

# BMSZC Verebély László

## Technikum

### PROG-TANK

Témavezető:

Somogyi Erika

Készítette:

Szilágyi Gergely Csaba

Szoftverfejlesztő tagozat

Budapest

2022-2023

# Nyilatkozat

## a záródolgozat eredetiségéről

Alulírott Szilágyi Gergely Csaba (név) / Husz Erika (anyja neve), 569101ME (szem.ig. szám) büntetőjogi és fegyelmi felelősségem tudatában kijelentem és aláírással igazolom, hogy a záródolgozat saját munkám eredménye. A felhasznált irodalmi és egyéb információs forrásokat az előírásoknak megfelelően kezeltem, a záródolgozat készítésre vonatkozó szabályokat betartottam.

Kijelentem, hogy ahol mások eredményeit, szavait vagy gondolatait idéztem, azt a záródolgozatomban minden esetben, beazonosítható módon feltüntettem, a dolgozatban található fotók és ábrák közlésével pedig mások szerzői jogait nem sértem.

Kijelentem, hogy a záródolgozatom adathordozón található elektronikus változata teljes egészében megegyezik a nyomtatott formával.

Hozzájárulok ahhoz, hogy az érvényben lévő jogszabályok és a BMSZC Verebély László Technikum belső szabályzata alapján az iskola saját könyvtárában megtekinthető (olvasható) legyen a záródolgozatom.

Budapest, 2023. április 16.

Tanuló aláírása:.....

# Tartalomjegyzék

<i>Nyilatkozat</i> .....	2
<i>Tartalomjegyzék</i> .....	3
<i>Bevezetés</i> .....	5
<i>Témaválasztás</i> .....	6
<b>A Comedius Logo</b> .....	6
<i>Oktató jellegű programozási nyelvek története</i> .....	7
<i>Fejlesztői dokumentáció</i> .....	8
<b>Adatbázis terv</b> .....	8
Adatbázis célja .....	8
Táblák .....	8
<b>Osztályok és metódusok</b> .....	12
btnRegR_Click.....	17
btnLoginL_Click.....	19
ResetWorolod.....	21
setTankColor.....	22
GenerateMap.....	22
btnRunCodeClick.....	23
DrawTank .....	25
<b>Tesztelés</b> .....	25
Talált problémák .....	26
<b>Fejlesztési és Megvalósítási Ütemterv</b> .....	26
Fejlesztői környezet kialakítása .....	26
Adatbázis tervezés.....	26
Frontend .....	27
Backend.....	27
Tesztelés.....	27
<b>Fejlesztői környezet</b> .....	28
Szoftwerek .....	28
Hardwerek.....	28
<i>Felhasználói dokumentáció</i> .....	29

<b>Ajánlás.....</b>	<b>29</b>
<b>Regisztrálás.....</b>	<b>30</b>
<b>Bejelentkezés.....</b>	<b>34</b>
<b>A Játéktér.....</b>	<b>37</b>
<b>A játék menete .....</b>	<b>44</b>
<b>Hardware és Software igény .....</b>	<b>45</b>
<b><i>Fejlesztési lehetőségek .....</i></b>	<b><i>46</i></b>
<b>Lövés.....</b>	<b>46</b>
<b>Többjátékosmód .....</b>	<b>46</b>
<b>Pályaválasztás .....</b>	<b>46</b>
<b><i>Összegzés .....</i></b>	<b><i>47</i></b>
<b><i>Irodalomjegyzék.....</i></b>	<b><i>Hiba! A könyvjelző nem létezik.</i></b>

## Bevezetés

A záróvizsgám beadandójának témájához, Comenius Logo és a hasonló egyszerű programozást tanító játékokból vettem ihletet. Azt gondolom fontos a fiatalokkal minél hamarabb megszerettetni a programozás alap gondolkodását, annak érdekében, hogy később hasznos fejlesztő válhasson belőlük.

Tapasztalataim alapján egy diák sokkal hamarabb tanul meg egy magas szintű nyelvet, ha előtte már találkozott egyszerű parancsokkal irányítható játékokkal. Ezen programok invitálók lehetnek kívülállóknak, hiszen megkerülik az igazi programozás esetenként szárazabb technikai részeit.

Nagyon fontos, hogy a programozás alapjait minél hamarabb megismertessük a fiatalokkal, mivel ez egy olyan készség, amelyre számos területen szükség lehet a jövőben. Az egyszerűbb programozást tanító játékok használata jó módszer lehet arra, hogy a diákok szórakoztató és interaktív módon tanulják meg a programozás alapjait. Ezek a játékok segíthetnek a diákoknak megérteni az algoritmikus gondolkodás alapjait és fejleszthetik a problémamegoldó készségeket.

Ezek a játékok valóban megkönnyítik a programozás tanulását és vonzóbbá tehetik a diákok számára, így azt javaslom, hogy használják őket az alapok megtanítására. Azonban fontos, hogy ezek a játékok ne helyettesítsék teljesen a valódi programozási gyakorlatot, hanem inkább kiegészítsék azt. A diákoknak meg kell tanulniuk a nyelv szintaxisát, a változók és függvények használatát, valamint az algoritmusok tervezését. Ha a diákok megértik ezeket az alapelveket, akkor könnyebben áttérhetnek magasabb szintű nyelvekre, hatékonyabb és kreatívabb programokat fejleszthetnek.

## Témaválasztás

A témaválasztásomhoz hozzájárult, hogy korábban tanítóként dolgoztam egy fiatalok számára programozást oktató iskolában. Tanításaim során sok hibát és lehetőséget ismertem meg, és elkezdtem gondolkodni azon, hogy mivel lehetne hatékonyabbá tenni ezt a területet. Szerettem volna olyan megoldást találni, amely lehetővé teszi a diákok számára, hogy élvezetesebbé és izgalmasabbá tegyék a programozás tanulását, és segítsen nekik megérteni a programozás alapjait. Úgy éreztem, hogy egy olyan programozást segítő játék szoftver lehet az ideális megoldás, amely lehetővé teszi a diákok számára, hogy interaktív módon tanulják a programozás alapjait, fejlesszék a kreativitásukat és problémamegoldó képességüket. Ez motiválóbbá és eredményesebbé teheti a tanulási folyamatot, és segíthet a diákoknak megszerezni a programozás alapvető ismereteit.

### A Comedius Logo

A Comenius Logo vagy egyszerűen csak Comlogo egy a Logo programozási nyelven alapuló program, melyet a pozsonyi Comenius Egyetem Informatika Oktatási Tanszékének munkatársai hoztak létre. (Kiadó, 2016)

A logo nyelvet 1967-ben alkotta meg két matematikus és informatikus Wally Feurzeig és Seymour Papert. Ez a nyelv a Lisp programozási nyelv könnyített, egyszerűsített változata. A Logo nyelv használatára többféle program is született, de nemzetközileg is az egyik legismertebb a Comenius Logo, amelyet egy pozsonyi csoport készített. A csoport tagjai mind a Comenius Egyetem Informatika Oktatási Tanszékéhez kötődnek. A csoport tagja volt Ivan Kalas, az egyetem tanszékvezetője, Andrej Blahó és Tomcsányi Péter programozók, emellett segédkezett még Tucsányiné Szabó Márta is. A program gyorsan népszerűvé vált és hamar elterjedt nemcsak az egész országban, hanem több európai országban is, mint Hollandia, Németország, Franciaország, az Egyesült Királyság, Görögország stb. Máig az egyik legnépszerűbb Logo nyelven működő program. (Kiadó, 2016)

A fejlesztők mára már kiadták a továbbfejlesztett, megújított Imagine Logót.

## Oktató jellegű programozási nyelvek története

Az oktató jellegű programozási nyelvek olyan programozási nyelvek, amelyeket az oktatásban alkalmaznak a programozás alapjainak tanítására. Az ilyen nyelvek általában egyszerűbb szintaxisúak és kevesebb funkcióval rendelkeznek, mint a valódi programozási nyelvek, hogy azok könnyebben tanulhatók és könnyebben érthetők legyenek az új programozók számára.

