ez függ a konkrét tanulási céloktól, valamint azoktól az egyéni jellemzőktől, amelyek befolyásolhatják a tanulás kimenetelét. Az Evidence-Centered Design (Mislevy & Heartel, 2006) egy olyan keretrendszert kínál, amely segíthet a játékokon belüli értékelések kialakításában (erről bővebben: Plass, Homer et al., 2013).

A kulcsfontosságú információk kétféle adatforrásból szerezhetők meg:

1. Folyamatadatokból – vagyis abból, hogy a tanuló milyen tevékenységeket végez a játék során.

2. Eredményadatokból – vagyis abból, hogy a tanuló milyen produktumokat hoz létre a játékban .

Az oktatási célú játékok gyakran tudatosan úgy vannak kialakítva, hogy a játékosok olyan specifikus tevékenységeket végezzenek, amelyek információt nyújtanak a tanuló tudásszintjéről és készségeiről. Plass, Homer és munkatársai (2013) ezt a folyamatot a játékokban alkalmazott értékelési mechanizmusokkal kapcsolatosan tárgyalták.

A pontosan megtervezett játékbeli értékelések nemcsak azt biztosítják, hogy a játék hatékonyan alkalmazkodjon a tanulók igényeihez, hanem akár szükségtelenné is tehetik a külső, hagyományos tanulási eredményértékeléseket [13].

A Játékalapú tanulás motivációs alapjai

Ez az alfejezet a Plass, J.L., Homer, B.D. és Kinzer, C.K. által írt „Foundations of Game Based Learning" [13] című tanulmány elemzése alapján készült.

A játékalapú tanulás motivációs szemszögből nézve elsősorban arra épít, hogy a játékok képesek lekötni és motiválni a játékosokat azáltal, hogy élvezetes és folytatásra ösztönző élményeket nyújtanak (Gee, 2003; Ryan, Rigby Przbylski, 2006; Zusho, Anthony, Hashimoto Robertson, 2014). Az oktatási játékok esetében feltételezzük, hogy a játékosok interakciója a játékkal motiválja őket, és elősegíti a játék tartalmának kognitív feldolgozását, ezáltal javítva a tanulási folyamatot (Delacruz, 2012). Ugyanakkor egyes kutatók szerint az élvezeti célú játékok magas szintű elköteleződése nem feltétlenül vihető át az oktatási környezetbe (Hoffman Nadelson, 2010).

Mégis, számos kutatás próbálta meghatározni azokat az elemeket, amelyek hozzájárulnak a játékokban megélt motivációhoz és elköteleződéshez. Ezek közé tartoznak például az ösztönző rendszerek, vizuális esztétika, játékmenet-mechanizmusok, narratíva, fantázia és zenei aláfestés (Gee, 2003; Loftus Loftus, 1983; Malone, 1981; Squire, 2011). Az oktatási játékok fejlesztésénél fontos megvizsgálni, hogy ezek az eszközök hogyan alkalmazhatók a tanulás támogatására.

Motivációs elméletek és a tanulás kapcsolata [13]

Ez az alfejezet a Plass, J.L., Homer, B.D. és Kinzer, C.K. által írt „Foundations of Game Based Learning" [13] című tanulmány elemzése alapján készült.

Bár az oktatásban a motiváció kutatása hosszú múltra tekint vissza, a motivációs elméleteket eddig csak korlátozottan alkalmazták az oktatási játékok értelmezésében. A korai megközelítések behaviorista alapokra épültek, amelyek a tanulók belső késztetéseire és szükségleteire helyezték a hangsúlyt (Graham Weiner, 1996). Hasonlóan, a videojátékok motivációjának korai magyarázatai is megerősítési mechanizmusokat használtak a játékokban tapasztalt elköteleződés magyarázatára (Loftus Loftus, 1983) .

A modern motivációs elméletek azonban szélesebb perspektívát alkalmaznak. Eccles, Wigfield és Schiefele (1998) három kulcskérdés köré szervezte az akadémiai

motiváció megértését:

1. Képes vagyok rá?
2. Akarom ezt csinálni, és ha igen, miért?
3. Mit kell tennem a siker érdekében?

A jelenlegi motivációs elméletek – mint például az elvárás-érték elmélet (Wigfield Eccles, 2000b), a önmeghatározás-elmélet (Ryan Deci, 2000b), az énhatékonyság-elmélet (Schunk, 1991), az attribúciós elmélet (Weiner, 2012), az eredménycél-orientációs elmélet (Ames Archer, 1988; Dweck Leggett, 1988; Elliot, 2005), valamint az érdeklődéselmélet (Schiefele, 1991) – eltérő hangsúlyokat helyeznek a fenti kérdések egyes aspektusaira, különböző tényezők motivációra gyakorolt hatását vizsgálva.

Hogyan támogatják a videojátékok a motivációt? .

A videojátékok kiválóan alkalmasak arra, hogy válaszokat adjanak az első és a harmadik motivációs kérdésre:

1. Képes vagyok rá? – A játékok úgy vannak tervezve, hogy a játékosok elérjék céljaikat, fokozatosan fejlesztve képességeiket .

2. „Mit kell tennem a siker érdekében?" – A játékok világosan meghatározzák a célokat és a szükséges lépéseket a sikerhez .

Ezt többféleképpen érik el:

1. Rugalmas kudarcmechanizmusokkal – Ha egy játékos elbukik, a játék lehetőséget biztosít a tanulásra, és újrapróbálkozásra ösztönöz .

2. Tréningmódokkal és oktató szintekkel – A játékok bemutatják a mechanikákat, és lehetőséget adnak a gyakorlásra .

3. Adaptív játékmenettel – Ha egy játékos nehézségekbe ütközik, a játék automatikusan csökkentheti a nehézséget vagy további támogatást nyújthat [13].

4. Online közösségek támogatásával – A játékosok egymástól is tanulhatnak és segítséget kérhetnek .

A második kérdés, azaz „Akarom ezt csinálni, és ha igen, miért?", azonban összetettebb kihívás. Ennek megválaszolásához az elméletek a belső motiváció, az egyéni érdeklődés és az elérni kívánt célok vizsgálatára helyezik a hangsúlyt (Wig-

field, Eccles, Schiefele, Roeser , Davis-Kean, 2006; Zusho et al., 2014) .

A következő lépés annak feltárása, hogy az oktatási játékok hogyan alkalmazhatják ezeket a motivációs elméleteket a hatékonyabb tanulási élmény érdekében [13].

A játékalapú tanulás motivációs alapjai

Ez az alfejezet a Plass, J.L., Homer, B.D. és Kinzer, C.K. által jegyzett „Foundations of Game Based Learning" [13] című tanulmány főbb gondolatainak és szakirodalmi megállapításainak feldolgozásán alapul.

Amikor a játékalapú tanulást motivációs szempontból vizsgáljuk, akkor a játékok azon képességére helyezzük a hangsúlyt, hogy képesek lekötni és motiválni a játékosokat azáltal, hogy élményeket nyújtanak, amelyeket élveznek és folytatni akarnak (Gee, 2003; Ryan, Rigby és Przbylski, 2006; Zusho, Anthony, Hashimoto és Robertson, 2014). Feltételezhető, hogy egy oktatási játék játszása során a játékosok interakcióba lépnek a játékkal, ezáltal motiválódnak, és elősegítik a játék tartalmának kognitív feldolgozását, ami javítja a tanulást (Delacruz, 2012), bár egyes kutatók szerint az élvezeti célú játékokkal kapcsolatos magas szintű elköteleződés nem biztos, hogy átültethető az oktatási kontextusba (Hoffman és Nadelson, 2010) .

Ennek ellenére számos erőfeszítés történt annak érdekében, hogy azonosítsák azokat a konkrét elemeket, amelyek hozzájárulnak a játékokban való elköteleződéshez és motivációhoz, például az ösztönző rendszereket, a vizuális esztétikát, a játékmenetet, a narratívát/fantáziát és a zenei aláfestést (pl. Gee, 2003; Loftus és Loftus, 1983; Malone, 1981; Squire, 2011), és ezeket figyelembe vegyék az oktatási játékok tervezésekor. Ugyanakkor, annak ellenére, hogy nagy az érdeklődés ezen a területen, kevés szisztematikus kísérlet történt a motivációs elméletek alkalmazására a játékokon keresztüli tanulás megértése érdekében, noha az oktatásban a motiváció elméleti és empirikus alapja kiterjedt.

A motiváció szerepének kezdeti magyarázatai a behaviorista hagyományból eredtek, és a tanulók ösztöneire, szükségleteire és viselkedésére helyezték a hangsúlyt (Graham és Weiner, 1996). Hasonlóképpen, a videojátékok motivációs magyarázatainak korai kísérletei is behaviorista konstrukciókat alkalmaztak, például a megerősítési

mechanizmusokat, hogy megmagyarázzák a motivációt és az elköteleződést (pl. Loftus és Loftus, 1983). Az újabb elméletek szélesebb perspektívából közelítik meg, hogy mi motiválja a diákokat. Eccles, Wigfield és Schiefele (1998) szerint a kortárs teljesítménymotivációs elméletek három alapvető kérdés köré építhetők, amelyeket a diákok egy tanulási feladattal szembesülve tesznek fel maguknak: „Képes vagyok erre?" „Akarom ezt csinálni, és miért?" „Mit kell tennem a siker érdekében?" .

A jelenlegi motivációs elméletek, például az elvárás-érték elmélet (Wigfield és Eccles, 2000b), az önmeghatározás elmélete (Ryan és Deci, 2000b), az én-hatékonyság elmélete (Schunk, 1991), az attribúciós elmélet (Weiner, 2012), a teljesítménycél-orientáció elmélete (Ames és Archer, 1988; Dweck és Leggett, 1988; Elliot, 2005), valamint az érdeklődés elmélete (Schiefele, 1991) ezekre a kérdésekre összpontosítanak, és különböző szempontokat emelnek ki arról, hogyan alakul a motiváció.

A videojátékok sok szempontból jól alkalmazhatók ezen motivációs kérdések megválaszolására (Zusho et al., 2014). A játékokat úgy tervezték, hogy biztosítsák a játékosok sikerélményét, ezáltal megerősítve az első kérdésre adott pozitív választ: „Képes vagyok erre?", valamint hogy egyértelmű iránymutatást adjanak a harmadik kérdésre: „Mit kell tennem a siker érdekében?". Ezt többféleképpen érik el, például a „kegyes kudarc" mechanizmusával, amely lehetővé teszi a játékosok számára, hogy a hibáikból tanuljanak és újra próbálkozzanak. Emellett sok játék rendelkezik oktató módokkal vagy bevezető szintekkel, amelyek bemutatják a játék funkcióit és lehetőséget biztosítanak a gyakorlásra. Továbbá a legtöbb játék adaptív: ha egy játékos nehézségekbe ütközik, a játék csökkentheti a nehézségi szintet vagy további támogatást nyújthat. Végül sok játék rendelkezik online közösségekkel, ahol a játékosok segítséget és támogatást kaphatnak.

A második motivációs kérdés— „Akarom ezt csinálni, és miért?" megválaszolása azonban összetettebb. Az elméleti megközelítések arra összpontosítanak, hogy a tanulók miért akarnak valamit megtanulni, figyelembe véve az intrinszik motivációjukat, értékeiket, érdeklődésüket és teljesítménnyel kapcsolatos céljaikat (Wigfield, Eccles, Schiefele, Roeser és Davis-Kean, 2006; Zusho et al., 2014) .

1.3. Elméleti modellek és keretrendszerek:

Ez az alfejezet a Hunicke, R., LeBlanc, M. és Zubek, R. által megalkotott MDA-modellre és a „MDA: A formal approach to game design and game research” [5] című tanulmányban bemutatott elméleti megközelítésre épül.

Egy kutatás vagy játékfejlesztés megalapozásához szükségszerű, hogy az elméleti háttér logikusan és átláthatóan legyen felépítve. Ezt pedig gyakran elméleti modellek és keretrendszerek segítségével valósítjuk meg. Az elméleti modellek rendszerezik a témához kapcsolódó fogalmakat és összefüggéseket, és meghatározzák a kutatásunk vagy játékfejlesztésünk főbb céljait is. Ezek a keretrendszerek fontos szerepet töltenek be a játékfejlesztésben, mivel segítenek megérteni és értelmezni a játékfejlesztéshez kapcsolódó játékelméleti alapokat. A szakdolgozatomban egy olyan konkrét keretrendszert fog bemutatásra kerülni, amely meghatározó szerepet játszik a játéktervezés és a játékélmény elemzésének több szempontjából is. Ez a keretrendszer nem más, mint az MDA, amely a Mechanika-Dinamika-Esztétika angol szavak rövidítése .

Az MDA egy formális megközelítés a játékok megértésére, amely megpróbálja áthidalni a játéktervezés és fejlesztés, valamint a játékkritika és a technikai kutatások közötti szakadékokat. A különbség a játékok és az egyéb szórakoztató termékek között, mint például a könyvek, filmek és színdarabok, hogy a fogyasztásuk viszonylag kiszámíthatatlan. A játék során bekövetkező események és azok végkimenetele a játék elindításánál még nem ismertek. Az MDA keretrendszer a játékokat fogyasztását úgy formalizálja, hogy azokat külön összetevőkre bontja. Ez a 3 összetevő jelen esetben:

- Szabályok
- Rendszer
- Szórakozás és a tervezési megfelelőik létrehozása, amely a keretrendszer nevéből adódóan a mechanika, dinamika és az esztétika .

A mechanika a játék egyes összetevőit írja le, az adatreprezentáció és az algoritmus szintjén .

A dinamika azt írja le, hogyan viselkedik a játék menet közben, hogyan reagálnak a játékszabályok és a rendszerek a játékos döntéseire és a játék különböző elemeinek egymásra gyaykorolt hatásaira az idő folyamán .