Az első oktatási célra kifejlesztett programozási nyelv az 1960-as években jelent meg, és a BASIC volt. A BASIC nagyon népszerűvé vált az iskolákban és az egyetemeken, mivel könnyen tanulható volt, és lehetővé tette a diákok számára, hogy rövid időn belül valódi programokat írjanak. Az 1970-es években megjelentek más oktatási nyelvek is, például a Pascal, amely szintén nagyon népszerűvé vált. (ELTE IK, VisualBasic, 2010)

Az 1980-as években megjelent az Ada, amelyet az amerikai hadsereg rendelt meg, hogy egységesítse a katonai szoftverfejlesztési folyamatot. Az Ada volt az első olyan nyelv, amelyet a katonai és ipari szektorban egyaránt használtak, és jelentős hatást gyakorolt a későbbi oktatási nyelvekre. (Tamás)

Az 1990-es években jelentek meg az oktatási nyelvek között az Objektum-Orientált Programozás (OOP) nyelvek, mint például a Java és a Python. A Java kifejezetten az oktatási szektor számára lett kifejlesztve, míg a Python általános célú programozási nyelv, de sok diák és oktató használja a programozás alapjainak tanítására. (Róbert)

Az oktatási nyelvek folyamatosan változnak és fejlődnek az idővel, hogy megfeleljenek a diákok és az oktatók igényeinek. A mai napig számos oktatási nyelv áll rendelkezésre, mint például a Scratch, Blockly, és Swift Playgrounds, amelyek a programozás alapjainak tanítására szolgálnak az új programozók számára. (Cash, 2018)

# Fejlesztői dokumentáció

## Adatbázis terv

### Adatbázis célja

Szakedolgozatomban az adatbázis legfontosabb eleme a felhasználó kezelés, a felhasználó adatainak tárolása. Ezek mellett, az adatbázisban tárolom az alábbi dolgokat:

- Felhasználók
- Generált térképek és adataik
- Játék közbeni lépések
- Játékok

### Táblák

Jatekos		
Felhasznalonev	VARCHAR(30)	A játékos által kitalált egyedi felhasználónév, elsődleges kulcs (PK).
Jelszo	BINARY(64)	A játékos jelszava titkosított formátumban eltárolva.
Szin	INT	A játékos tankjának választott színe.

Ezt a táblát az alábbi kóddal hoztam létre:

```
CREATE TABLE Jatekos (  
    Felhasznalonev VARCHAR(30) NOT NULL,  
    Jelszo BINARY(64) NOT NULL,  
    Szin INT NOT NULL  
    PRIMARY KEY (Felhasznalonev)  
);
```

A regisztrálásnál vizsgálom, hogy a felhasználónév foglalt-e, és csak abban az esetben engedélyezem és viszem fel az adatbázisba amennyiben nem létezik ilyen felhasználónév.

Terkep		
TerkepID	INT	A térkép ID-je automatikusan generált azonosító (AI, PK)
Nev	VARCHAR(30)	A térkép neve, automatikusan generált a program által (fejleszthetőségi lehetőségként el lehetett volna nevezni)



TerkepMap	VARCHAR(120)	Maga a térkép betűkkel ábrázolva.
Pont	INT	A térkép értékelése, ahol 0 a nem értékelt, 5 a nagyon jóra értékelt. (fejlesztési lehetőség)

Ezt a táblát az alábbi kóddal hoztam létre:

```
CREATE TABLE Terkep (
    TerkepID INT NOT NULL IDENTITY,
    Nev VARCHAR(120),
    TerkepMap INT,
    Pont INT,
    PRIMARY KEY (TerkepID)
);
```

Lepesek		
LepesID	INT	Automatikusan generált ID érték (AI, PK).
TerkepID	INT	Az éppen generált térkép ID-je (FK).
Felhasznalonev	VARCHAR(30)	Az éppen játszó játékos felhasználónéve (FK).
Lepes	VARCHAR(120)	Az épp lefuttatott lépés.

Ezt a táblát az alábbi kóddal hoztam létre:

```
CREATE TABLE Lepesek (
    LepesID INT NOT NULL IDENTITY,
    TerkepID INT NOT NULL,
    Felhasznalonev VARCHAR(30) NOT NULL,
    Lepas VARCHAR(120) NOT NULL
    PRIMARY KEY (LepesID)
);
```

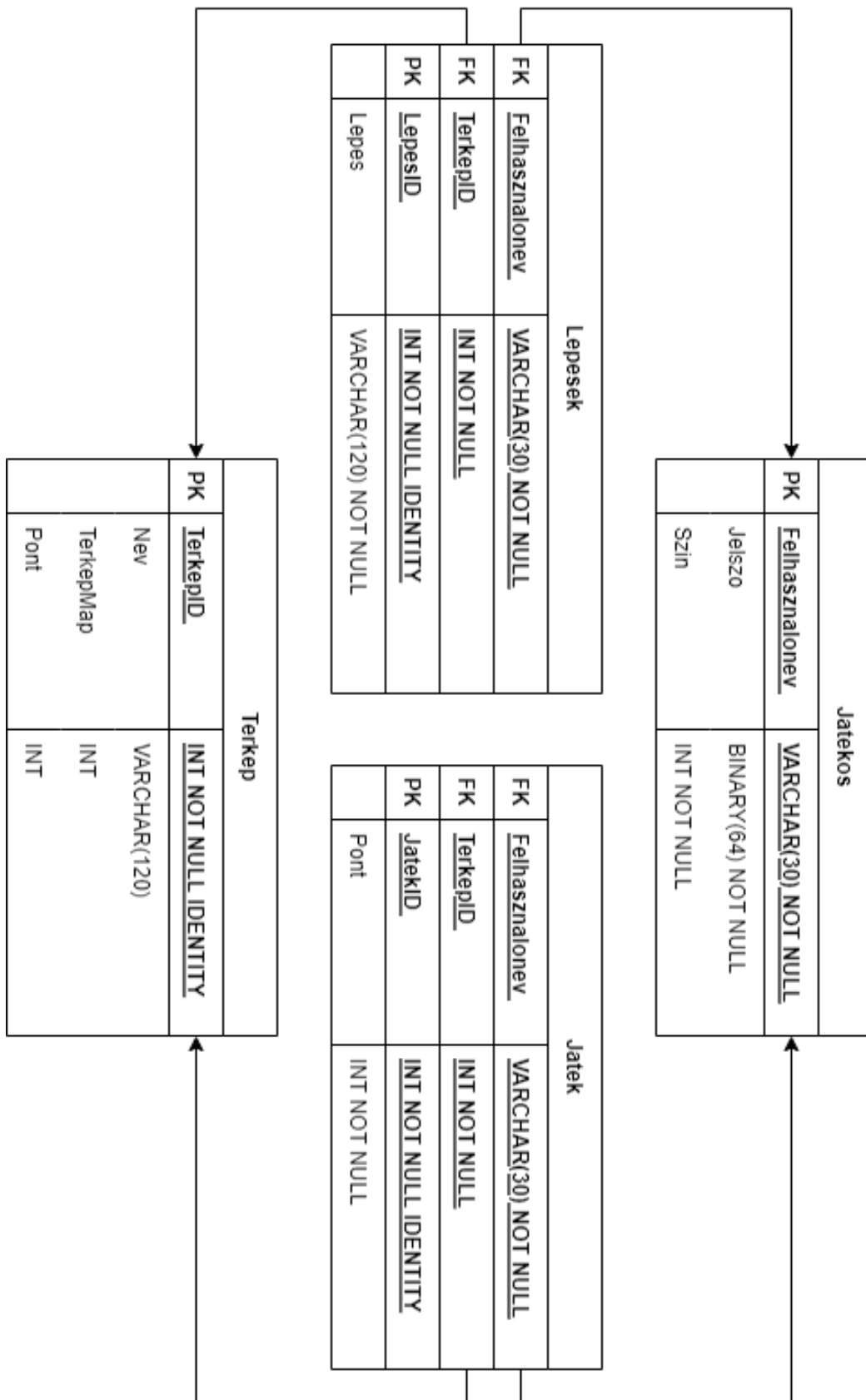
Jatek		
JatekID	INT	Automatikusan generált ID érték (AI, PK).
TerkepID	INT	Annak a térképnek az ID-je, ahol a kör játszódott (FK).
Felhasznalonev	VARCHAR(30)	Az játszó játékos felhasználónéve (FK).
Pont	INT	A játékos által elért pontszám.

Ezt a táblát az alábbi kóddal hoztam létre:

```
CREATE TABLE Jatek (
    JatekID INT NOT NULL IDENTITY,
    TerkepID INT NOT NULL,
    Felhasznalonev VARCHAR(30) NOT NULL,
    Pont INT NOT NULL,
    PRIMARY KEY (JatekID)
);
```

Az elsődleges kulcsokat PK-val, az idegen kulcsokat FK-val jelöltem.