Az esztétika azokat a kívánatos érzelmi reakciókat írja le, amelyeket a játékosban kivált, amikor az adott játékkal interakcióba lép . Az MDA keretrendszer egyik alapvető gondolata, hogy a játékok inkább hasonlítanak egy tervezett műtárgyhoz vagy mesterséges objektumhoz, mintsem egy hagyományos médiumhoz. Ez azt jelenti, hogy a játék tartalma nem az , amit megjelenít vagy sugároz a játékos felé, hanem az a viselkedés és működés, amely a játék során létrejön. Ha a játékokra úgy tekintünk, mint interakció által viselkedést létrehozó rendszerekre, az segít abban, hogy világosabban átlássuk a tervezési döntéseket, és pontosabb elemzést végezhessünk mind a fejlesztés, mind a kutatás különböző szintjein . Az MDA-ról részletesen:

A játékok tervezésekor érdemes egyszerre szem előtt tartani a tervező és a játékos nézőpontját is. Így könnyebben észrevehetjük, hogy egy látszólag apró változtatás az egyik rétegben miként „hullámzik át" a többi szinten. Emellett a játékos szempontjának tudatos figyelembevétele, elősegíti az élményközpontú tervezést. Ezért az MDA-elemzést az Esztétikával kezdjük, majd a Dinamikán át jutunk el az alapul szolgáló Mechanikához [5].

Esztétikai modellek:

Az esztétika, az, amit játék közben érzünk vagy átélünk. Ez a keretrendszerünk azon része, amellyel meghatározhatjuk és leírhatjuk a játékélményt. Például a versengés akkor jön létre, ha a játékosok érzelmileg elköteleződnek egymás legyőzésében. Ehhez elengedhetetlen a játékokban egy pontrendszer és egy összesítő rangsor, annak érdekében, hogy a diákok győzelme egyértelműen látható legyen. Ezen funkciók hiánya pedig csökkentheti a játékos, jelen esetben a diákok motivációját és érdeklődését a játék iránt .

Dinamikai modellek:

A dinamikák a játék működése közben létrejövő viselkedési minták, amelyek életre keltik az esztétikai élményeket:

• Kihívás: például, ha a játék időhöz van kötve.

• Közösség: ez akkor alakul ki, ha a diákoknak lehetőségük van információt megosztaniuk egymással, es ezáltal csapatban kezdenek el dolgozni.

• Dramatikus feszültség: ez akkor jön létre, ha a dinamikák finoman építik fel a feszültséget, majd felszakítják és egy levezető szakaszban feloldják azt.

Mechanikai modellek:

Mechanikák alatt azokat a különféle cselekvéseket, viselkedéseket és vezérlőelemeket értjük, amelyeket a játékos egy adott játékkörnyezetben használhat. Ezek együtt a játék tartalmával formálják a játék menetét és a belőle kialakuló dinamikát . Például az egy matematikai kvízjáték mechanikái az alábbiak lehetnek:

• Kérdésgenerálás: véletlenszerűen vagy előre összeállított témakörök alapján jelennek meg a feladatok .

• Válaszválasztás: a játékosnak több lehetőség közül kell kiválasztania a helyes megoldást (A/B/C/D) .

• Időkorlát: minden kérdésre adott időkorlát, amely időnyomás alá helyezi a játékost .

• Pontszámítás: a helyes válaszok után jóváírt pontok, esetleg bónuszpontok a gyors válaszokért .

E mechanikákból olyan dinamikák alakulhatnak ki, mint az időnyomás okozta feszültség, vagy a pontverseny élménye .

1.4. Empirikus eredmények és meta analzisek [17]

Ez az alfejezet a Wouters, P., van Nimwegen, C., van Oostendorp, H. és van der Spek, E. által készített „A meta analysis of the cognitive and motivational effects of serious games" [17] című átfogó tanulmány eredményeinek feldolgozásán alapul.

A"komoly" játékok definíciója: Számos kutató adott már meg definíciókat vagy osztályozásokat a számítógépes játékok jellemzőire vonatkozóan (Garris, Ahlers Driskell, 2002; Malone, 1981; Prensky, 2001). A jelen meta-analízis céljaira a számítógépes játékokat az alábbi tulajdonságok mentén írjuk le:

• Interaktivitás: a játékos folyamatosan beavatkozik a játék menetébe, és annak kimenetele a döntéseitől függ (Prensky, 2001; Vogel et al., 2006) .

• Megállapodott szabályok és korlátok: a játékmenetet előre definiált szabályok és mechanizmusok keretezik (Garris et al., 2002).

• Világos célkitűzés és kihívás: minden játék egy egyértelmű cél felé irányul, amely gyakran valamilyen kihívás formájában jelenik meg (Malone, 1981).

• Folyamatos visszajelzés: a játékos pontszám vagy a játékvilág változásai révén

követheti előrehaladását a cél elérésében (Prensky, 2001).

Egyes szerzők a versengést (más játékos, a számítógép vagy önmaga legyőzése) is a definíció részének tekintik, ám kérdéses, hogy ez feltétlenül minden számítógépes játék lényegi jellemzője-e. Vannak olyan játékok (például SimCity), ahol a játékosok inkább egy virágzó város megalkotását élvezik, anélkül hogy versengésre törekednének. Hasonlóképp, a narratíva vagy történetfejlődés fontos lehet az adventure-játékokban, de nem elvárás minden játék esetében (például az akciójátékoknál általában nincs hangsúlyos történet) .

Komoly (serious) játék alatt azt értjük, hogy a játék elsődleges célja nem pusztán a szórakoztatás, bár ez nyilván előny, hanem az, hogy a szórakoztató jellegét felhasználva képzési, oktatási, egészségügyi, közpolitikai vagy stratégiai kommunikációs célokat szolgáljon (Zyda, 2005) .

Elméleti keretrendszer:

Elméletileg a játékok két módon hatnak a tanulásra: egyrészt megváltoztatják a kognitív folyamatokat, másrészt befolyásolják a motivációt. A számítógépes játékok interaktív jellege összhangban áll az oktatáspszichológia jelenlegi hangsúlyával, miszerint az anyag aktív, kognitív feldolgozása előfeltétele a hatékony és tartós tanulásnak (Wouters, Paas van Merriënboer, 2008). Másodszor, a számítógépes játékok lehetővé teszik olyan feladatok szimulálását, ahol a játékban végzett tevékenység ugyanezeket a kognitív folyamatokat igényli, mint a való világban (Tobias, Fletcher, Dai Wind, 2011). Harmadszor, a játékok azonnali visszajelzést adnak a cselekvések helyességéről, így a játékosok korrigálhatják az esetleges hibákat (Cameron Dwyer, 2005; Moreno Mayer, 2005).

Számos besorolás létezik a tanulási eredményekről. Ebben a meta-analízisban a tanulás kognítiv dimenzióját helyezzük előtérbe. A Wouters et al.(2009) szerinti felosztásban ez a dimenzió tudásra és kognitív készségekre oszlik .

• A tudás a kódolt ismeretkre utal, ideértve a szövegalapú és az ábrákkal származó tudást is.

• A kognitív készség összetettebb gondolkodási folyamatokat rejt magában, például probléma-megoldást, amikor a tanuló alkalmazza az ismereteket és a szabályokat egy új helyzet megoldásához. A fentiek alapján a tanulásnál megkülönböztetjük a tudás és a készségfejlődését, amelyre három hipotézis került felállításra:

H1: A serious game-ekkel történő oktatás nagyobb tanulási eredményeket hoz, mint a hagyományos oktatási módszerek.

Például a szimulációs játékok esetében vannak bizonyítékok arra, hogy a megszerezett tudás és a készségek idővel is megmaradnak (Pierfy, 1977; Sitzmann, 2011; van der Spek, 2011). Ebből adódik a második hipotézis:

H2: A serious game-ekkel történő oktatás hosszabb távon is magasabb szintű tudásmegtartást eredményez, mint a hagyományos módszerek .

Számos elmélet hangsúlyozza a serious game-ek belső (intrinzik) motivációt fokozó hatását (Garris et al., 2002; Malone, 1981), vagyis azt, hogy a játékosok nem külső jutalmakért, hanem magáért a játékélményért fektetnek be időt és energiát. Malone (1981) szerint a legfontosabb intrinszik motiváló tényezők a kihívás, a kíváncsiság és a fantázia. Emellett az önmeghatározás-elméletből ismert autonómia (a döntési szabadság lehetősége) és kompetencia (a feladat legyen kihívást jelentő, de ne túl nehéz) szintén pozitívan befolyásolja a motivációt (Przybylski, Rigby Ryan, 2010; Ryan, Rigby Przybylski, 2006). Ebből pedig adódik a harmadik hipotézis:

H3: A serious game-ekkel történő oktatás motiválóbb, mint a hagyományos oktatási módszerek .

1.5. Moderator változók a tanulásban

Ez az alfejezet a Wouters, P., van Nimwegen, C., van Oostendorp, H. és van der Spek, E. által készített „A meta analysis of the cognitive and motivational effects of serious games" [17] című átfogó tanulmány eredményeinek feldolgozásán alapul.

A modern oktatáselmélet szerint a tanulás akkor lehet igazán hatékony, ha a tanulók aktívan dolgozzák fel az anyagot, ez a tanulás egyik elengedhetetlen feltétele. (Chi, de Leeuw, Chiu és LaVancher, 1994; Mayer, 2001; Wittrock, 1974). A kutatások többsége azt mutatja, hogyha a tanulók aktívan bevonódnak a tanulási folyamatba, akkor jobban tudják az újonnan szerzett tudásukat a meglévővel összekapcsolni,ezáltal pedig jobban tudják alkalmazni más szituációkban is. (Chi et al., 1994; Renkl és Atkinson, 2002; Roy és Chi, 2005).

Ezen adatokból kifolyólag, egy olyan tanulási környezet, amely aktív gondolkodásra és tevékenységre ösztönzi a tanulókat, például feladatmegoldással vagy gya-

korlással, hatékonyabb lehet, mint egy olyan környezet, ahol nem ösztönzik őket aktív részvételre, például csak szöveget olvasnak vagy előadást hallgatnak .

A számítógépes játékoknál a játékosok általában cselekszenek, majd látják ezeknek a cselekedeteknek az eredményeit, ami a játékvilág vagy környezet változásaiban jelenik meg. Ez egyfajta intuitív tanuláshoz vezethet: a játékos tudja, hogyan kell alkalmazni a tudását, de nem feltétlenül képes azt tudatosan, szavakkal megfogalmazni (Leemkuil de Jong, 2011) [17].

Ugyanakkor fontos, hogy a tanulók szavakkal is képesek legyenek megfogalmazni a megszerzett tudást, mivel ez segíti az új ismeretek összekapcsolódását a meglévőkkel, ez pedig jobb emlékezést és nagyobb mértékű tudás transzfert eredményez .

Lehetséges, hogy a kiegészítő oktatási módszerek, mint például a megbeszélés, célzott gyakorlás) lehetővé teszik a tanulók számára, hogy olyan tanulási tevékenységekben vegyenek részt, amelyek elősegítik a tudás tudatos megfogalmazását és mélyebb megértését .

Ezt alátámasztja Sitzmann(2011) kutatása is, aki azt találta, hogy azok a tanulási elrendezések, amelyekben a szimulációs játékot más oktatási módszerrel egészítették ki, nagyobb tanulási eredményeket hoztak .

Az egyik érv a számítógépes játékokon alapuló kollaboratív tanulás mellett az, hogy ez segíti a tanulókat abban, hogy kimondják és megfogalmazzák azokat az ismereteket, amelyek egyébként csak intuitív szinten maradnának (van der Meij, Albers , Leemkuil, 2011) .

Ugyanakkor a kollaboratív és egyéni játékmenet összehasonlításával kapcsolatos kutatások ellentmondásos eredményeket hoztak. Inkpen, Booth, Klawe és Upitis (1995) azt figyelték meg, hogy a közös játék jelentősen jobb motivációs és tanulási eredményeket hozott, mint az egyéni játék – ezt azonban van der Meij és munkatársai (2011) nem tudták megerősíteni .

Vogel és munkatársai (2006) metaanalízise szerint mind az egyéni, mind a csoportos játékosok nagyobb kognitív fejlődést mutattak az interaktív szimulációkban, mint a hagyományos oktatási módszerek esetében – ám az egyéni játékosok esetében sokkal nagyobb volt a fejlődés mértéke.

1.6. Játékok taxiómái

Ez az alfejezet az Anderson, L.W. és Krathwohl, D.R. által szerkesztett „A taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives" [2] című szakirodalmi mű elemzésére épül.

A taxonómia elsődleges célja, hogy segítsen a tanároknak és oktatáskutatóknak egyértelműen meghatározni és osztályozni a tanulási célokat, az „egyszerű emlékezéstől" a „kreatív újrateremtésig" terjedő skálán. Bloom eredeti modelljének felülvizsgálatával az új változat hangsúlyosan igyekszik meghatározni a kognitív folyamatokat és a tudástípusokat így a célkitűzések megfogalmazása és a hozzájuk kapcsolódó tevékenységek tervezése sokkal áttekinthetőbbé válik .

Signifikanciája: a két dimenzió (kognitív folyamatok és tudástípus) között:

A legfontosabb újítás, hogy a taxonómia többé nem pusztán egy hierarchikus lista, hanem két dimenzió mentén szervezi a tanulási célokat: egyrészt a hat kognitív művelet (emlékezés, megértés, alkalmazás, elemzés, értékelés, alkotás), másrészt a négy tudástípus (faktuális, koncepcionális, eljárási és metakognitív) metszetében. Ennek köszönhetően: könnyebben átlátható, milyen típusú tudással és milyen gondolkodási művelettel dolgoztatjuk a diákokat, és lehetőségünk adódik arra, hogy a tananyagot és a jegyek általi értékelést jobban összehangoljuk. A metakognitív tudás beemelése reflektív, önszabályozó tanulási célokat is támogat. Gyakorlati előnyöknél három fő pontot érdemes megemlíteni, ezek:

• A feladatigék listája, amely segíti a tesztkérdések és feladatleírások tervezését,

• Tanári és tanulói önértékelési eszközök (például rubrikák) építhetők rá,

• A folyamatok dokumentálása és nyomon követésének egyszerűsítése, ami támogatja a pedagógiai kutatásokat és azok fejlesztését is.

Mint ahogy azt már fentebb is említettük, a kognitív folyamatdimenziónak hat fő szintjét különböztetünk meg, ennek az első szintje:

1.Emlékezés, amelynek célja, hogy a tanulók a már korábban megszerzett tudásukat megfelelően fel tudják idézni a hosszú távú memóriájukból, változatlan, vagy csak minimálisan módosított formában, Ez a legelső és legalapvetőbb szintje a Bloom taxonómiában. Ezen képesség nélkülözhetetlen elemét képezi a magasabb szintű gondolkodáshoz és a problémamegoldáshoz. Az emlékezésnek két főbb típusát különböztetjük meg:

• Felismerés. A tanuló összeveti a jelenlegi információit a memóriájában tárolt ismeretekkel és eldönti, hogy van-e egyezés .

• Felidézés. A tanuló önállóan hív elő információt egy kérdés vagy más inger alapján. Az értékelési feladatokban az emlékezés mérése történhet:

• Kevés mennyiségű támpontokkal, ami annyit jelent, hogy a tanuló kevés információ alapján idéz fel választ .

• Nagy mennyiségű támponttal: itt már segítő kulcsok is megjelennek a kérdésekben, például mondatkiegészítés .

2.A második szint, nem más, mint a megértés. Akkor mondhatjuk, hogy a tanuló megértett valamit, ha képes értelmet adni annak, amit hallott, látott vagy épp olvasott. A megértés során a diák összekapcsolja az új ismereteket a már meglévő tudásával. Ennek értelmében pedig új információkat épít be a fejében már kialakult információ rendszerek összeségébe. Mivel ezek a rendszerek alapvetően fogalmakból tevődnek össze, a fogalmi tudás nagyon fontos a megértéshez. A megértéshez kapcsolódó gondolkodási műveletek közé tartozik például az értelmezés, példák hozása, kategorizálás, összefoglalás, következtetés levonása, összehasonlítás és magyarázat .

3.A harmadik szintnek a tudás alkalmazását nevezzük. Az alkalmazás azt jelenti, hogy a tanulók eljárásokat használnak feladatok megoldására vagy problémák megoldására .

Feladatnak vagy gyakorlatnak azt nevezzük, amikor a tanuló már ismeri a megoldás menetét, ekkor a megoldás rutinszerűen zajlik. Problémának pedig azt nevezzük, amikor a tanulónak először rá kell jönnie, milyen módon tudja megoldani azt, tehát nem biztos abban, hogy hogyan is kezdjen hozzá. az alkalmazás kategóriájában kétféle gondolkodási műveletet különböztetünk: az első a végrehajtás, amikor ismerős feladatokról van szó. A második pedig az alkalmazás / megvalósítás amikor új problémát kell megoldani .

4.A negyedik szint, az elemzés, ami azt jelenti, hogy a tanulók képesek egy anyagot részeire bontani és megérteni, hogyan kapcsolódnak ezek a részek, illetve az egész szerkezethez. Ez a gondolkodási művelet három fő részből áll:

• Megkülönböztetés: felismerni, mely részek fontosak vagy lényegesek egy feladatban

• Rendszerezés: megérteni, hogyan épülnek fel ezek a részek, milyen a szerkezetük

• Cél felismerése: felfedezni a feladat szövege mögött rejtőző szándékot vagy célt

.

Az elemzés elsajátítása legtöbb esetben a megértés kibővítéseként vagy az értékelés előkészítéseként jelenik meg az oktatásban. Sok tantárgyban, köztük a matematikában is fontos célt képez a tanulók elemzőképességének fejlesztése. A tanárok legfőbb célja pedig elérni, hogy a diákok ezen témában képesek legyenek megkülönböztetni:

• a tényt a véleménytől [2].

• felismerni, hogyan támasszák alá az állításokat [2].

• szétválasztani a lényeges és lényegtelen információkat [2].

• megérteni, hogyan kapcsolódnak egymáshoz a feladatok egyes elemei [2].

• felismerni a kimondatlan feltételezéseket [2].

• azonosítani a főbb és mellékes gondolatokat [2].

Fontos megjegyezni, hogy a megértés, az elemzés és az értékelés gyakran összefonódnak, és egy gondolkodási folyamat során többször is visszatérhetünk hozzájuk. Ugyanakkor ezek különálló gondolkodási kategóriák, mert például valaki értheti ugyan az anyagot, de mégsem tudja jól elemezni, vagy valaki képes részletes elemzésre, de nem tud jó értékelést adni .

5.Az ötödik szintünk az értékelés, amely azt jelenti, hogy valaki megítél egy folyamatot, feladatot vagy módszert bizonyos előre meghatározott szempontok és mércék alapján. Ezek a szempontok lehetnek például a minőség, a hatékonyság, az eredményesség vagy az állandóság. A mércéknek két típusát különböztetjük meg, az első a mennyiségi mérce(például: elég nagy ez az adatmennyiség?), a második pedig a minőségi mérce (elég jó ez a megoldás?). Az értékelés során ezek a mércék kerülnek alkalmazásra a választott szempontokra, például: „Elég hatékony ez az eljárás?”, vagy „Megfelelő minőségű ez a termék?” .

Ez a gondolkodási kategória két fő mentális műveletet foglal magában:

• ellenőrzés (checking): amikor azt vizsgáljuk, mennyire belsőleg következetes valami (pl. egy szöveg vagy módszer),

• kritika (critiquing): amikor külső szempontok alapján értékelünk (pl. egy adott szabvány vagy cél alapján).

Fontos megemlíteni, hogy nem minden megítélés számít valódi értékelésnek. Például amikor egy diák eldönti, hogy egy példa illik-e egy kategóriába, vagy hogy

egy adott módszer megfelelő-e egy feladat megoldásához, ezek is ítéletek, de nem tekinthetők önálló értékelésnek. A legtöbb gondolkodási folyamat valamilyen szintű megítélést igényel, de az értékelés kategória sajátossága, hogy minden meghatározott mércék szerint történik .

6.A hatodik és egyben utolsó szint az alkotás, amelynek lényege, hogy a tanuló különböző elemeket úgy kapcsol össze, hogy azokból új, értelmes dolgot hozzon létre. Ez gyakorlatban annyit jelent, hogy a tanulók a már meglévő, tanult elemeiket rendszerezik és rendezik újra vagy alakítják át. Az alkotás tehát, mindig a korábbi tanulási tapasztalatokra épül, és ezek tudatos felhasználásával jön létre valami új.

Bár sok alkotás típusú célkitűzés hangsúlyozza az eredetiséget, nem mindig ez a fő szempont. Fontosabb, hogy a diákok képesek legyenek különféle elemeket szintetizálni, vagyis új egésszé alakítani, ez egy dolgozatnál például egy értelmesen összeállított struktúrában nyílvánulhat meg, amely az eddig tanult információkat kapcsolja össze .

Az alkotás különböző a többi gondolkodási folyamattól (mint a megértés, alkalmazás stb), mivel itt nem egy adott rendszer részeit kell értelmeznünk, hanem egy új rendszert kell létrehoznunk. Az alkotás pedig 3 fő fázisból áll:

• Problémaértelmezés: a tanuló megérti a feladatot és különböző megoldási lehetőségeken gondolkodik .

• Megoldástervezés: a tanuló kiválasztja az általa megjobbnak vagy leglogikusabbnak tűnő megoldást és kidolgoz egy konkrét megoldás tervet rá [2].

• Megoldáskivitelezés: végrehajtja az általa kidolgozott tervet, létrehozva egy végső választ .

Ez a kategória leggyakrabban az írásbeli vagy mélyebb megértést igénylő feladatoknál kerül megmutatkozásra, ahol a diákoknak nem csak az eddig tanultakra kell emlékezniük, hanem valami újat is kell alkotniuk belőle [2].

1.7. Pedagógiai előnyök és kihívások

Ez az alfejezet a MindResearch által közzétett „Game based learning vs. gamification" [15] című, tudományos szemléletű szakmai anyagon alapul.

A digitális technológiák elterjedése forradalmasította a tanítás és a tanulás gya-

korlatát. Az új eszközök nem csupán az ismeretközlést alakítják át, hanem a pedagógiai szerepeket, a tanulás folyamatát és a tanulók aktív részvételét is új alapokra helyezik.

Pedagógiai előnyöknél érdemes megemlíteni, hogy a digitális oktatás a diákok körében mind a kreativitást, mind pedig az önállóságot képes magas szinten fejleszteni, mivel a digitális pedagógia arra ösztönzi a diákok kreativitását és ösztönzi őket abban, hogy önállóan szerezzenek ismeretek miközben például versengenek egymással. A digitális játékok és eszközök segítségével a diákoknak lehetőségük van gyorsabban és szélesebb körben szert tenni ismeretekre, ezen felül nagyban hozzá járul a térlátásuk, vizuális képességük fejlesztésében is. Abból kifolyólag pedig, hogy napjainkban az okos eszközök mennyire elterjedtek és milyen népszerűségnek örvendenek a fiatal diákok körében, az ilyen eszközök használata növelheti a diákok motivációját és elköteleződését a tanulási folyamatok iránt.

Pedagógiai kihívások alatt érdemes megemlíteni az alábbi négy pontot:

1.Tanári szerep átalakulása: a digitális eszközök és játékok megkövetelik, hogy a pedagógusok naprakészek legyenek és új ismereteket sajátítsanak el.

2.Az okos eszközök elérhetősége: a digitális pedagógia játékok vagy eszközök által ugyan hatékony, de nem minden diák számára elérhető, ami a diákok körében egyenlőtlenségekhez vezethet .

3.Digitális játékok hiánya: a tanárok esetében előfordulhat a digitális játékok hiánya, ami akadályozhatja a játékok és az okoseszközök használatát a tanórák keretein belül.

4.Információs túlterheltség: a digitális környezetben a tanulók könnyen túlterheltségbe kerülhetnek, amennyiben a digitális oktatás játékok alapján nem fokozatosan kerül bevezetésre, ez pedig csökkentheti a tanulás hatékonyságát és demotiváltsághoz vezethet .

2. fejezet

A Python programozási nyelv és a Kivy keretrendszer a játékfejlesztésben

2.1. A Python nyelv áttekintése

Ez az alfejezet Lutz, M. „Learning Python" [9] című könyvének szakmai feldolgozásán alapul.

A Python napjaink egyik legismertebb és legnépszerűbb programozási nyelve, amely világszerte több mint egymillió aktív felhasználóval rendelkezik. Az oktatás területén különösen az utóbbi években vált meghatározóvá, mivel könnyen tanulható, jól strukturált nyelvként ideális bevezető nyelv kezdő programozók számára.

A Python szintaxisát úgy tervezték, hogy az egyszerű, letisztult és jól olvasható legyen. Ez nemcsak a tanulást könnyíti meg, hanem a programkód karbantarthatóságát és újrafelhasználhatóságát is elősegíti. Ennek köszönhetően a diákok már a tanulás korai szakaszában is képesek összetettebb programok írására és megértésére, összehasonlítva például a Java vagy a C++ szintaktikailag bonyolultabb felépítésével
.

A Python sokoldalúságát jól mutatja, hogy nem csupán oktatási célokra alkalmas, hanem számos más területen is sikerrel alkalmazzák, például webfejlesztésben, adatbázis-kezelésben, játékfejlesztésben vagy akár tudományos és numerikus számításokban is. Ebben fontos szerepet játszanak a Pythonhoz elérhető széles körű beépített és külső könyvtárak .

2.2. A Pygame és Kivy összehasonlítása

Ez a fejezet Aliyev, M. „Comparative analysis of Kivy and other mobile development platforms" [1] című tanulmányának feldolgozásán alapul.

A Pygame és Kivy két népszerű Python alapú keretrendszer a játékfejlesztő iparban, mivel mindkettővel interaktív alkalmazásokat és játékokat tudunk létrehozni. Míg a Pygame a 2D-s játékokra és alkalmazásokra specializálódott addig a Kivy egy komplexebb viszont egyben modernebb crossplatformos keretrendszer, amelynek segítségével a felhasználó nem csak asztali, de mobilalkalmazások fejlesztésére is alkalmas .

A két keretrendszer összehasonlításánál fontos figyelembe venni, hogy míg a Pygame főként asztali rendszerekre optimalizált, addig a Kivy fut iOS-en, Androidon, Windowson, Linuxon, és MacOS-en is, anélkül, hogy az eredeti alkalmazásunk kódsorában bármilyen módosítást is végeznénk .

Fejlesztési sebességet tekintve, ugyan mindkét keretrendszer Python alapú, de a design, GUI tervezésekor a Pygame több manuális munkát igényel, mint a Kivy, azaz az utóbbival gyorsabban tudunk létrehozni alkalmazás vagy játék prototípusokat.

Az UI lehetőségeket tekintve a Pygame-ben az összes elemet, tehát például a gombokat és címkéket egyedileg kell leprogramozni. Ezzel ellentétben a Kivy rendelkezik beépített widgetekkel ezzel is időt spórolva a diákoknak, vagy fejlesztőknek .

Használatát tekintve a Pygame ismertebb és elterjedtebb a klasszikus oktatásban, mint a Kivy, amely ugyan kevesebb aktív felhasználóval rendelkezik, mégis egyedibb alkalmazások készítésére alkalmas .

Összefoglalva a Kivy mobilbarátabb, míg a Pygame-el egyszerűbb asztali feladatok elvégzéséhez ideális. Véleményem szerint az oktatásban mindkét keretrendszer használata elengedhetetlen. A Pygame segítségével a diákok betekintést nyerhetnek a játékfejlesztés alapjaiba, míg a Kivy segítségével egyedibb és komplexebb játékokat hozhatnak létre.

2.3. A Kivy keretrendszer bemutatása

Ez az alfejezet a Kivy hivatalos dokumentációjának [6] szakmai tartalmára épül.

A Kivy keretrendszer egyik legnagyobb előnye a crossplatform támogatás mellett, hogy számos beépített könyvtárral rendelkezik, amelyek segítségével könnyebben kialakíthatóak interaktív alkalmazások és játékok egyaránt .

Az alfejezetemben szeretnék pár főbb Kivy modult bemutatni, amelyek nagy segítséget nyújtottak a szakdolgozatomban készített játék létrehozásában:

Label – ez egy egyszerű szövegmegjelenítő elem, amelyet akkor használunk alkalmazásokban, ha instrukciót, kérdést vagy címet szeretnénk megjeleníteni. A megjelenített szöveg mérete, színe és betűtípusa is testreszabható .

Button – ez egy interaktív elem, amely kattintásra egy általunk előre megadott eseményt fog elindítani. A Kivy ezen verérlőeleme egy játék létrehozásánál elengedhetetlen, mivel ennek segítségével töltjük be például a szinteket, vagy ellenőrizzük, hogy a válaszunk helyes vagy helytelen .