Az adattáblák közti kapcsolatokat az alábbi relációs ábra mutatja:



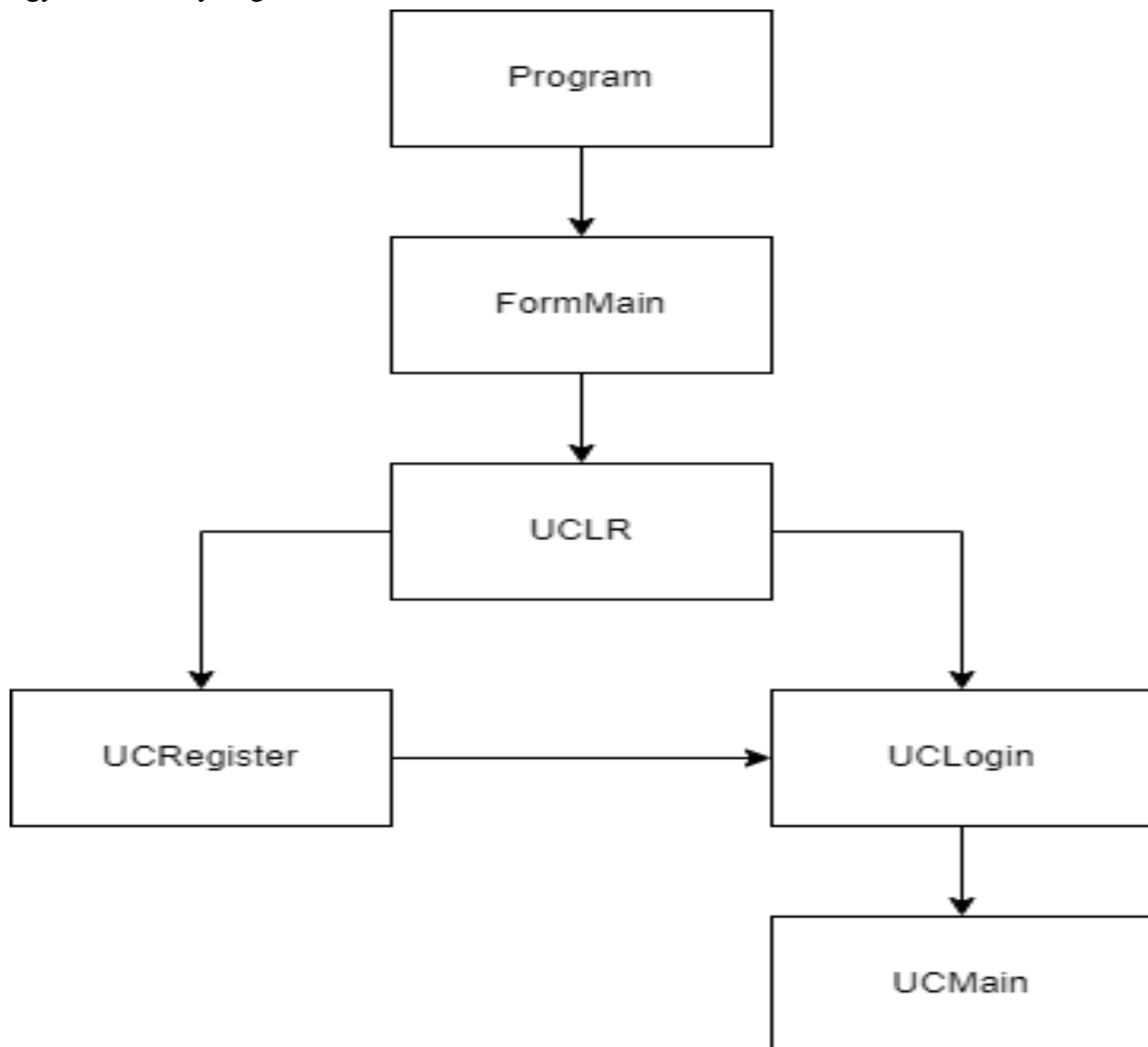
1. ábra Adatbázis UML ábra

## Osztályok és metódusok

A program osztályai:

Név	Leírás
FormMain	A fő ablak, ezen belül történik minden.
Program	Vele indul a program, ő hívja meg a FormMain-t-
UCLR	Meghívja az UCRregistert, vagy az UCLogint
UCRegister	Sikeres regisztráció esetén, meghívja az UCLogin metódust.
UCLogin	Megfelelően beírt adatok után meghívja az UCMaint.
UCMain	Az UCLogin által megöröklöt felhasználó adataival új játékot indít.

Ugyanez Osztálydiagrammon:



2. ábra Osztálydiagramm

Program osztály:

■

3. ábra Main Program Class elemei

Metódusok			
Név	Bemenet	Kimenet	Leírás
Main			Konstruktor

FormMain:

□

4. ábra FormMain elemei

Változók	
Név	Leírás
LogReg	UCLR típusú változó, akit kiteszünk kontrollra.

Metódusok			
Név	Bemenet	Kimenet	Leírás
FormMain			Innen indítható a kezdő kontroll.

UCLR:

n

5. ábra Login-Registerl elemei

Változók	
Név	Leírás
btnLogin	Belépésre szolgáló gomb.
btnReg	Regisztrálásra szolgáló gomb.

Metódusok			
Név	Bemenet	Kimenet	Leírás
UCLR			A vizuális elemek megjelenítésére szolgál.
btnLogin_Click	küldő elem		Eltünteti ezt a kontrollt és az UCLogot jeleníti meg helyette.
btnReg_Click	küldő elem		Eltünteti ezt a kontrollt és az UCRegistert jeleníti meg helyette.

UCRegister:

0

6. ábra Register elemei

Változók	
Név	Leírás
btnBackR	Előző UC-re visszaléptető gomb.
btnRegR	Regisztrálásra szolgáló gomb.
lblColor	Színválasztót feltüntető szöveg.
lblRPass	Jelszó helyét feltüntető szöveg.
lblRPassA	Második jelszót feltüntető szöveg.
lblUser	Felhasználónevet feltüntető szöveg.
pnlColorPic	Színválasztó panel.
rBBlue	Kék színt jelölő RadioButton.
rBRed	Piros színt jelölő RadioButton.
rBYellow	Sárga színt jelölő RadioButton.
tbRPass	Jelszót kezelő TextBox.
tbRPassA	Második Jelszót kezelő TextBox.
tBUser	Felhasználónevet kezelő TextBox.



Metódusok			
Név	Bemenet	Kimenet	Leírás
UCRegister			Megjeleníti a vizuális elemeket.
btnBackR_Click	küldő elem		Bezárja az aktuális UserControllt és megjeleníti az előzőt.

btnRegR\_Click

Egyetlen bemeneti paramétere a küldő elem. Ez a metódus először megvizsgálja, hogy a beírt felhasználónév foglalt-e, ezt az alábbi lekérdezéssel teszi meg:

```
"SELECT COUNT(*) FROM Jatekos WHERE Felhasznalonev = @Username"
```

Ahol a „@Username” helyére a tBUser-be beírt szöveget keresi az adatbázisban, amennyiben talál legalább egyet, az azt jelenti, hogy a felhasználónév nem egyedi, így erre figyelmezteti a felhasználót egy „MessageBox.Show()” paranccsal. Amennyiben a felhasználónév egyedi, a két jelszót vizsgálja, hogy egyeznek-e, amennyiben ez így van átfordítja a választott színt egy számmá egytől háromig, ahol az egy az piros, a kettő az kék, a három pedig sárga. Ellenkező esetben ismét a „MessageBox.Show()” paranccsal figyelmezteti a felhasználót, hogy hibásak az adatok. Ezután az SHA256 osztály használatával titkosítja a jelszavát, ami egy 256 bit hosszú hash kóddá alakítja át (ebből a kódból a jelszó nem alakítható vissza és sokkal biztonságosabb ezt tárolni az adatbázisban). Miután a jelszót titkosította, felveszi az adatbázisba, a felhasználónévvel, illetve a választott szín számával együtt:

```
"INSERT INTO Jatekos (Felhasznalonev, Jelszo, Szin) VALUES (@Username, @Password, @Color)"
```

Ezután utoljára egy „MessageBox.Show()” paranccsal tudatja a felhasználóval a sikeres regisztrációt, bezárja ezt az UserControllt és nyit egy UCLogint, ahol a frissen regisztrált felhasználó beléphet.