TextInput – ez a modul lehetőséget biztosít arra, hogy a felhasználó szöveget vigyen be az alkalmazásba. A funkciót leggyakrabban névmegadásnál, vagy saját válasz begépelésére használják a játékokban .

ScreenManager – a modul lehetővé teszi több képernyő kezelését is egy alkalmazáson belül, ez szintén elengedhetetlen eleme egy játéknak vagy alakalmazásnak, mivel az alábbi modul segítséget nyújt abban, hogy például a főképernyő, szintek, vagy ranglista menüpontok között gond nélkül ugráljon a felhasználó.

2.4. Oktatási játék készítése Kivyben

Ez az alfejezet a Python nyelv json moduljának funkcióit és használatát ismertető hivatalos dokumentáció [14] feldolgozásán alapul.

A Kivy keretrendszer az egyik legjobb az oktatási játékok készítéséhez és fejlesztéséhez, mivel a beépített könyvtárai segítségével könnyebben lehetősége van a felhasználónak komplexebb többszintű játékok létrehozására. Ebben a fejezetben szeretném bemutatni azon fő modulokat, amelyek elengedhetetlen részét képezték a játékom oktatási részének létrehozásához .

Json fájl és annak használata a kérdésekhez:

A kérdések és válaszlehetőségek tárolásához a JSOn fájl használata a legopcionálisabb. Ez egy egyszerű szövegfájl amely tárolja a szükséges adatokat, jelen

esetben a diákok elért pontszámjait. Pythonban ezt a json modul segítségével az alábbi módon tudjuk beépíteni a játékunkba:

```
import json
with open( ranglista .json , r , encoding=
    utf -8 ) as f:
  ranglista = json.load(f) [14].
```

Pontrendszer:

A pontrendszer segítségével visszajelzést tudunk adni a diákok számára az eddig elért teljesítményükről. A pontokat minden helyes válasz után növeljük, és akár a képernyőre is megjeleníthetjük egy Label vagy ProgressBar modul segítségével.

Többszintek megvalósítása:

Egy komplexebb játéknál, különösképpen az oktatásban érdemes szintrendszert létrehozni. Erre a Kivy egy egyszerű megoldást kínál, amelyre a ScreenManager modul használható, ahol minden szint egy külön Screen objektumként szerepel .

Rangsor készítés: A rangsor készítése lehetővé teszi a diákok számára, hogy összehasonlítsák eredményeiket, versengés alakuljon ki náluk ezzel is motiválva őket a jobb eredmények elérésében. Ez gyakorlatban annyit jelent, hogy miután meghívjuk a json fájlunkat, amelyekben a diákok pontszámaik kerültek tárolásra, egy egyszerű parancs segítségével rendezni tudjuk őket, a TextInput segítségével pedig, mint ahogy azt fentebb is taglaltam a játékosok nevét is bekérhetjük egyidejűleg .

Például:

```
1    with open("rangsor.json", "r") as f:
2        lista = json.load(f)
3
4    rendezettlistsa = sorted(lista, key=lambda x: x["pont"],
   reverse=True)
```

3. fejezet

A Játék

A szakdolgozatom játékát és annak grafikus felületét a Kivy keretrendszer segítségével hoztam létre, amely könnyen átlátható és letisztult felületet kínál. A játékom célja az, hogy a hetedik osztályos algebrát a diákok kicsit más, interaktívabb módon tudják elsajátítani és begyakorolni. A Játékom kódsora 3 fő részre bontható:

- A szükséges könyvtárak importálása, valamint a grafikus felület és vezérlőgombok létrehozása
- A játék szintjeinek és feladatinak generálása
- A program működéséért felelős osztályok és függvények

3.1. A grafikus felületért felelős kódsor

```
1     from kivy.app import App
2 from kivy.uix.screenmanager import ScreenManager, Screen,
      FadeTransition
3 from kivy.lang import Builder
4 from kivy.properties import NumericProperty, StringProperty,
      ListProperty
5 from kivy.core.window import Window
6 from kivy.metrics import dp
7 from kivy.uix.button import Button
8 from kivy.uix.label import Label
9 import random, json, os
10
```

```
Window.size = (360, 640)
Window.clearcolor = (1, 1, 1, 1)

KV = '''
<MainMenu>:
    name: 'menu'
    canvas.before:
        Color:
            rgba: 0.972, 0.976, 0.980, 1
        Rectangle:
            pos: self.pos
            size: self.size
        Color:
            rgba: 0.416, 0.353, 0.804, 1
        Rectangle:
            pos: self.x, self.top - dp(56)
            size: self.width, dp(56)
    BoxLayout:
        orientation: 'vertical'
        padding: dp(20)
        spacing: dp(20)
        Label:
            text: 'Algebrai Játék'
            font_size: '20sp'
            size_hint_y: None
            height: dp(56)
            color: 1, 1, 1, 1
            halign: 'center'
            valign: 'middle'
            text_size: self.size
        Widget:
            size_hint_y: None
            height: dp(10)
        Label:
```

```
            text: 'Középiskola'
            font_size: '32sp'
            color: 0.416, 0.353, 0.804, 1
            halign: 'center'
            size_hint_y: None
            height: self.texture_size[1]
        Label:
            text: 'Algebrai Játék'
            font_size: '16sp'
            color: 0.576, 0.439, 0.859, 1
            halign: 'center'
            size_hint_y: None
            height: self.texture_size[1]
        BoxLayout:
            size_hint_y: None
            height: dp(100)
            padding: dp(20)
            spacing: dp(20)
            canvas.before:
                Color:
                    rgba: 0.937, 0.929, 0.961, 1
                RoundedRectangle:
                    pos: self.pos
                    size: self.size
                    radius: [dp(10),]
            Label:
                text: '    '
                font_size: '32sp'
                color: 0.416, 0.353, 0.804, 1
            Label:
                text: '    '
                font_size: '32sp'
                color: 0.416, 0.353, 0.804, 1
            Label:
```

```
79                 text: '    '
80                 font_size: '32sp'
81                 color: 0.416, 0.353, 0.804, 1
82         TextInput:
83             id: player_name_input
84             hint_text: 'Írd be a neved'
85             multiline: False
86             size_hint_y: None
87             height: dp(40)
88 
89         Button:
90             text: 'Új játék'
91             size_hint_y: None
92             height: dp(50)
93             background_normal: ''
94             background_color: 0.416, 0.353, 0.804, 1
95             color: 1, 1, 1, 1
96             on_release:
97                 app.root.get_screen('select').player_name =
                        player_name_input.text
98                 root.manager.current = 'select'
99 
100 
101        Button:
102            text: 'Ranglista'
103            size_hint_y: None
104            height: dp(50)
105            background_normal: ''
106            background_color: 1, 1, 1, 1
107            color: 0.416, 0.353, 0.804, 1
108            on_release: root.manager.current = 'ranking'
109 
110 
111        Button:
```

```
112             text: 'Kilépés'
113             size_hint_y: None
114             height: dp(50)
115             background_normal: ''
116             background_color: 1, 1, 1, 1
117             color: 0.416, 0.353, 0.804, 1
118             on_release: app.stop()
119 <RankingScreen>:
120     name: 'Rangsor'
121     BoxLayout:
122         orientation: 'vertical'
123         padding: dp(20)
124         spacing: dp(10)
125         Label:
126             text: 'Ranglista'
127             font_size: '20sp'
128             color: 0,0,0,1
129         ScrollView:
130             GridLayout:
131                 id: ranking_box
132                 cols: 1
133                 size_hint_y: None
134                 height: self.minimum_height
135         Button:
136             text: 'Vissza'
137             on_release: root.manager.current = 'menu'
138
139 <LevelSelect>:
140     name: 'select'
141     canvas.before:
142         Color:
143             rgba: 1, 1, 1, 1
144         Rectangle:
145             pos: self.pos
```

```
146             size: self.size
147         Color:
148             rgba: 0.416, 0.353, 0.804, 1
149         Rectangle:
150             pos: self.x, self.top - dp(56)
151             size: self.width, dp(56)
152     GridLayout:
153         cols: 1
154         padding: dp(20)
155         spacing: dp(10)
156         Label:
157             text: 'Válassz szintet'
158             font_size: '20sp'
159             size_hint_y: None
160             height: dp(40)
161         ScrollView:
162             GridLayout:
163                 id: level_box
164                 cols: 1
165                 size_hint_y: None
166                 height: self.minimum_height
167                 spacing: dp(10)
168                 padding: dp(10)
169     Button:
170         text: 'Vissza'
171         size_hint_y: None
172         height: dp(40)
173         on_release: root.manager.current = 'menu'
174
175 <OptionButton@Button>:
176     size_hint_y: None
177     height: dp(50)
178     background_normal: ''
179     background_color: 0.8,0.8,1,1
```

```
180     text_size: self.width - dp(20), None
181     halign: 'left'
182     valign: 'middle'
183     padding_x: dp(10)
184     on_release: app.root.get_screen('game').select_level(int(self
            .level))
185
186 <GameScreen>:
187     name: 'game'
188     canvas.before:
189         Color:
190             rgba: 1, 1, 1, 1
191         Rectangle:
192             pos: self.pos
193             size: self.size
194         Color:
195             rgba: 0.416, 0.353, 0.804, 1
196         Rectangle:
197             pos: self.x, self.top - dp(56)
198             size: self.width, dp(56)
199     BoxLayout:
200         orientation: 'vertical'
201         padding: dp(20)
202         spacing: dp(10)
203         BoxLayout:
204             size_hint_y: None
205             height: dp(30)
206             Button:
207                 text: 'Vissza'
208                 on_release: root.manager.current = 'select'
209         Label:
210             text: 'Szint {}: {}'.format(root.current_level, root.
                    level_topic)
211             font_size: '18sp'
```

```
212             size_hint_y: None
213             height: dp(30)
214         Label:
215             text: 'Kérdés {}/{}'.format(root.index+1, root.total)
216             color: 0, 0, 0, 1
217             font_size: '16sp'
218             size_hint_y: None
219             height: dp(25)
220         Label:
221             text: root.question_text
222             color: 0, 0, 0, 1
223             font_size: '20sp'
224             text_size: self.width, None
225             size_hint_y: None
226             height: self.texture_size[1] + dp(10)
227         GridLayout:
228             id: options_box
229             cols: 1
230             size_hint_y: None
231             height: self.minimum_height
232             spacing: dp(10)
233         Label:
234             text: root.feedback_text
235             color: 0, 0, 0, 1
236             size_hint_y: None
237             height: dp(30)
238         Button:
239             text: 'Következő'
240             size_hint_y: None
241             height: dp(40)
242             on_release: root.next_question()
243         Button:
244             id: end_button
245             text: 'Pont mentése és vissza a főmenübe'
```

```
246                 size_hint_y: None
247                 height: dp(50)
248                 background_normal: ''
249                 background_color: 0.416, 0.353, 0.804, 1
250                 color: 1, 1, 1, 1
251                 opacity: 0
252                 disabled: True
253                 on_release:
254                     root.save_score()
255                     root.manager.current = 'menu'
```

Mint ahogy az a kódsor első kilenc sorában is látható, a program elején elengedhetetlen a játékhoz tartozó kulcsfontosságú könyvtárak importálása, ilyen például a Button, Window... amelyek nélkül a program ezen részei nem működnének megfelelően.

A Főmenü (MainMenu) képernyő:

A képernyő ezen része határozza megaz alkalmazás kezdő nézetének szerkezetét. A name:menu segítségével pedig hivatkozni fogunk majd a kódsor többi részében erre a képernyőre.

A canvas.before blokk a vizuális megjelenésért felel: a Color és Rectangle utasítások a háttérszínt és egy felső fejlécsávot jelenítenek meg, amely vizuálisan kiemeli a játék címét.

A BoxLayout elrendezés függőleges irányban rendezi el az elemeket (orientation: 'vertical'), így a szövegek és a gombok egymás alatt helyezkednek el. A padding és spacing paraméterek határozzák meg az elemek közötti távolságokat és a belső margókat (jelen esetben mindkettő 20 dp).

A Label elemek különböző szövegeket jelenítenek meg. A 34. sorban például a játék főcíme szerepel. Egy másik Label blokk három matematikai szimbólumot tartalmaz, amelyek – bár közvetlenül nem részei a játékmenetnek – segíthetnek a diákok érdeklődésének felkeltésében vizuális érdekességükkel.

A Ranglista és Feladatválasztó képernyők hasonló szerkezetet követnek, így azok részletes elemzése ebben az esetben nem szükséges.

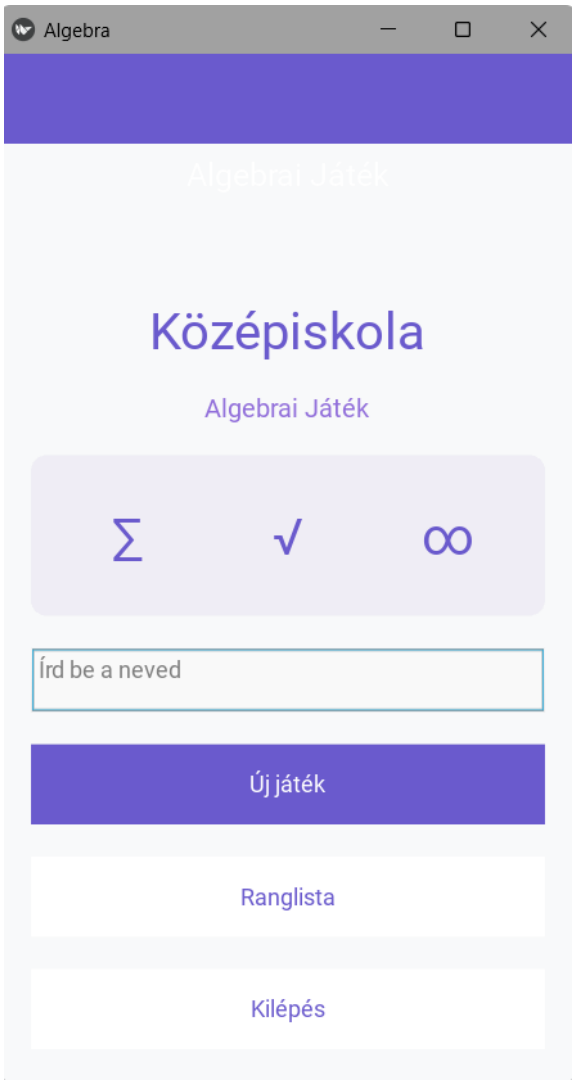
Az OptionButtonButton egy sablonként definiált válaszgomb, amely egyedi háttérszínt,

szövegigazítást és interaktív viselkedést kapott. A levelselect képernyőről hívja meg az adott feladatot az on_release eseményen keresztül. A GameScreen képernyő tartalmazza a következő kulcsfontosságú elemeket:

- question_text: a kérdés szövegéért felelős parancs
- options_box: a válaszlehetőségekért felelős parancs
- feedback_text: a válasz utáni visszajelzésért felelős parancs
- next question: a következő kérdésre való ugrásért felelős parancs
- save_score: a pontszám mentéséért felelős parancs.

A feladatok száma, a kérdések, és a pontszámok dinamikus frissítéséért a root objektumhoz kapcsolt Python függvények felelnek.

Ezek a képernyők és komponensek együttesen felelnek a játék logikai és vizuális részéért, valamint interaktív megvalósításáért.



3.1. ábra. Kezdőképernyő

3.2. A Játék szintjei és feladatai

Első feladatsor:

```
    'Builder.load_string(KV)

LEVEL_TOPICS = {
    1: "Függvények",
    2: "Függvény megadásának módjai",
    3: "Lineáris Függvények Táblázatból",
    4: "Lineáris függvények tulajdonságai",
    5: "Egyenletek",
    6: "Lineáris egyenletek grafikon pontjai",
    7: "Lineáris egyenletrendszerek",
    8: "Másodfokú függvények",
    9: "Exponenciális és logaritmus függvények",
    10: "Vegyes feladatok"
}


def load_level_1():
    """Első szint: Függvények"""
    data = {
    "level": 1,
    "topic": "Függvények",
    "questions": [
      {
        "id": 1,
        "question": "Egy függvény hozzárendeli az x értékhez
           annak kétszeresét. Mi a függvényérték, ha x = 4?",
        "options": ["6", "8", "12", "16"],
        "answer": "8"
      },
      {
        "id": 2,
        "question": "Melyik hozzárendelés nem függvény?",
```

```
      "options": [
        "Minden emberhez hozzárendeljük a születési évét.",
        "Minden autóhoz hozzárendeljük a rendszámát.",
        "Minden diáknak hozzárendelünk egy osztályfőnököt.",
        "Minden emberhez hozzárendeljük a testvérét."
      ],
      "answer": "Minden emberhez hozzárendeljük a testvérét."
    },
    {
      "id": 3,
      "question": "Egy függvény x-hez az x+5 értéket rendeli.
         Milyen értéket kapunk, ha x = -3?",
      "options": ["2", "1", "-8", "5"],
      "answer": "2"
    },
    {
      "id": 4,
      "question": "Melyik grafikon nem lehet egy függvényé?",
      "options": [
        "Egy egyenes, ami metszi az y tengelyt.",
        "Egy parabola.",
        "Egy kör.",
        "Egy csökkenő lineáris függvény."
      ],
      "answer": "Egy kör."
    },
    {
      "id": 5,
      "question": "Egy függvény hozzárendeli az x értékhez az x
         négyzetét. Mi az érték, ha x = -2?",
      "options": ["-4", "4", "-2", "0"],
      "answer": "4"
    },
    {
```

```
64          "id": 6,
65          "question": "Melyik állítás igaz minden függvényre?",
66          "options": [
67            "Egy x értékhez több y értéket is rendelhet.",
68            "Minden függvény lineáris.",
69            "Egy x értékhez legfeljebb egy y érték tartozhat.",
70            "Minden függvénynek van grafikonja."
71          ],
72          "answer": "Egy x értékhez legfeljebb egy y érték
                tartozhat."
73        },
74        {
75          "id": 7,
76          "question": "Ha f(x) = 2x - 1, akkor mennyi f(3)?",
77          "options": ["5", "6", "7", "8"],
78          "answer": "5"
79        },
80        {
81          "id": 8,
82          "question": "Egy függvény hozzárendeli x-hez az 5 - x ért
                éket. Milyen értéket kapunk, ha x = 7?",
83          "options": ["-2", "2", "12", "-12"],
84          "answer": "-2"
85        },
86        {
87          "id": 9,
88          "question": "Melyik függvény NEM monoton növekvő?",
89          "options": [
90            "f(x) = x + 2",
91            "f(x) = 3x",
92            "f(x) = -x",
93            "f(x) = 0,5x + 1"
94          ],
95          "answer": "f(x) = -x"
```

```
96          },
97          {
98            "id": 10,
99            "question": "Egy függvény hozzárendeli az x számhoz az x
                /2 értéket. Mi az érték, ha x = 10?",
100           "options": ["2", "4", "5", "10"],
101           "answer": "5"
102         }
103     ]
104   }
105     return data["questions"]
```

Magyarázat:

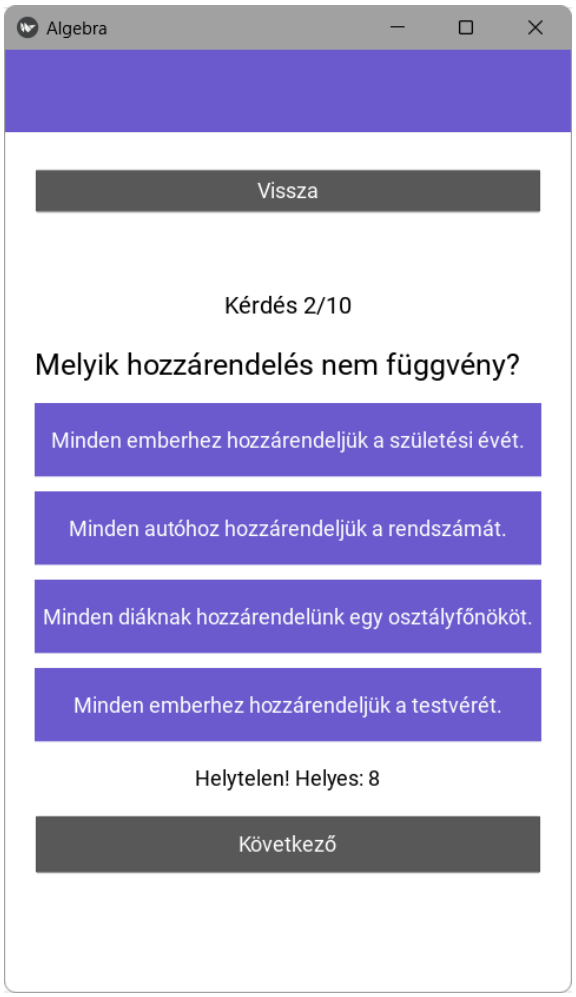
A LEVEL_TOPICS tartalmazza a játék tíz feladatához rendelt matematikai témaköröket. Ezek alapján történik meg a feladatok címének dinamikus betöltése, illetve ezek határozzák meg a kérdések típusát is.

3.2. ábra. Feladatsorválasztás

A load_level_1() függvény az első feladathoz tartó kérdés adatokat tárolja. Ez a feladat a hetedikes algebra alapfogalmaira és egyszerű hozzárendelési szabályaira koncentrál, megismertetve a diákokkal a függvények fogalmát és alapvető tulajdonságait kicsit interaktívabb módon. A tíz kérdésben olyan kulcsfontosságú kérdések szerepelnek, mint például:

- konkrét számok behelyettesítése egyszerű kifejezésekbe,
- függvénydefiníciók felismerése szöveges példák alapján,
- grafikonnal kapcsolatos értelmezések (pl. kör nem lehet függvény),
- olyan elméleti fogalmak, mint a monotonitás vagy az egyértelmű hozzárendelés.

A válaszlehetőségek segítik a tanulót az összefüggések gyakorlati megértésében, míg a return data["questions"] parancs biztosítja a kérdések betöltését a játékon belül.



3.3. ábra. 1. Feladatsor példa

Második feladatsor:

```
    def load_level_2():
    """Második szint: Függvény megadásának módjai"""
    data = {
    "level": 2,
    "topic": "Függvény megadásának módjai",
    "questions": [
      {
        "id": 1,
        "question": "Melyik NEM egy függvény megadási mód?",
        "options": ["Képlettel", "Táblázattal", "Szöveges leírá
            ssal", "Véletlenszerű számmal"],
        "answer": "Véletlenszerű számmal"
      },
```

```
    {
      "id": 2,
      "question": "Mit jelent az, hogy egy függvényt táblá
          zattal adunk meg?",
      "options": [
        "Egy képletet használunk a számításhoz",
        "Grafikonon ábrázoljuk az értékeket",
        "Párokba rendezzük az x és hozzátartozó y értékeket egy
            táblázatban",
        "Szövegesen elmondjuk, mit csinál a függvény"
      ],
      "answer": "Párokba rendezzük az x és hozzátartozó y érté
          keket egy táblázatban"
    },
    {
      "id": 3,
      "question": "Egy függvényt így adunk meg: f(x) = 2x + 1.
          Milyen típusú megadás ez?",
      "options": ["Táblázatos", "Grafikus", "Szöveges", "Ké
          pletes"],
      "answer": "Képletes"
    },
    {
      "id": 4,
      "question": "Mi jellemző a szöveges megadásra?",
      "options": [
        "Egy x-hez tartozó y értékeket sorolunk fel",
        "Leírjuk szavakkal, hogyan alakul az y érték x szerint"
            ,
        "Egy egyenletet írunk fel",
        "Egy táblázatban soroljuk az értékeket"
      ],
      "answer": "Leírjuk szavakkal, hogyan alakul az y érték x
          szerint"
```

```
40        },
41        {
42          "id": 5,
43          "question": "A következő megadás milyen típusú: 'Az x szá
              mhoz az x-et kétszerezve adjuk az eredményt'?",
44          "options": ["Képletes", "Szöveges", "Táblázatos", "
              Grafikus"],
45          "answer": "Szöveges"
46        },
47        {
48          "id": 6,
49          "question": "Hogyan lehet képletes megadásból táblázatos
              megadást készíteni?",
50          "options": [
51            "Kiszámoljuk több x értékre az y értéket és egy táblá
                zatba írjuk őket",
52            "Lerajzoljuk a képletet",
53            "Szöveggé alakítjuk",
54            "Változókat kicseréljük"
55          ],
56          "answer": "Kiszámoljuk több x értékre az y értéket és egy
              táblázatba írjuk őket"
57        },
58        {
59          "id": 7,
60          "question": "A f(x) = -x + 3 függvényt hogyan adhatjuk
              meg grafikusan?",
61          "options": [
62            "Megadjuk az értelmezési tartományt",
63            "Kiszámoljuk néhány x-re az y-t, pontokat rajzolunk,
                majd összekötjük őket",
64            "Leírjuk szavakkal a szabályt",
65            "Nem lehet grafikusan megadni"
66          ],
```

```
67          "answer": "Kiszámoljuk néhány x-re az y-t, pontokat
                rajzolunk, majd összekötjük őket"
68        },
69        {
70          "id": 8,
71          "question": "Mi a grafikus megadás előnye?",
72          "options": [
73            "Pontosan kiszámolhatók az értékek",
74            "Rögtön látható a függvény viselkedése",
75            "Egyszerűbb leírni szavakkal",
76            "Könnyebb elrejteni a hibákat"
77          ],
78          "answer": "Rögtön látható a függvény viselkedése"
79        },
80        {
81          "id": 9,
82          "question": "Ha egy függvényt képlettel adunk meg, mit
                NEM tudunk közvetlenül belőle?",
83          "options": [
84            "Bármely x-hez tartozó y értéket",
85            "Hogy milyen típusú a függvény",
86            "A konkrét grafikus ábráját",
87            "A hozzárendelés szabályát"
88          ],
89          "answer": "A konkrét grafikus ábráját"
90        },
91        {
92          "id": 10,
93          "question": "Az alábbiak közül melyik NEM jó példa ké
                pletes megadásra?",
94          "options": ["f(x) = 3x - 2", "y = 0,5x + 1", "Az x-hez az
                x négyzetét rendeljük", "g(x) = x / 2"],
95          "answer": "Az x-hez az x négyzetét rendeljük"
96        }
```

```
97        ]
98    }
99        return data["questions"]
```

Magyarázat:

A load_level_2() függvény a második feladatsor kérdéskészletét tartalmazza, amely a függvények megadásának különböző módjaira fókuszál. Ez a feladatsor segít rendszerezni azokat az alapvető ismereteket, melyeket a diákok a függvények leírásáról tanulnak meg: mikor beszélünk képletes, szöveges, táblázatos vagy grafikus megadásról, és milyen esetekben melyik a legpraktikusabb.

A kérdések célja, hogy a tanulók könnyen felismerjék az egyes megadási formákat, tudják őket egymásba átalakítani (például képletből táblázat), és képesek legyenek megérteni, hogy az egyes formák milyen előnyökkel vagy korlátokkal járnak.

Ez a feladatsor már elindítja a diákokat abba az irányba, hogy ne csak alkalmazzák, hanem össze is tudják hasonlítani a különböző megadási módokat, és hogy átlássák, hogy ugyanazt a függvényt többféleképp is le lehet írni – attól függően, épp mire van szükség.

3.4. ábra. 2. Feladatsor példa

Harmadik feladatsor:

```
    def load_level_3():
    """Harmadik szint: Lineáris Függvények Táblázatból"""
    def generate_linear_question():
        a = random.choice([-3, -2, -1, 1, 2, 3])
        b = random.choice([-5, -3, 0, 2, 4, 6])

        x_values = list(range(-2, 3))  # -2, -1, 0, 1, 2
        y_values = [a * x + b for x in x_values]

        question_text = f"Az alábbi táblázat egy függvény érté
            keit tartalmazza:\n\n"
        table = "x: " + "  ".join([str(x) for x in x_values]) + "
            \n"
        table += "y: " + "  ".join([str(y) for y in y_values]) +
            "\n\n"
        question_text += table
        question_text += "Melyik függvény írja le ezt a hozzá
            rendelést?"

        correct = f"y = {a}x + {b}"
        wrong1 = f"y = {a+1}x + {b+1}"
        wrong2 = f"y = {-a}x + {b}"
        wrong3 = f"y = {a}x - {b}"

        options = [correct, wrong1, wrong2, wrong3]
        random.shuffle(options)

        return {
            "question": question_text,
            "options": options,
            "answer": correct
        }

```

```
30      return [generate_linear_question() for _ in range(10)]
```

Magyarázat:

A harmadik feladatsorunk, már szerkezetileg eltér az első kettő feladatsortól, mivel itt a kérdéseket és számadatokat a program véletlenszerűen generálja. A véletlenszerűen létrehozott kérdések és számok előnye, hogy a tanárok elkerülhetik az azonos feladatokat, így pedig a diákok kénytelenek önmaguk tudására hagyatkozni osztálytársaiké helyett. Ezen a feladatsoron is 10 kérdés kerül létrehozásra, minden kérdés egy-egy olyan táblázatot mutat be, ahol az x és az y értékek egy y = ax + b alakú lineáris összefüggés szerint kerülnek kiszámításra, majd a játékosnak ki kell választania, hogy a megadott lehetőségek közül melyik függvény képlet írja le helyesen a táblázat adatait.