UCLLogin:

☐

7. ábra Login elemei

Változók	
Név	Leírás
btnBackL	Előző UC-re visszaléptető gomb.
btnLoginL	Bejelentkezésre szolgáló gomb.
lblLPass	Jelszó helyét feltüntető szöveg.
lblUser	Felhasználónevet feltüntető szöveg.
tBLPass	Jelszót kezelő TextBox.
tBLUser	Felhasználónevet kezelő TextBox.

Metódusok			
Név	Bemenet	Kimenet	Leírás
UCLLogin			Megjeleníti a vizuális elemeket.
btnBackL_Click	küldő elem		Bezárja az aktuális UserControllt és megjeleníti az előzőt.

btnLoginL\_Click

Egyetlen bemeneti paramétere a küldő elem. Ez a metódus ismét kiveszi a tBLPass-ból a jelszót, amit rögtön hash formátumba alakít hiszen előzőleg is ebben tároltuk az adatbázisban, erre ugyanazt a SHA256 osztályt használom. Ennek használatával a program és az adatbázis között sosem kell nyers jelszót küldeni, ez nagyon hasznos a biztonság szempontjából. Ezután kivesszük a szöveget a tBLUser-ből és ezzel megvan az összes szükséges adatunk a belépéshez így egy egyszerű lekérdezéssel le lehet ellenőrizni, hogy megfelelő adatokat kaptuk-e: **"SELECT \* FROM Jatekos WHERE Felhasznalonev = @Username AND Jelszo = @Password"**, a lekérdezés ellenőrzi, hogy létezik-e adatsor a megadott felhasználónévvel és jelszóval. Ha a felhasználó rossz adatokat adott meg, felhívjuk a figyelmét a „MessageBox.Show()” parancssal, hogy adatai hibásak, ellenkező esetben amennyiben a két adat megtalálható az adatbázisban, bezárjuk az aktuális login kontrollt és nyitunk egy UCMain kontrollt, a már megvizsgált felhasználónevet tovább küldi neki.

UCMain:

7

8

*8. ábra UCMain metódusai*

*9. ábra UCMain mezői*

Változók	
Név	Leírás
UserName	Itt tárolom a megöröklött felhasználónevet.
picBoxT	Ebben a kétdimenziós tömbben tárolom a megjelenítéshez szükséges PictureBoxokat.
TankImage	Itt tárolom a tank képét, alapvetően piros, ha a lekérdezés nem sikerülne.
x	Az éppen rajzolt elem y koordinátája.
y	Az éppen rajzolt elem x koordinátája.
oldx	Az előző elem x koordinátája a felesleges elemek eltisztításához.
oldy	Az előző elem y koordinátája a felesleges elemek eltisztításához.
mapID	A térkép eltárolt ID je későbbi pont frissítéshez.
partCount	A már összeszedett pontokat számolja.
movesLeft	A meglévő lépéseket tartja számon, húsz lépéssel kezd.
partsToGet	A generált összeszedhető elemeket tartja számon.
Diection	Az égtájakhoz egy adatszerkezet, amiből a directions származik.
directions	Directions típusa változó, kezdéskor North.
btnRunCode	Ez a gomb indítja el a kód ellenőrzését aztán futtatását.
fLPMain	Esztétikai pozicionálásra szolgáló elem.
lblCodeHeader	A kód mező fejléce.
lblError	Esetleges hibák megjelenítésére szolgáló szöveg.
lblMoves	Hátralévő lépéseket mutató szöveg.
lblPoints	Megszerzett pontokat mutató szöveg.
tbCode	Ebbe a szövegdobozba lehet begépelni a lépéseket.
tLP	Esztétikai pozicionálásra szolgáló elem.
tLPMain	Fő esztétikai pozicionálásra szolgáló elem.
tLPStats	Esztétikai pozicionálására szolgáló elem.

Metódusok			
Név	Bemenet	Kimenet	Leírás
UCMain			Megjeleníti a vizuális elemeket.

#### ResetWorld

Ez a metódus törli az éppen megjelenített Main userkontrollt, és megjelenít egy újat, ezzel az egészet alaphelyzetbe állítja. A metódusba-szervezés célja, hogy bármikor visszaállíthatom a program-világot az eredeti állapotba.

setTankColor

Ebben a metódusban az alábbi sorral lekérdezzük, hogy az épen belépett játékos milyen színű tankot választott regisztrálásnál:

```
"SELECT Szin FROM Jatekos WHERE Felhasznalonev = @Username"
```

Ezért van jelentősége átadni a belépett felhasználó nevét ennek az osztálynak. Ezzel egy számot kapunk egytől háromig, ha egy a tank piros, ha kettő a tank az kék, ha pedig három a tank sárga. Ennek megfelelően a Resourcesbe elmentett képet hozzá rendeljük a „TankImage”-hez.

GenerateMap

Ez a metódus fut le először, rögtön a megjelenített UI elemek után. Ebben a metódusban készítünk először egy hatvannégy karakter hosszú char tömböt, ami a világ nyolcszor-nyolcas térképét tárolja majd és indítok egy hatvannégyszer lefutó számláló ciklust. Ebben a ciklusban először inicializálok egy PictureBox típusú ideiglenes változót, kikapcsolom a kitöltést, illetve elnevezem "picBox" + i-nek, ahol az i a ciklusszámláló aktuális értéke. Ezután beállítom a kép méretét negyvennyolcszor-negyvennyolc pixelre, és végül kitörlöm az esetlegesen automatikusan generált szöveget. Ezek után adok értékeket a képnek, az első elem mindenképp zöld, illetve a háttérben tárolt eleme a „G”, mint Green. Ezután, ha már nem az első elemnél tart a ciklus belépünk egy több kimeneteles elágazásba, itt derül ki egy véletlenszám-generálással, hogy az adott elem:

- Az akna lesz, amit a piros szín jelöl a háttérben „R” betűvel, mint Red (erre húsz százalék esély van).
- Fal pedig fekete lesz, jelölése „B”, mint Black (erre szintén húsz százalék esély van).
- Az egyik különleges elem a “pont”, ezt a sárga szín jelöli, ami a háttérben „Y” lesz, mint Yellow.
- Végül pedig az egyszerű “talaj”, amivel kezdtünk is (erre ötven százalék az esély), háttérbeli jelölése „G”, mint Green.

Az elágazások után a random generált képet betesszük az erre létrehozott picBoxT kétdimenziós tömbbe, hogy később ki tudjuk rajzolni azt. A ciklus végén segítünk a ciklusnak, hogy soronként lépkedjen. Végül pedig feltesszük a kontroll listára a frissen generált képet, hogy látható legyen.

A cikluson kívül átalakítjuk az eddig feltöltött char tömbünket string formátumba, amit utána feltöltünk az adott pontozással az adatbázisba az alábbi kód segítségével:

```
"INSERT INTO Terkep (Nev, TerkepMap, Pont) VALUES (@Nev, @TerkepMap, @Pont)"
```

btnRunCodeClick

Ez a leghosszabb metódus, és arra szolgál, hogy beveszi az adatot a kód számára fenttartott szöveg dobozból, és tördeli azt. Ezt úgy teszi, hogy először betölti egy stringbe, amit utána a „`Split(';',')`” függvény segítségével minden „;” mentén feldarabol és egy string típusú tömbbe helyezi. Így már egyszerűen egy „`foreach`” ciklus segítségével végig léphetünk az összes beírt kódon. Mikor belépünk a ciklusba először a „`Trim()`” használatával letömörítem a kódsort, hogy ne kelljen foglalkozni a felesleges kis és nagybetűkkel. A konkrét azonosításra egy számomra új dolgot, a „`Regex`”-et használtam, ez az osztály abban segít, hogy egy tömör formátumban megadhatom neki milyen egyezést szeretnék keresni. Jelen esetben a „`\"([^(]+)\"-t`”, ez lebontva a következőket jelenti:

- “(” : nyitó zárójel karakter;
- “([^(]+)”: bármilyen karakterláncot elfogad a zárójelen belül, amely nem tartalmaz záró zárójel karaktert. Az “([^(]+)” rész jelenti ezt, ahol:
  - “[^]”: bármilyen karaktert elfogad, kivéve a záró zárójel karaktert;
  - “+”: a megadott karakterláncot legalább egyszer, de akár többször is megismétli;
  - “)” : záró zárójel karakter.