A kérdések a generate_linear_question() függvény segítségével jönnek létre amely minden alkalommal az előre megadott a és b számhalmazból véletlenszerűen választott értékével dolgozik. Ez a fajta kérdésgenerálás elősegíti, hogy a tanulók ne csak felismerjék a szabályszerűségeket, hanem a függvények konkrét értelmezését is gyakorolják. A válaszlehetőségek között mindig szerepel a helyes megoldás, valamint három helytelen lehetőség annak érdekében, hogy a diákok méginkább elgondolkodjanak a megoldáson.

3.5. ábra. 3. Feladatsor példa

Negyedik feladatsor:

```
    def load_level_4():
    """Negyedik szint: Lineáris függvények tulajdonságai"""
    def generate_linear_properties_question():
        a = random.choice([-3, -2, -1, 1, 2, 3])
        b = random.choice([-5, -3, -1, 0, 2, 4, 5])

        question_type = random.choice([
            "növekedés/csökkenés",
            "metszi-e az origót",
            "meredekség meghatározása",
```

```
            "y-tengelymetszet",
            "x = 0 helyettesítése"
        ])


        question_text = ""
        correct = ""
        wrong = []


        if question_type == "növekedés/csökkenés":
            question_text = f"A(z) y = {a}x + {b} függvény hogyan viselkedik?"
            correct = "Növekvő" if a > 0 else "Csökkenő"
            wrong = ["Növekvő", "Csökkenő", "Állandó", "Ingovány os"]
            wrong.remove(correct)

        elif question_type == "metszi-e az origót":
            question_text = f"A(z) y = {a}x + {b} függvény metszi-e az origót (0;0)?"
            correct = "Igen" if b == 0 else "Nem"
            wrong = ["Igen", "Nem", "Részben", "Soha"]
            wrong.remove(correct)

        elif question_type == "meredekség meghatározása":
            question_text = f"Mi a meredeksége a következő függvénynek: y = {a}x + {b}?"
            correct = str(a)
            wrong = [str(a + 1), str(a - 1), str(-a), str(a * 2)]

        elif question_type == "y-tengelymetszet":
            question_text = f"Hol metszi a függvény az y tengelyt? y = {a}x + {b}"
```

```
40              correct = f"(0; {b})"
41              wrong = [f"(0; {b+1})", f"(0; {b-1})", f"({-b}; 0)",
                    f"(1; {b})"]
42
43          elif question_type == "x = 0 helyettesítése":
44              question_text = f"Mi az y értéke, ha x = 0 a y = {a}x
                    + {b} függvényben?"
45              correct = str(b)
46              wrong = [str(b + 1), str(b - 1), str(-b), str(b * 2)]
47
48
49          if question_type in ["metszi-e az origót", "növekedés/csö
                kkenés"]:
50
51              base = ["Igen", "Nem"] if question_type == "metszi-e
                    az origót" else ["Növekvő", "Csökkenő", "Állandó"]
52              options = [correct] + [o for o in base if o !=
                    correct]
53              while len(options) < 4:
54                  options.append("   ")
55              random.shuffle(options)
56          else:
57
58              options = [correct] + random.sample(wrong, 3)
59              random.shuffle(options)
60
61          return {
62              "question": question_text,
63              "options": options,
64              "answer": correct
65          }
66
67
68      return [generate_linear_properties_question() for _ in range
```

```
        (10) ]
```

Magyarázat:

A load_level_4() függvény a negyedik feladatsor kérdéseiért felel, amelyek a lineáris függvények fontosabb tulajdonságaira fókuszálnak. Itt a tanulóknak nem csak azt kell felismerniük, hogy milyen alakú a függvény, hanem azt is meg kell érteniük, hogyan viselkedik az adott függvény, milyen a meredeksége, hol metszi az y tengelyt és hogy metszi e az origót.

A kérdések ismételten dinamikusan generálódnak, az alkalmazás futtatásakor a program véletlenszerűen választ egy-egy típust a tulajdonságok közül, majd ehhez kapcsolódó kérdéseket készít.

Az opciók között most is négy esetet különböztetünk meg, amelyből három helytelen és egy helyes. Ez azért jó , mert a négy válaszlehetőség miatt a diákok kénytelenek átgondolni az összes válaszlehetőséget a függvény tulajdonságai segítségével ezzel is fejlesztve a tudásukat.

3.6. ábra. 4. Feladatsor példa

Ötödik feladatsor:

```
    def load_level_5():
    """Ötödik szint: Egyenletek"""
    def generate_equation_question():

        a = random.choice([-3, -2, -1, 1, 2, 3])
        c = random.choice([-5, -3, -1, 0, 2, 4, 5])

        x0 = random.choice([-2, -1, 0, 1, 2, 3])
        d = a * x0 + c

        question_text = f"Milyen x értékre igaz, hogy {a}x + {c}
```

```
            = {d}?"
        correct = str(x0)

        wrong = [
            str(x0 + random.choice([1,2])),
            str(x0 - random.choice([1,2])),
            str(-x0),
            str(x0 * random.choice([2,3]))
        ]

        # Válaszlehetőségek
        options = [correct] + random.sample(wrong, 3)
        random.shuffle(options)

        return {
            "question": question_text,
            "options": options,
            "answer": correct
        }


    return [generate_equation_question() for _ in range(10)]
```

Magyarázat: A load_level_5() függvény az ötödik feladatsorhoz tartozó kérdéseket generálja, amelyek az egyszerű elsőfokú egyenletek megoldására épülnek. A cél itt az, hogy a tanulók képesek legyenek fejből vagy egy gyors számolással megoldani egy adott egyenletet.

A kérdések felépítéséért a generate_equation_question() függvény felel, amelynek segítségével a kérdések és az értékeit véletlenszerű számokból generálódnak.

3.7. ábra. 5. Feladatsor Példa

Hatodik feladatsor:

```
    def load_level_6():
    """Hatodik szint: Lineáris egyenletek grafikon pontjai"""
    def generate_linear_equation_question():

        a = random.choice([-3, -2, -1, 1, 2, 3])
        b = random.choice([-3, -2, -1, 1, 2, 3])
        c = random.randint(-10, 10)


        question_text = f"Melyik pont van rajta az alábbi egyenes
            grafikonján?\n{a}x + {b}y = {c}"
```

```
11
12
13          x = random.randint(-3, 3)
14          if b != 0:
15              y = (c - a * x) / b
16              if y.is_integer():
17                  correct = f"({x}; {int(y)})"
18              else:
19
20                  x = random.randint(-3, 3)
21                  y = (c - a * x) / b
22                  if y.is_integer():
23                      correct = f"({x}; {int(y)})"
24                  else:
25
26                      x, y = 1, 1
27                      c = a * x + b * y
28                      correct = f"({x}; {y})"
29          else:
30              x = c / a
31              if x.is_integer():
32                  y = random.randint(-3, 3)
33                  correct = f"({int(x)}; {y})"
34              else:
35
36                  x, y = 1, 1
37                  c = a * x + b * y
38                  correct = f"({x}; {y})"
39
40          wrong = []
41          while len(wrong) < 3:
42              wx = random.randint(-5, 5)
43              wy = random.randint(-5, 5)
44              if a * wx + b * wy != c:
```

```
                wp = f"({wx}; {wy})"
                if wp != correct and wp not in wrong:
                    wrong.append(wp)

        options = [correct] + wrong
        random.shuffle(options)

        return {
            "question": question_text,
            "options": options,
            "answer": correct
        }

    return [generate_linear_equation_question() for _ in range
        (10)]
```

Magyarázat:

A hatodik feladatsor a lineáris egyenletek grafikon pontjai, ahol a diákok koordinátageometriai ismereteinek fejlesztése az elsődleges cél. Mindig az a fő kérdés, hogy egy adott pont rajta van-e az egyenes grafikonján, tehát kielégíti-e az adott egyenletet.

A generate_linear_equation_question() függvény segítségével egy $ax + by = c$ típusú általános elsőfokú egyenletet generál ahol a számok véletlenszerűek, és mindig az alábbi kérdés tevődik fel: „Melyik pont van rajta az alábbi egyenes grafikonján?”

A feladatsorban található logikai feltételek is, amik azért felelnek, hogyha esetleg a véletlenszerűen generált számnál az y nem egész, akkor újrapróbálkozik, ha pedig másodjára sem sikerül akkor fixen (1;1) pont kerül kiírásra.

Ezen feladatsor célja, hogy a diákok minél inkább elsajátítsák és begyakorolják a pontos ellenőrzését és az egyenesek kapcsolatát.

3.8. ábra. 6. Feladatsor példa

Hetedik feladatsor:

```
    def load_level_7():
    """Hetedik szint: Lineáris egyenletrendszerek"""
    def generate_linear_system_question():

        x0 = random.randint(-5, 5)
        y0 = random.randint(-5, 5)

        a1, b1 = random.randint(1, 5), random.randint(1, 5)
        a2, b2 = random.randint(1, 5), random.randint(1, 5)

        c1 = a1 * x0 + b1 * y0
```

```
        c2 = a2 * x0 + b2 * y0

        eq1 = f"{a1}x + {b1}y = {c1}"
        eq2 = f"{a2}x + {b2}y = {c2}"

        correct = f"({x0}; {y0})"

        wrong = []
        while len(wrong) < 3:
            wx = random.randint(-5, 5)
            wy = random.randint(-5, 5)
            if (wx, wy) != (x0, y0):
                wp = f"({wx}; {wy})"
                if wp not in wrong:
                    wrong.append(wp)

        options = [correct] + wrong
        random.shuffle(options)

        question_text = f"Melyik pont oldja meg az alábbi
            egyenletrendszert?\n{eq1}\n{eq2}"

        return {
            "question": question_text,
            "options": options,
            "answer": correct
        }

    return [generate_linear_system_question() for _ in range(10)]
```

Magyarázat:

A játék hetedik feladatsorát a lineáris egyenletrendszerek képezik, amelynek célja, hogy a tanulók ezt a témát is kellőképpen begyakorolják.

A generate_linear_system_question() függvény segítségével az alkalmazás két

elsőfokú egyenletet generál, és a diákoknak meg kell találniuk azt a pontot, amely mindkét egyenletet egyszerre kielégíti. A számadatok, mint ahogy az előző feladatsorokban is egy előre megadott halmazból véletlenszerűen kerülnek kiválasztásra.

3.9. ábra. 7. Feladatsor példa

Nyolcadik feladatsor:

```
    def load_level_8():
    """Nyolcadik szint: Behelyettesítéses egyenletrendszerek"""
    def generate_substitution_method_question():
        # Véletlen megoldáspont (x, y), ez lesz az
            egyenletrendszer megoldása
        x0 = random.randint(-5, 5)
        y0 = random.randint(-5, 5)

```

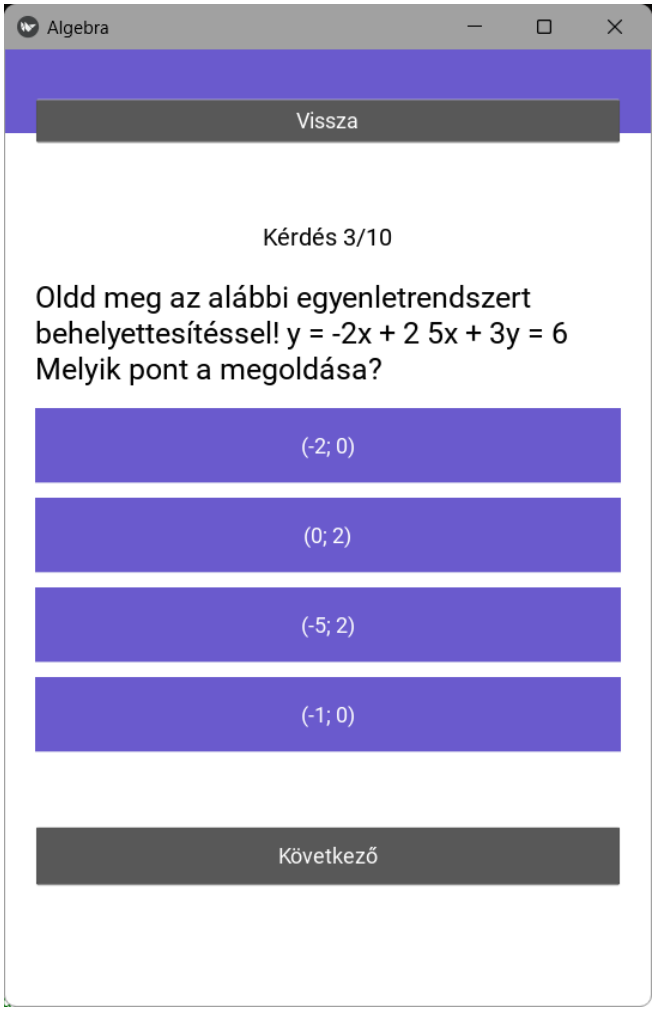
```
8
9           m = random.choice([-3, -2, -1, 1, 2, 3])
10          b = y0 - m * x0
11          eq1 = f"y = {m}x + {b}"
12
13          a = random.randint(1, 5)
14          b2 = random.randint(1, 5)
15          c = a * x0 + b2 * y0
16          eq2 = f"{a}x + {b2}y = {c}"
17
18          correct = f"({x0}; {y0})"
19
20          wrong = []
21          while len(wrong) < 3:
22              wx = random.randint(-5, 5)
23              wy = random.randint(-5, 5)
24              if (wx, wy) != (x0, y0):
25                  wp = f"({wx}; {wy})"
26                  if wp not in wrong:
27                      wrong.append(wp)
28
29          options = [correct] + wrong
30          random.shuffle(options)
31
32          question = f"""Oldd meg az alábbi egyenletrendszert
                behelyettesítéssel! {eq1} {eq2} Melyik pont a megoldá
                sa?"""
33
34          return {
35              "question": question,
36              "options": options,
37              "answer": correct
38          }
39
```

```
40      return [generate_substitution_method_question() for _ in
            range(10)]
```

Magyarázat:

Ezen a feladatsoron a lineáris egyenletrendszerek megoldására fókuszál az alkalmazás behelyettesítési módszerrel.

A program meghívja a generate_substitution_method_question() függvényt, amelynek segítségével kérdéseket állít elő véletlenszerűen. A program először kiválaszt egy véletlenszerű megoldáspárt az adathalmazokból ezután pedig két különböző egyenletet generál, a kódsor jegyzetében megemlített formában. A számadatok kiválasztása után a program mindig azt fogja kérni, hogy az egyenletrendszert behelyettesítési módszerrel oldjuk meg.



3.10. ábra. 8. Feladatsor példa

Kilencedik feladatsor:

```
    def load_level_9():
    """Kilencedik szint: Exponenciális és logaritmus függvények
       """
    def generate_exp_log_question():
        question_type = random.choice([
            "exponenciális",
            "logaritmus",
            "azonossag"
        ])

        if question_type == "exponenciális":
             Alap típusú kérdések: 2^x, 10^x, e^x számítása
            base = random.choice([2, 10, 3])
            x = random.choice([-2, -1, 0, 1, 2, 3])

            result = base ** x
            if x < 0 and base == 2:
                result_str = f"1/{base**(-x)}"
            else:
                result_str = str(result)

            question_text = f"Mennyi {base}^{x} értéke?"
            correct = result_str

            wrong = []
            if x < 0:
                wrong.append(str(base ** (-x))) %  Rossz elő
                   jellel
            wrong.append(str(base ** (x+1)))  % Eggyel nagyobb
               kitevő
            wrong.append(str(x ** base))  % Fordított hatványozás


            while len(wrong) < 3:
```

```
32                wrong_value = result + random.choice([-2, -1, 1,
                      2, 3])
33                if wrong_value > 0 and str(wrong_value) not in
                      wrong:
34                    wrong.append(str(wrong_value))
35
36            elif question_type == "logaritmus":
37
38                base = random.choice([2, 10])
39
40                exp = random.choice([1, 2, 3])
41                x = base ** exp
42
43                question_text = f"Mennyi log{base}({x})?"
44                correct = str(exp)
45
46                wrong = []
47                wrong.append(str(base))
48                wrong.append(str(x))
49                wrong.append(str(exp + 1))
50
51            elif question_type == "azonosság":
52
53                prop_type = random.choice([
54                    "szorzat",
55                    "hányados",
56                    "hatvány"
57                ])
58
59                if prop_type == "szorzat":
60                    question_text = "Melyik azonosság érvényes a
                          logaritmusra?"
61                    correct = "loga(M N) = loga(M) + loga(N)"
62                    wrong = [
```

```
                "loga(M N) = loga(M)     loga(N)",
                "loga(M N) = loga(M) - loga(N)",
                "loga(M N) = [loga(M)]^[loga(N)]"
            ]

        elif prop_type == "hányados":
            question_text = "Melyik azonosság érvényes a
                logaritmusra?"
            correct = "loga(M/N) = loga(M) - loga(N)"
            wrong = [
                "loga(M/N) = loga(M) / loga(N)",
                "loga(M/N) = loga(M) + loga(N)",
                "loga(M/N) = loga(M)     loga(N)"
            ]

        elif prop_type == "hatvány":
            question_text = "Melyik azonosság érvényes a
                logaritmusra?"
            correct = "loga(M^p) = p     loga(M)"
            wrong = [
                "loga(M^p) = [loga(M)]^p",
                "loga(M^p) = p / loga(M)",
                "loga(M^p) = loga(M) / p"
            ]

    options = [correct] + wrong
    random.shuffle(options)

    return {
        "question": question_text,
        "options": options,
        "answer": correct
    }

```

```
95      return [generate_exp_log_question() for _ in range(10)]
```

Magyarázat:

A játék kilencedik feladatsora három fő algebrai témát dolgoz fel, ezek:

• exponenciális kifejezések

• logaritmus értékek

• logaritmus azonosságok

Ebben a feladatsorban a generate_exp_log_question() függvény segítségével a program minden futtatáskor egy kérdéstípust választ a három közül, a kiválaszott típus alapján pedig az előre megadott halmazból választja a számadatokat, végül pedig a választ három hibás és egy helyes opcióval annak érdekében, hogy a számolás és ellenőrzés a diák számára elkerülhetetlen legyen.

3.11. ábra. 9. Feladatsor példa

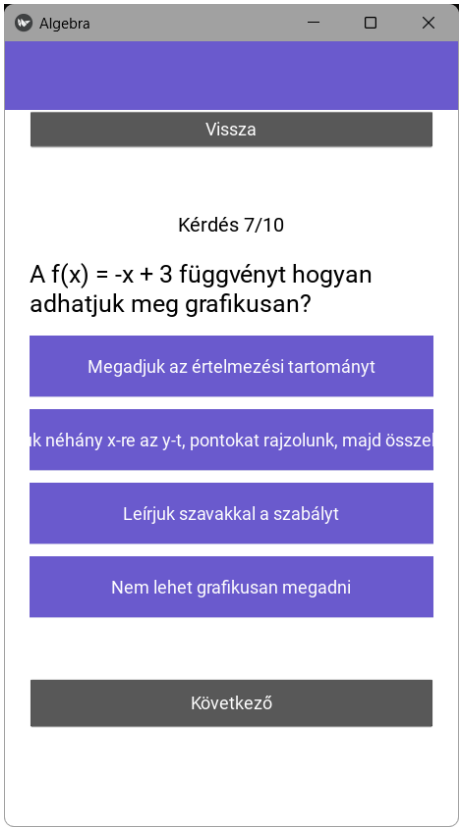
Tizedik feladatsor:

```
    def load_level_10():
    all_questions = []
    for lvl in range(1, 6):
        loader = globals().get(f'load_level_{lvl}')
        if loader:
            all_questions.extend(loader())

    random.shuffle(all_questions)
    return all_questions[:10]
```

Magyarázat:

A tizedik feladatsor egy összefoglaló feladatokként szolgál, amelynek segítségével a diákok az első 5 feladatsort ismétlik újra. A program ezekből a feladatsorokból véletlenszerűen választ ki és gyűjt össze kérdéseket, amelyekből összesen itt is tíz generálódik. A kérdéssorok minden futtatáskor különbözőek lesznek.



3.12. ábra. 10. Feladatsor példa

3.3. A program működéséért felelős osztályok és függvények

```
1    class MainMenu(Screen):
2    pass
```

A MainMenu osztály:

A MainMenu osztály a játék főmenüjéért felel és a Kivy keretrendszer Screen osztályát örökli, ami a játékkód elején lett definiálva. Mivel az osztály ezen felül másra nincs használva így csak egy pass található benne. Ez a szintaktikai elem azt jelenzi, hogy az osztály törzse üres, viszont bővíthető. Ettől függetlenül fontos szerepet tölt be a kódsorunkba és elengedhetetlen annak működéséhez.

```
class LevelSelect(Screen):
    player_name = StringProperty("")

    def start_game(self, level):
        game = self.manager.get_screen('game')
        game.player_name = self.player_name
        game.select_level(level)
        self.manager.current = 'game'


    def on_enter(self):
        box = self.ids.level_box
        box.clear_widgets()
        from kivy.factory import Factory
        for lvl, topic in LEVEL_TOPICS.items():
            btn = Factory.OptionButton(text=f"{lvl}. {topic}")
            btn.level = str(lvl)
            box.add_widget(btn)

    def select_level(self, level):
        game = self.manager.get_screen('game')
        game.player_name = self.player_name
        game.select_level(level)
```

A LevelSelect osztály:

Ez az osztály szintén a kód elején lévő Screen osztályból származik és célja a feladatsor választó képernyő létrehozása. Kódsorunk elején a player_name után definiált StringProperty segítségével tároljuk a játékos nevét.

Az egyik főmetódus a start_game, ami akkor kerül futtatásra, ha a játékos kiválaszt egy feladatsort. Ez a metódus először lekéri a game nevű képernyőt a képernyőmenedzsertől, majd átadja a játékos nevét, végül meghívja a select_level metódust, amely betölti a kiválasztott feladatsorhoz tartozó kérdéseket. A képernyő ezután átvált a játékra.

Az osztály második fontos része az on_enter metódus, amely mindig automatiku-

san lefut ha a játékos belép erre a képernyőre, itt először töröljük a kezdőképernyő elrendezését, majd a LEVEL_TOPICS segítségével létrehozzuk minden téma feladatsorára gombokat.

Végül pedig a select_level metódusnál kiválasztunk egy általunk tetszőleges feladatsort, ezen kódsorban biztosítjuk, hogy a játékos neve a képernyők váltakozása között véletlenül sem vesszen el.

```
    class GameScreen(Screen):
    player_name = StringProperty("")
    current_count = NumericProperty(0)
    current_level = NumericProperty(0)
    level_topic = StringProperty('')
    questions = ListProperty([])
    total = NumericProperty(0)
    index = NumericProperty(0)
    question_text = StringProperty('')
    feedback_text = StringProperty('')
    _answer = None

    def select_level(self, lvl):
        self.correct_count = 0
        self.current_level = lvl
        self.level_topic = LEVEL_TOPICS[lvl]
        loader = globals()[f'load_level_{lvl}']
        self.questions = loader()
        self.total = len(self.questions)
        self.index = 0
        self.feedback_text = ''
        self.display_question()
        self.manager.current = 'game'

    def display_question(self):
        q = self.questions[self.index]
        self.question_text = q['question']
        self._answer = q['answer']
```

```
29          box = self.ids.options_box
30          box.clear_widgets()
31          for opt in q['options']:
32              btn = Button(
33                  text=opt,
34                  size_hint_y=None,
35                  height=dp(50),
36                  background_normal='',
37                  background_color=(0.416, 0.353, 0.804, 1),
38                  color=(1, 1, 1, 1)
39              )
40              btn.bind(on_release=lambda inst, txt=opt: self.
                    check_answer(txt))
41              box.add_widget(btn)
42
43      def check_answer(self, selected):
44          if selected == self._answer:
45              self.correct_count += 1
46              self.feedback_text = 'Helyes!'
47          else:
48              self.feedback_text = f'Helytelen! Helyes: {self.
                    _answer}'
49
50
51      def next_question(self):
52          if self.index < self.total - 1:
53              self.index += 1
54              self.display_question()
55          else:
56              self.feedback_text = f'Vége! Pontszám: {self.
                    correct_count} / {self.total}'
57              self.ids.end_button.disabled = False
58              self.ids.end_button.opacity = 1
59
```

```
60
61
62      def save_score(self):
63          name = self.player_name.strip()  or "Névtelen"
64          entry = {
65              "name": name,
66              "score": self.correct_count,
67              "level": self.current_level
68      }
69          try:
70              with open("ranglista.json", "r", encoding="utf-8") as
                    f:
71               scores = json.load(f)
72          except:
73              scores = []
74
75          scores.append(entry)
76
77          with open("ranglista.json", "w", encoding="utf-8") as f:
78              json.dump(scores, f, ensure_ascii=False, indent=2)
```

A GameScreen osztály:

Ez az osztály már jóval összetettebb, mint az előző kettő, ez felel a játék fő menetéért, innen kapjuk vissza az összes fontos adatot, tartalmazza a kérdések megjelenítését, válaszok kezelését, pontszám rögzítését és a képernyő frissítését is.

A def select_level metódus felel a feladatsorok kilistázásáért, majd a kiválasztott feladatsor után a következő metódus, azaza a display_question segítségével elindítja az első kérdés megjelenését.

A display_question alapvetően azért felel, hogy minden kérdésnél az új gombokat létrehozza a megfelelő válaszokkal, és minden gombhoz hozzá van rendelve egy check_answer metódus, amellyel le ellenőrzi nekünk, hogy az adott válaszunk helyes, vagy helytelen.

A check_answer metódus, mint már említettem a kérdés ellenőrzéséért felel, ezen kívül ha a kérdés helyes növeli a pontszámot és visszajelzést ad, ellenkező esetben

kiírja a helyes megoldást, viszont a pontszámon nem változtat.

A kérdések között a next_question metódus segítségével váltogathatunk a kérdések között, ha megoldottuk azt. Ha az utolsó kérdésen is túlvagyunk kiírja az eredményt és a játék végét lezáró gombot, amit a program elején lévő GameScreen részben deklaráltunk.

Végül pedig a save_score metódus segít abban, hogy a játékos eredményei elmentésre kerüljenek, amit a ranglista.json fájlba ment le, ez később a főmenü ranglista részénél megtekinthető.

```
class RankingScreen(Screen):
def on_enter(self):
    self.ids.ranking_box.clear_widgets()
    try:
        with open("ranglista.json", "r", encoding="utf-8") as
            f:
            scores = json.load(f)
    except:
        scores = []

    scores = sorted(scores, key=lambda x: x.get("score", 0),
       reverse=True)[:10]
    for i, entry in enumerate(scores, 1):
        name = entry.get("name", "Névtelen")
        score = entry.get("score", 0)
        level = entry.get("level", "?")
        self.ids.ranking_box.add_widget(Label(
            text=f"{i}. {name} - {score} pont (Szint {level})
               ",
            color=(0, 0, 0, 1),
            size_hint_y=None,
            height=dp(30)
        ))
```

A RankingScreen osztály:

Ez az osztál a játék ranglistájáért felelős. Az osztály célja, hogy a játékosok

láthassák a legjobb eredményeket ezzel is viszályt és versengést kialakítva nálunk, növelve a motivációjukat és teljesítményüket.