Amennyiben egyezést talál, eltárolja azt egy Match típusú objektumba. Ha talált egyezést a „`match.Success`” értéke igaz lesz, így ezt vizsgálva belépünk egy elágazásba. Ezen belül kivesszük a Matchből a konkrét beírt értéket és eltároljuk egy integer típusú változóban. Így megvan az összes szükséges adat, nyitunk egy többirányú elágazást, ahol az adott kód szöveges részét vizsgáljuk.

Ez négy felismerhető kód lehet:

- **"fel"**: ebben az esetben először azt vizsgáljuk, hogy a játékos adott pozíciójától egygel fentebb lévő elem nem a fal, és nem is blokádnak, ebben az esetben az y pozíciót csökkentem, illetve a tankot Északi irányba állítom. Ellenkező esetben „`MessageBox.Show`” segítségével felhívom a figyelmet, hogy arra tovább nem tud közlekedni.
- **"le"**: ebben az esetben először azt vizsgáljuk, hogy a játékos adott pozíciójától egygel lentebb lévő elem nem a fal, és nem is blokádnak, ebben az esetben az y pozíciót növelem, illetve a tankot Déli irányba állítom. Ellenkező esetben „`MessageBox.Show`” segítségével felhívom a figyelmet, hogy arra tovább nem tud közlekedni.
- **"jobbra"**: ebben az esetben először azt vizsgáljuk, hogy a játékos adott pozíciójától egygel jobbra lévő elem nem a fal, és nem is blokádnak, ebben az esetben az x pozíciót növelem, illetve a tankot Keleti irányba állítom. Ellenkező esetben „`MessageBox.Show`” segítségével felhívom a figyelmet, hogy arra tovább nem tud közlekedni.
- **"balra"**: ebben az esetben először azt vizsgáljuk, hogy a játékos adott pozíciójától egygel balra lévő elem nem a fal, és nem is blokádnak, ebben az esetben az x pozíciót csökkentem, illetve a tankot Keleti irányba állítom. Ellenkező esetben „`MessageBox.Show`” segítségével felhívom a figyelmet, hogy arra tovább nem tud közlekedni.
- Végül amennyiben ezek közül egyik sem teljesül a „`MessageBox.Show`” paranccsal kiírom, hogy a parancs nem megfelelő, illetve kiírom a parancsot is.

Ez után az elágazás után felrajzolom a tankot a térkép megfelelő helyére, erre a „`DrawTank(x, y, direction)`” parancsot használom. A parancsok vizsgálása után ellenőrzöm a pontokat, elsősorban, hogy van-e még összeszedhető alkatrész. Amennyiben minden alkatrész elfogyott a játékos győzött, és ez ismét egy „`MessageBox.Show`” tudatja vele, illetve új világot generál neki, hogy tovább játszhaszon a „`ResetWorld()`” metódussal. Ha ez az elágazás nem fut le, ellenőrzi, hogy maradt-e még lépése. Amennyiben a lépése elfogyott, de van még összeszedendő elem, a játékos veszít, és egy „`MessageBox.Show`” tudatja ezt vele, illetve felhívja a figyelmet, hogy hány alkatrész maradt. Végül pedig feltölti a lépést az adatbázisba a „`LogMoveToDB()`” metódus segítségével.



DrawTank

Ez a metódus három értéket kap:

- egy vízszintes pozíciót, amit az „x” jelöl;
- egy függőleges pozíciót, amit az „y” jelöl;
- egy irányt, amit a „dir” jelöl.

Létrehozunk egy képet, ami megkapja az aktuális tank képét, utána egy többirányú elágazás segítségével ellenőrzi a kapott Directiont, és aszerint forgatja el az alap képet, hogy a tank mindig arra nézzen amerre megy. Most lesz fontos a y, x és a régi x, y koordináta. Először letöröljük az elő pozícióról a tankot „picBoxT[oldx, oldy]” és megjelenítjük a már megfelelően elforgatott képet az új koordinátákra „picBoxT[x, y]”.

Ezután jön az ellenőrzés, hogy éppen mire lépett a játékos. Amennyiben:

- „picBoxT[x, y].BackColor == Color.Yellow”, azaz amire lépett az sárga, alkatrészt talált. Ebben az esetben csökkentjük a megtalálható értékek mennyiségét „partsToGet--” és növeljük a talált alkatrészeket „partCount++” és frissítjük a pontszámát, hogy az új pontszámot jelenítse meg, végül kitörli a sárga négyzetet, vagyis a valóságban átszínezi zöldre „picBoxT[x, y].BackColor = Color.Green”.
- „picBoxT[x, y].BackColor == Color.Red”, azaz, amire lépett az piros, tehát akna. Ebben az esetben nullázódik a megtalált alkatrészek mennyisége, a játékos elveszít mindent, szintén frissül a pont kiíró szöveg, és itt is a háttérrel zöldre állítjuk ezzel eltüntetve az aknát.

Az elágazás után eltároljuk a mostani pozíciót az előző helyére, hogy a következő lépésnél tudjuk mit kell kitörölni, levonunk egyet a lépésekből és frissítjük a lépéseket kiíró szöveget „lblMoves.Text = **Lépések:**” + movesLeft”

Tesztelés

Elsősorban én teszteltem annak érdekében, hogy minden funkciója megfelelően működjön, és a játékmenet élvezhető legyen. Ezenkívül a szakmában dolgozó személyek is tesztelték a programomat, akik értékes visszajelzéseket adtak a játékmenet hatékonyságáról és a felhasználói felület használhatóságáról. Továbbá megfelelő korú diákok is tesztelték a játékot, akik véleményüket osztották meg arról, hogy a játék szórakoztató-e, és érdekes-e számukra. Az általuk nyújtott visszajelzések alapján finomítottam a programomat és javítottam a

használhatóságot és a felhasználói élményt. Az ilyen tesztek segítettek abban, hogy a program megfelelően működjön és eredményesen szolgálja a felhasználók igényeit.

Továbbá egységteszteket végeztem és igyekeztem a lehető legtöbb előfordulást vizsgálni. A talált problémákat javítottam. Igyekeztem minél több lehetőséget bejárni minél több értékkel. A fejlesztés közben igyekeztem a szemantikai hibákat megtalálni és javítani, de ebben a Visual Studio nagy segítség volt. Próbáltam a duplikációkat elkerülni.

Észlelt fő problémák:

- A form ablakába nem fért bele az összes user controll, de ezt egy standard mérettel megoldottam.
- A játékos kisétált a térképről és ezáltal kifutott a tömb, ezt egy egyszerű vizsgálattal javítottam, és ezzel egy funkciót is hozzá adtam.

## Fejlesztési és Megvalósítási Ütemterv

### Fejlesztői környezet kialakítása

A Visual Studio .NET Forms fejlesztői környezet használata során szükséges telepíteni a Visual Studio-t, amely a Microsoft hivatalos oldaláról tölthető le. A Windows Forms alkalmazások fejlesztéséhez a .NET Framework telepítése is szükséges, amely szintén elérhető a Microsoft oldalán. Az adatbázis-kezelő rendszer használata esetén további szoftverek telepítése szükséges az SQL Server Management Studio. A szükséges szoftverek telepítéséhez ajánlott az aktuális verziók használata, és a telepítési útmutatók alapos betartása.

2022. december 3.

### Adatbázis tervezés

Az adatbázis tervezésénél elsősorban a felhasználó kezelést beléptetést vettem figyelembe. Emellett pont, lépés, illetve térkép loggoló rendszert is tartalmaz a program, így ezekre a funkciókra is gondolnom kellett. A táblák elsődleges primitív ábrázolására a draw.io oldalt használtam, ahol átlátható UML diagramokkal tudtam követni a munkámat. Az adatbázisok létrehozására a TSQL-t használtam.

2022. január 31.

## Frontend

Először egyszerűen papíron lerajzoltam az elképzeléseimet, utána egy dumpy projektben teszteltem az elrendezést. Miután létrehoztam a projektet, elkezdtem dolgozni a felhasználói felületen. A Visual Studio .NET Forms lehetővé teszi a vizuális felületkészítést, amely nagyban felgyorsítja a fejlesztést. Az alkalmazás felületének kialakítása során számos vezérlőt használtam, például gombokat, szövegdobozokat, címkéket, listákat stb. A felhasználói felület kialakítása során a Visual Studio lehetőséget biztosít a drag and drop módszer használatára is, ami még egyszerűbbé teszi a felületkészítést.

2023. február 2. – február 28.