Az on_enter metódus segítségével a játék betölti a ranglista.json fájl adatait, beleértve a diákok neveit és pontszámaikat. Ennek értelmében a diákok könnyen láthatják ki melyik feladatsoron hány pontot is ért el.

```
    class AlgebraApp(App):
    def build(self):
        sm = ScreenManager(transition=FadeTransition())
        sm.add_widget(RankingScreen(name='ranking'))
        sm.add_widget(MainMenu(name='menu'))
        sm.add_widget(LevelSelect(name='select'))
        sm.add_widget(GameScreen(name='game'))
        sm.current = 'menu'
        return sm

if __name__ == '__main__':
    AlgebraApp().run()
```

Az alkalmazás elindításáért és a teljes felhasználói felület egybentartásáért az AlgebraApp nevű osztály felel, amely a Kivy keretrendszer app osztályából származik. Ez a rövid kódsor határozza meg, hogy a program elindításakor milyen képernyők épülhetnek fel.

A program build metódusa négy különböző képernyőt ad hozzá a játékhoz, ez a ranglista, főmenü, feladatsor választás és a GameScreen, ami magát a játékot valósítja meg. Itt határozzuk meg az indulási sorrendet is.

A program utolsó sorában található AlgebraApp().run() metódus indítja el az alkalmazást és tölti be a grafikus felületet.

Összegzés

A szakdolgozatom célja a számítógépes játékok matematikai oktatásban betöltött szerepének bemutatása, különös tekintettel a Python nyelvre és a Kivy keretrendszerre épülő játékfejlesztés lehetőségeire. A kutatás és fejlesztés során azt vizsgáltam, hogyan lehet a játékosítás és az interaktív tanulási környezet hozzájárulni a diákok motivációjának növeléséhez és a tananyag mélyebb megértéséhez.

Az első fejezetben áttekintést adtam a számítógépes játékok oktatási célú alkalmazásának elméleti hátteréről, történeti fejlődéséről és pedagógiai vonatkozásairól. Bemutattam a játék alapú tanulás elméletét, különböző modelleket és keretrendszereket, valamint empirikus kutatási eredményeket. A fejezet zárásaként ismertettem a játékok taxonómiáját, valamint azokat a pedagógiai előnyöket és kihívásokat, amelyekkel a tanárok szembesülhetnek a játékosított oktatás során.

A második fejezet középpontjában a Python programozási nyelv és a Kivy keretrendszer állt, amelyek technikai alapot adtak a saját fejlesztésű oktatójátékhoz. Összehasonlítottam a Pygame és a Kivy könyvtárakat, majd részletesen bemutattam a Kivy lehetőségeit, különösen az oktatási célú alkalmazások készítésének szempontjából.

A harmadik fejezetben bemutattam a saját készítésű játékot, amely az algebrai ismeretek gyakorlását segíti elő interaktív, vizuális módon. Részletesen elemeztem a játék grafikus felületét, a különböző szintek felépítését és a feladatok logikai szerkezetét. Emellett kitérek a program működéséért felelős főbb osztályokra és függvényekre, amelyek a játék logikáját irányítják

Irodalomjegyzék

[1] 1. Aliyev, M. (2024). Comparative analysis of Kivy and other mobile development platforms. International Journal of Scientific Research and Management (IJSRM), 12(06), EC 2024 1277.3993

[2] 2. Anderson, L.W., Krathwohl, D.R. (Eds.). (2001). A taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives. Longman.

[3] 3. Clark, R.C., Mayer, R.E. (2016). e Learning and the science of instruction (4thed.). Wiley.

[4] 4. Gee, J.P. (2003). What video games have to teach us about learning and literacy. Palgrave Macmillan..

[5] 5. Hunicke, R., LeBlanc, M., Zubek, R. (2004). MDA: A formal approach to game design and game research. In Proceedings of the AAAI Workshop on Challenges in Game AI (pp.1–5). AAAI Press..

[6] 6. Kivy Organization. (n.d.). Kivy documentation. Retrieved May19,2025, from https://kivy.org/doc/stable/.

[7] 7. Krathwohl, D.R. (2002). A revision of Bloom's taxonomy: An overview. Theory into Practice, 41(4), 212–218

[8] 8. Luk'ianenko, K. (2014). Комп'ютерні ігри на уроках математики [Computer games in mathematics lessons]. Фізико математична освіта [Physical and Mathematical Education], 1(2), 19–25. Retrieved from http://fmo journal.fizmatsspu.sumy.ua/

[9] 9. Lutz, M. (2013). Learning Python (5thed.). O'Reilly Media. Retrieved from https://edu.anarchocopy.org/Programming

[10] 10. MindResearch Blog. (n.d.). Game based learning vs. gamification. Retrieved May19,2025, from https://blog.mindresearch.org/blog/game-based-learning-vs-gamification

[11] 11. MindTheGraph Blog. (n.d.). Mi az elméleti keretrendszer? Retrieved May19,2025, from https://mindthegraph.com/blog/hu/mi-az-elmeleti-keretrendszer/

[12] 12. Papert, S. (1980). Mindstorms: Children, computers, and powerful ideas. Basic Books.

[13] 13. Plass, J.L., Homer, B.D., Kinzer, C.K. (2015). Foundations of Game Based Learning. Educational Psychologist, 50(4), 258–283.

[14] 14. Python Software Foundation. (2025). json—JSON encoder and decoder. Python3Library Documentation. Retrieved May18,2025, fromhttps://docs.python.org/3/library/json.html

[15] 10. MindResearch Blog. (n.d.). Game based learning vs. gamification. Retrieved May19,2025, from https://blog.mindresearch.org/blog/game-based-learning-vs-gamification

[16] 16. Szűts, Z. (n.d.). [Doctoral dissertation, Eszterházy Károly University]. Retrieved May19,2025, from https://disszertacio.uni-eszterhazy.hu/82/1/Sz

[17] 17. Wouters, P., vanNimwegen, C., vanOostendorp, H., vanderSpek, E. (2013). A meta analysis of the cognitive and motivational effects of serious games. Journal of Educational Psychology, 105(2), 249–265

Ábrák jegyzéke

Резюме

Метою моєї дипломної роботи є представлення ролі комп'ютерних ігор у викладанні математики, з особливим акцентом на можливості розробки ігор на базі мови Python та фреймворку Kivy. У процесі дослідження та розробки я вивчав, як гейміфікація та інтерактивне навчальне середовище можуть сприяти підвищенню мотивації учнів і глибшому розумінню навчального матеріалу.

У першому розділі я представив огляд теоретичних основ застосування комп'ютерних ігор в освітніх цілях, їх історичного розвитку та педагогічних аспектів. Було розглянуто теорію навчання на основі гри, різні моделі й рамкові підходи, а також наведено результати емпіричних досліджень. На завершення розділу описано таксономію ігор, а також педагогічні переваги й виклики, з якими стикаються викладачі під час гейміфікованого навчання.

У другому розділі центральне місце займають мова програмування Python і фреймворк Kivy, які лягли в основу розробки моєї освітньої гри. Я порівняв бібліотеки Pygame та Kivy, після чого детально продемонстрував можливості Kivy саме з погляду створення застосунків для навчання.

У третьому розділі я представив власний ігровий продукт, що сприяє практичному закріпленню алгебраїчних знань через інтерактивний візуальний інтерфейс. Було проаналізовано графічний інтерфейс гри, структуру рівнів і логіку завдань. Крім того, розглянуто основні класи та функції, які забезпечують роботу програми та відповідають за ігрову логіку.

Nyilatkozat

Alulírott, Balog Krisztián, 014. Középiskolai oktatás (Matematika) képzési program hallgatója, kijelentem, hogy a dolgozatomat a II. Rákóczi Ferenc Kárpátaljai Magyar Főiskolán, a Matematika és Informatika Tanszéken készítettem, 014. Középiskolai oktatás (Matematika) BSc diploma megszerzése végett.

Kijelentem, hogy a dolgozatot más szakon korábban nem védtem meg, saját munkám eredménye, és csak a hivatkozott forrásokat (szakirodalom, eszközök stb.) használtam fel.

Tudomásul veszem, hogy dolgozatomat a II. Rákóczi Ferenc Kárpátaljai Magyar Főiskola könyvtárában a kölcsönözhető könyvek között helyezik el.

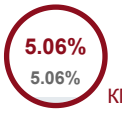
Звіт подібності

метадані

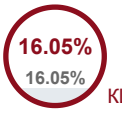
Назва організації
Hungarian College of Higher Education Ferenc Rakoczi II Transcarpathian
Заголовок
Vegleges_szakdolgozat_B_K
Науковий керівник / Експерт
Автор**Габріелла Пап**
підрозділ
Закарпатський угорський інститут імені Ференца Ракоці II

Обсяг знайдених подібностей

Коефіцієнт подібності визначає, який відсоток тексту по відношенню до загального обсягу тексту було знайдено в різних джерелах. Зверніть увагу, що високі значення коефіцієнта не автоматично означають плагіат. Звіт має аналізувати компетентна / уповноважена особа.



КП 1



КЦ

25
Довжина фрази для коефіцієнта подібності 2

18606
Кількість слів

118637
Кількість символів

Тривога

У цьому розділі ви знайдете інформацію щодо текстових спотворень. Ці спотворення в тексті можуть говорити про МОЖЛИВІ маніпуляції в тексті. Спотворення в тексті можуть мати навмисний характер, але частіше характер технічних помилок при конвертації документа та його збереженні, тому ми рекомендуємо вам підходити до аналізу цього модуля відповідально. У разі виникнення запитань, просимо звертатися до нашої служби підтримки.

Заміна букв		524
Інтервали	A→	0
Мікропробіли	·	0
Білі знаки		0
Парафрази (SmartMarks)	a	42

Подібності за списком джерел

Нижче наведений список джерел. В цьому списку є джерела із різних баз даних. Колір тексту означає в якому джерелі він був знайдений. Ці джерела і значення Коефіцієнту Подібності не відображають прямого плагіату. Необхідно відкрити кожне джерело і проаналізувати зміст і правильність оформлення джерела.

10 найдовших фраз	Колір тексту

ПОРЯДКОВИЙ НОМЕР	НАЗВА ТА АДРЕСА ДЖЕРЕЛА URL (НАЗВА БАЗИ)	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	https://link.springer.com/article/10.1007/s11423-020-09835-9	26 0.14 %
2	https://link.springer.com/chapter/10.1007/978-3-030-63539-8_5	22 0.12 %
3	https://link.springer.com/referenceworkentry/10.1007/978-981-19-0351-9_74-1	21 0.11 %
4	https://files.eric.ed.gov/fulltext/EJ1144034.pdf	20 0.11 %
5	https://ouci.dntb.gov.ua/en/works/9ZjerYa9/	17 0.09 %

6	https://ouci.dntb.gov.ua/en/works/9ZjerYa9/	17 0.09 %
7	https://ouci.dntb.gov.ua/en/works/9ZjerYa9/	16 0.09 %
8	http://ijlter.org/index.php/ijlter/article/view/1327	14 0.08 %
9	https://link.springer.com/article/10.1007/s12144-021-01744-1	14 0.08 %
10	https://files.eric.ed.gov/fulltext/ED081486.pdf	14 0.08 %

з домашньої бази даних (0.00 %)

ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)

з програми обміну базами даних (0.38 %)

ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	Professionele webapplicaties ontwikkelen in Swift met SwiftWasm: proof of concept aan de hand van een componentenbibliotheek [redacted] 4/26/2024 Hogeschool Gent (Scriptie)	66 (12) 0.35 %
2	Математичне та комп'ютерне моделювання хвиль у мотузковій системі [redacted] 12/3/2024 V. N. Karazin Kharkiv National University (KKNU) (Факультет математики і інформатики - кафедра прикладної математики)	5 (1) 0.03 %

з Інтернету (4.68 %)

ПОРЯДКОВИЙ НОМЕР	ДЖЕРЕЛО URL	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	https://www.duo.uio.no/bitstream/handle/10852/8946/1/larsskar.pdf	199 (34) 1.07 %
2	https://cyberleninka.ru/article/n/sde-math-a-software-package-for-the-implementation-of-strong-high-order-numerical-methods-for-ito-sdes-with-multidimensional-non	120 (22) 0.64 %
3	https://arxiv.org/pdf/1307.7042.pdf	75 (11) 0.40 %
4	https://ouci.dntb.gov.ua/en/works/9ZjerYa9/	50 (3) 0.27 %
5	https://www.ocf.berkeley.edu/~ddriver/wp-content/uploads/2015/04/PythonOOP.pdf	45 (6) 0.24 %
6	https://files.eric.ed.gov/fulltext/ED081486.pdf	44 (4) 0.24 %
7	https://cyberleninka.ru/article/n/a-systematic-approach-to-the-study-of-modern-students-image-sistemnyy-podhod-v-izuchenii-imidzha-sovremennyh-studentov	44 (6) 0.24 %
8	http://lipn.fr/~kanawati/python/M2207-cours.pdf	37 (5) 0.20 %
9	https://www.sri.com/wp-content/uploads/2021/12/digital-games-design-and-learning-executive_summary.pdf	36 (3) 0.19 %
10	https://link.springer.com/chapter/10.1007/978-3-030-63539-8_5	29 (2) 0.16 %
11	**https://link.springer.com/article/10.1007/s11423-020-09835-9**	**26 (1) 0.14 %**
12	https://link.springer.com/referenceworkentry/10.1007/978-981-19-0351-9_74-1	21 (1) 0.11 %
13	https://files.eric.ed.gov/fulltext/EJ1090277.pdf	21 (2) 0.11 %
14	https://www.mdpi.com/1424-8220/21/15/5079	21 (2) 0.11 %

15	https://files.eric.ed.gov/fulltext/EJ1144034.pdf	20 (1) 0.11 %
16	https://arxiv.org/pdf/2203.00846v1	18 (2) 0.10 %
17	https://open.library.ubc.ca/handle/2429/33407	15 (2) 0.08 %
18	https://link.springer.com/article/10.1007/s12144-021-01744-1	14 (1) 0.08 %
19	http://ijlter.org/index.php/ijlter/article/view/1327	14 (1) 0.08 %
20	http://repository.ub.ac.id/11479/6/Daftar%20Pustaka.pdf	12 (1) 0.06 %
21	http://www3.cs.stonybrook.edu/~lori/classes/games/EST579syllabus.html	10 (1) 0.05 %

Список прийнятих фрагментів (немає прийнятих фрагментів)

ПОРЯДКОВИЙ НОМЕР	ЗМІСТ	КІЛЬКІСТЬ ОДНАКОВИХ СЛІВ (ФРАГМЕНТІВ)