## Backend

Ezt egy egyszerű regisztráló és beléptető rendszerrel kezdtem, amit rögtön az adatbázishoz tudtam csatolni. A jelszót titkosítottam és nem engedtem duplikált felhasználóneveket, hiszen abban a táblában az a kulcsmező. Miután ezzel végeztem, nekiálltam a térkép generálásnak, itt igyekeztem nem túl sok és nem túl kevés elemet integrálni az átlátható, de izgalmas játékelmény érdekében. Miután megvolt a térkép, amit a már meglévő táblába tárolni tudtam, nekiállhattam a kód parsolásnak. Ez egy keményebb feladat volt, megismertem a különböző parsolási technológiákat, de végül a Regexet választottam. Először nehezemre esett megérteni a szintaktikáját, de miután elsajátítottam könnyedén implementáltam a programba. A parsolt parancsok hatásait már könnyű volt elkészíteni, hiszen csak a pozíciót kellett mozgatni, amit a kirajzolás nagyon jól kezel. Végül a lépések és a játékok loggolása a számukra létrehozott táblákba szinte gyerekjáték volt.

2023. február 25. – április 2.

## Tesztelés

Nem csak én teszteltem, hanem diákok, szakmabeliek és külső szemlélők is tesztelték annak érdekében, hogy minden funkciója megfelelően működjön. Az általuk visszajelzett információk alapján finomíthattam a programomat, és lehetővé vált a használhatóság és a felhasználói élmény további javítása is. Az ilyen tesztek segítenek abban, hogy a program megfelelően működjön és eredményesen szolgálja a felhasználók igényeit.

2023. március 2. – április 9.

## Fejlesztői környezet

### Szoftverek

- draw.io tervezés vizualizálás, ábrák elkészítése
- Visual Studio Community 2022
  - .NET Windows Forms Application
  - SQL Server Manager
- Paint.net
- Microsoft Word
- Microsoft Excel
- Windows 11 Pro
- Windows 10

### Hardverek

- Otthoni asztali számítógép: AMD Ryzen 5 3600 6-Core Processor 3.60 GHz, 16 GB Ram
- Fujitsu laptop
- Iskolai gépek

## Felhasználói dokumentáció

A program célja egy tank irányítása parancssorral, ahol véletlen generált pályákon kell végig navigálni a háború törmelékét kikerülve és ezzel minél több pontot elérni.

### Ajánlás

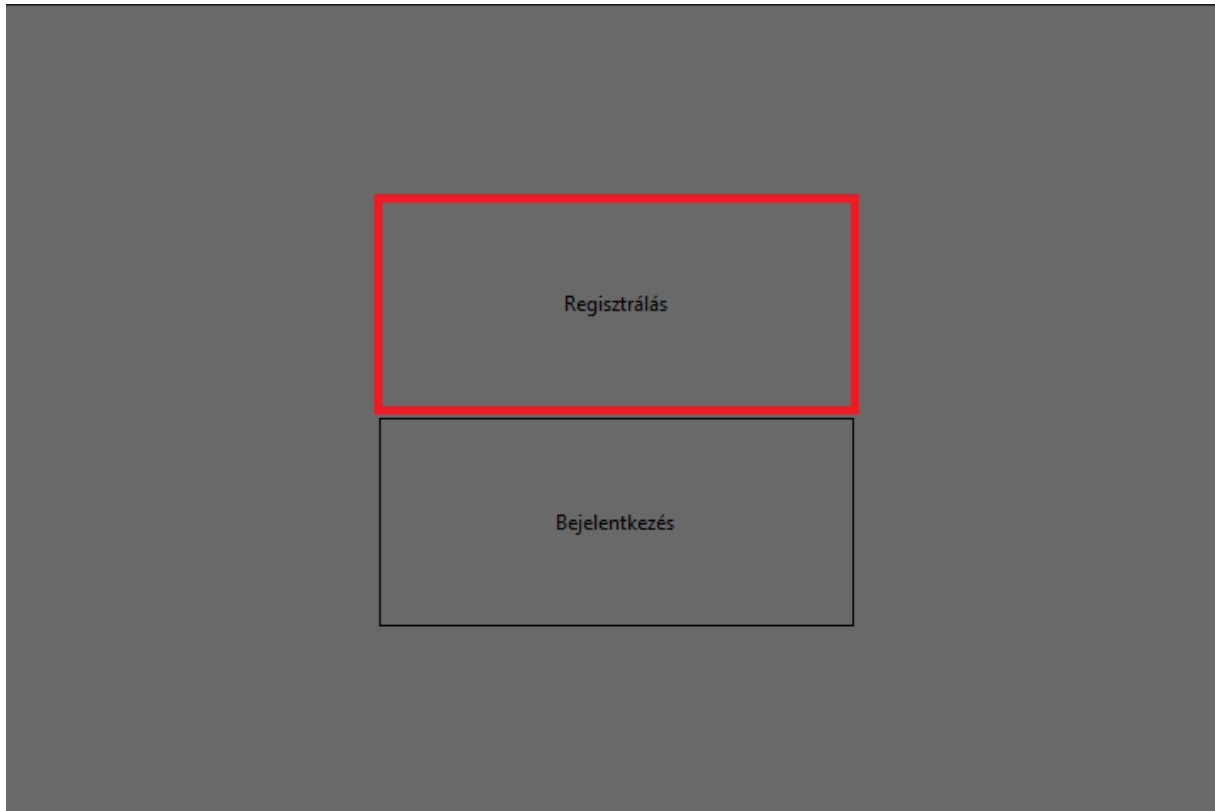
Ezt a programot hét-tizennégy éves korosztálynak ajánlom, ugyanis a program írás és olvasás tudását igényli, illetve a programban megjelenített tank a fiatalabb korosztály számára felkavaró lehet, így számukra a program használatát nem tudom ajánlani. Emellett a program a fiúk számára lehet kedvezőbb, de ez nem zárja ki, hogy lányok is játszhasassak vele. A programot egyszerre egy felhasználó tudja használni, de a pontok összehasonlításával egy többjátékos élménnyé is kinőheti magát a program. A játék fejleszti elsősorban a térbeli gondolkodást a gyermekeknek, emellett egy nagyon jó módja a lineáris gondolkodás elsajátításának. A lineáris modellezés gondolkodás azt jelenti, hogy egy problémát lépésről lépésre az egyik lépést a másikkra próbáljuk megoldani, illetve ahogy a programban parancsokat formázunk a tank mozgatásához az tökéletesen elsajátíttatja a gyermekekkel ezt a fajta gondolkozásmódot.

Lehetőségeink a programban:

- Regisztrálás
- Bejelentkezés
- Tank színének testre szabása
- Világ generálása
- A tank irányítása különböző parancsokkal

## Regisztrálás

A program használatához szükséges regisztrálni ezt a program indítása után a „Regisztrálás” gombra kattintva teheti meg.

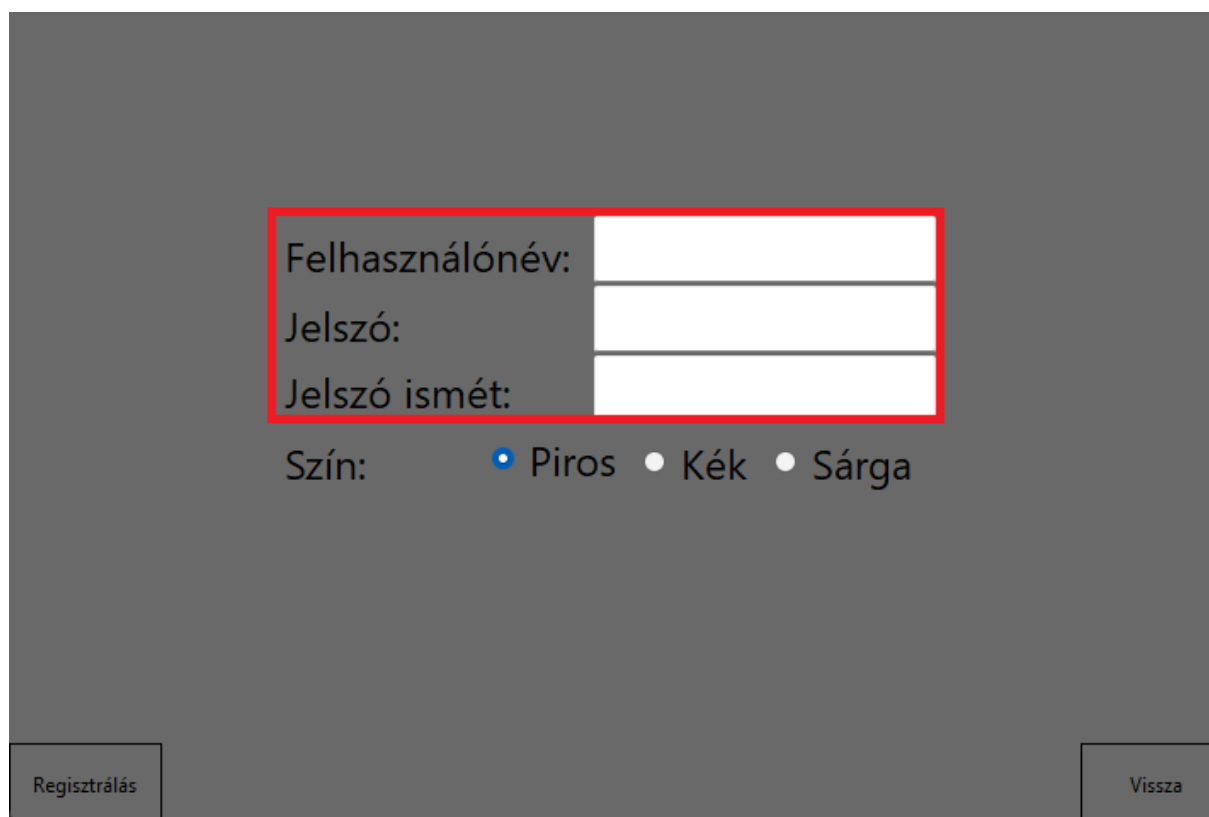


*10. ábra Regisztrálás gomb*

Ezután ki kell találnunk egy tetszőleges felhasználó nevet, egy erős jelszót, ami legalább nyolc karakter, tartalmaz kis és nagybetűt és számot. Ügyeljünk arra, hogy a két jelszó teljes

mértékben

megegyezzen!



The image shows a registration form on a dark gray background. A red rectangular box highlights the input fields for 'Felhasználónév:', 'Jelszó:', and 'Jelszó ismét:'. Below these fields is a color selection section labeled 'Szín:' with three radio buttons: 'Piros' (selected), 'Kék', and 'Sárga'. At the bottom left is a 'Regisztrálás' button, and at the bottom right is a 'Vissza' button.

Felhasználónév:

Jelszó:

Jelszó ismét:

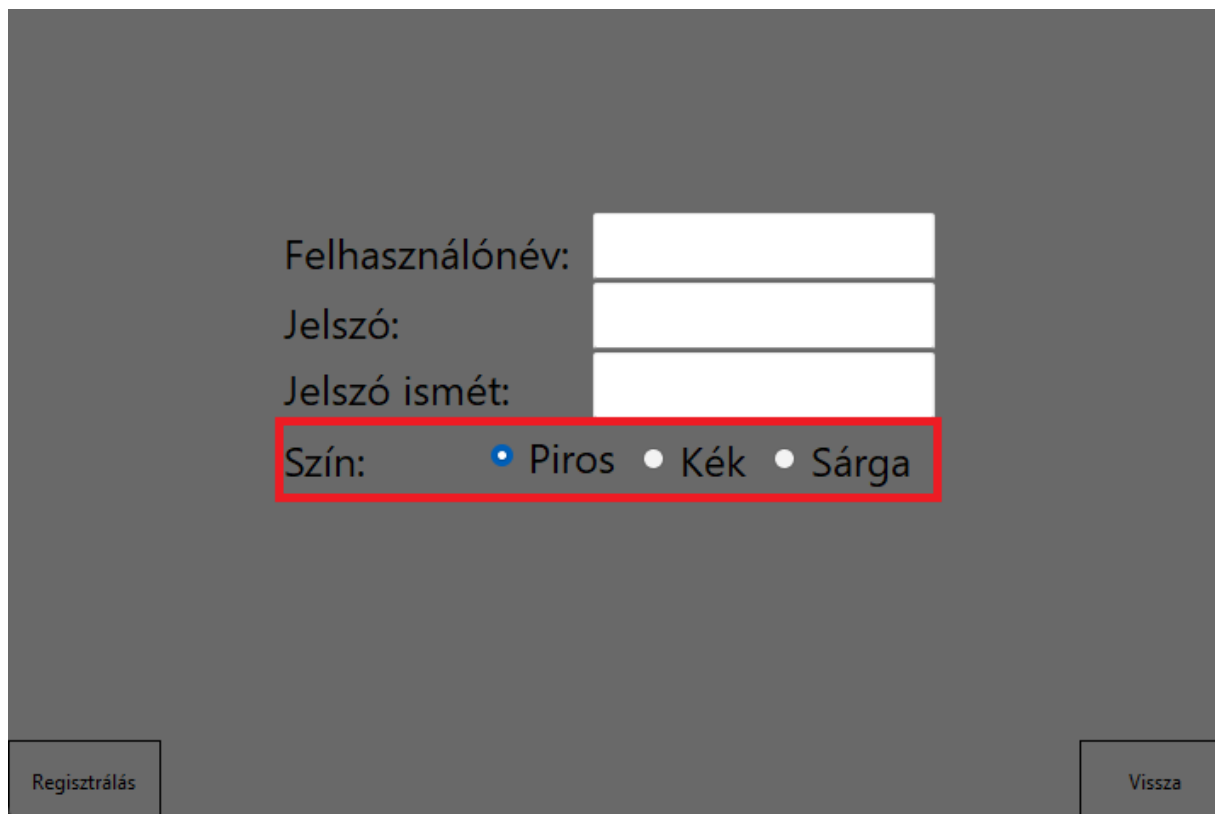
Szín: ☒ Piros ☐ Kék ☐ Sárga

Regisztrálás

Vissza

11. ábra Regisztráláshoz szükséges mezők

Ezek kitöltése után lehet tetszőlegesen színt választani a megadott lehetőségek közül.



The image shows a registration form on a dark gray background. It contains four input fields stacked vertically: 'Felhasználónév:', 'Jelszó:', 'Jelszó ismét:', and 'Szín:'. The first three fields are white text boxes. The 'Szín:' field is a radio button group with three options: 'Piros' (selected with a blue dot), 'Kék', and 'Sárga'. A red rectangular box highlights the 'Szín:' field and its options. At the bottom left is a button labeled 'Regisztrálás' and at the bottom right is a button labeled 'Vissza'.

Felhasználónév:

Jelszó:

Jelszó ismét:

Szín: ☒ Piros ☐ Kék ☐ Sárga

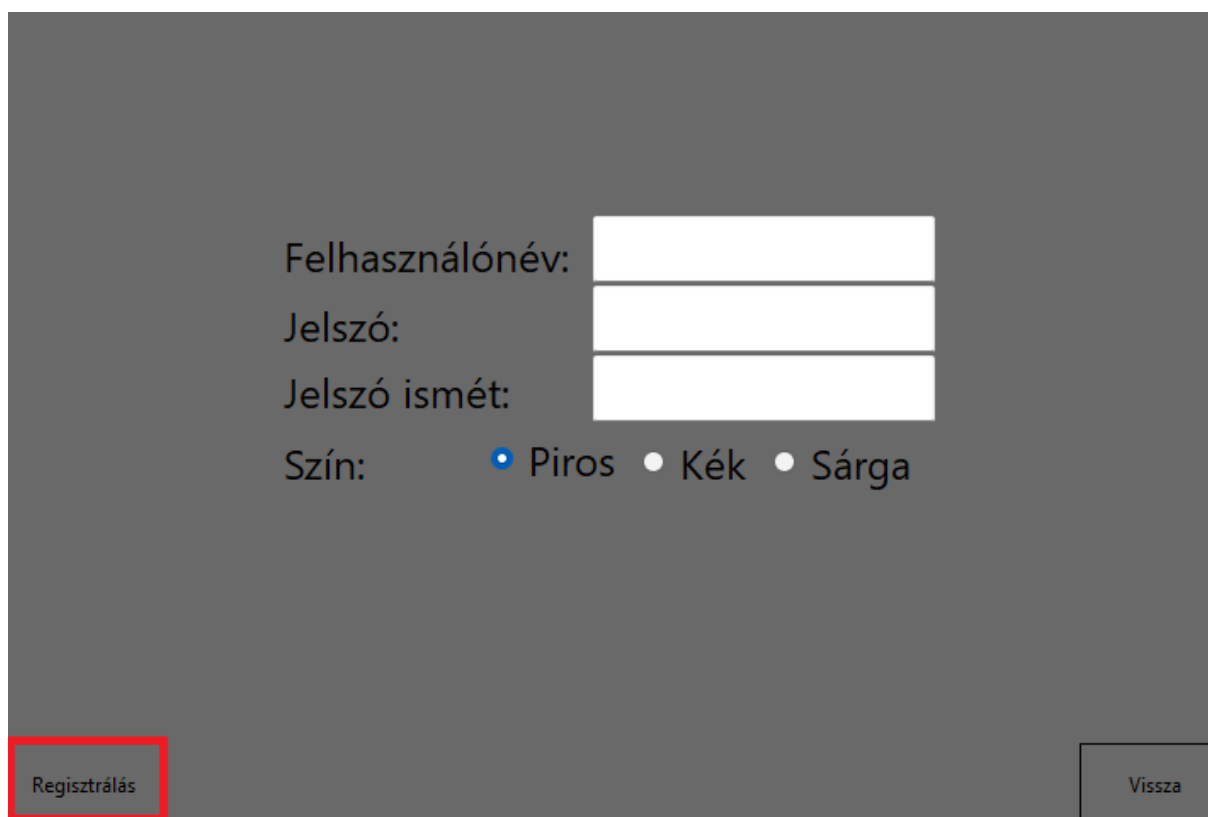
Regisztrálás

Vissza

12. ábra Színválasztás



Ha minden adatunkat leellenőriztük a „Regisztrálás” gombbal véglegesíthetjük a regisztrálást.



Felhasználónév:

Jelszó:

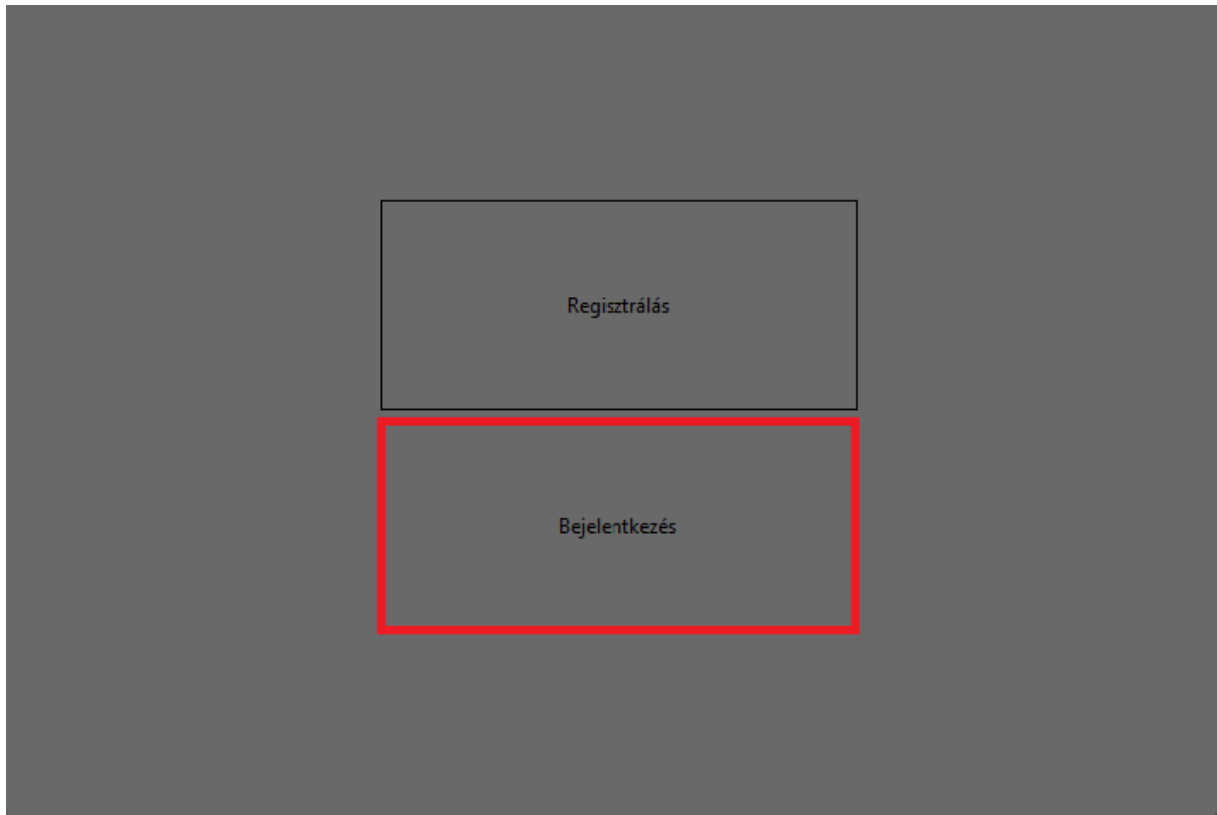
Jelszó ismét:

Szín: ☒ Piros ☐ Kék ☐ Sárga

13. ábra Regisztrálás véglegesítése

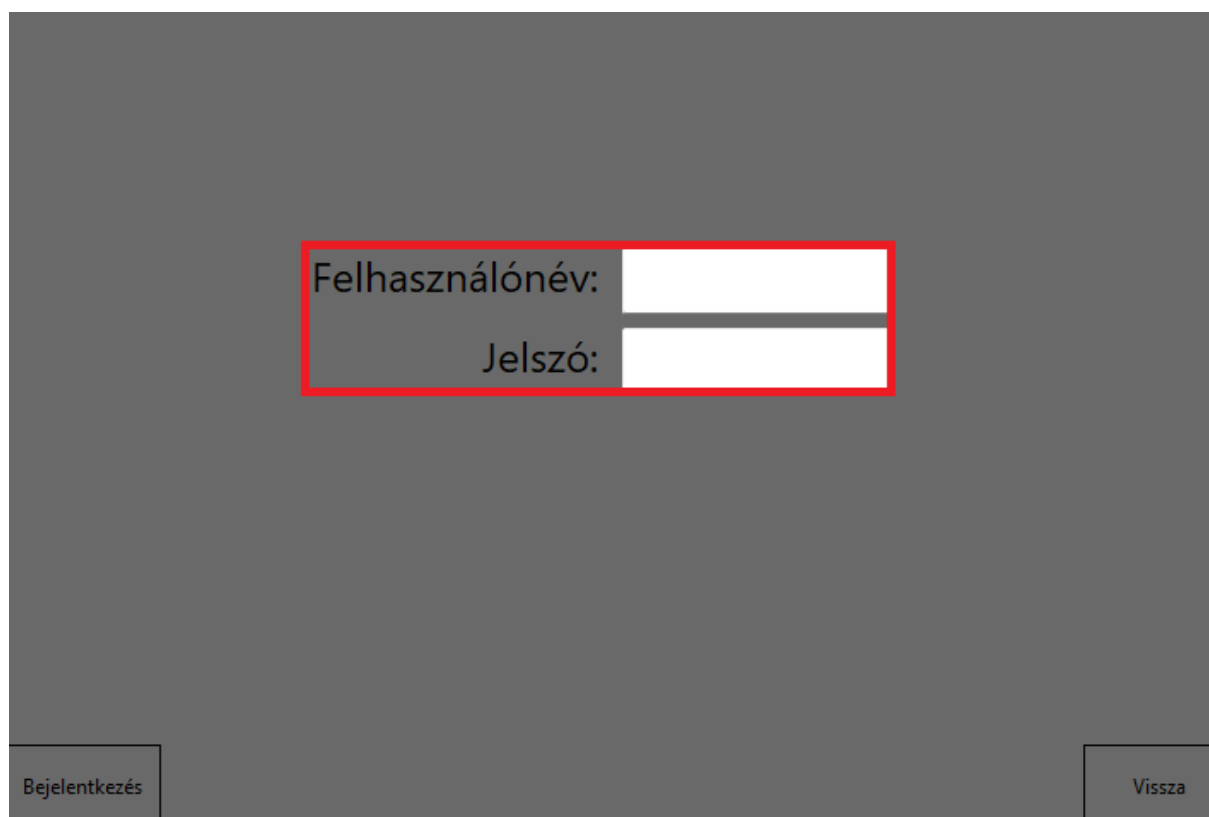
## Bejelentkezés

Ha már regisztráltunk a „Bejelentkezés” gombra kattintva nekikezdhethetünk a belépésnek.



14. ábra Bejelentkezés gomb

Ezek után be kell írunk a regisztrációkor megadott felhasználó nevünket és jelszavunkat.



The image shows a registration form on a dark gray background. The form consists of two input fields, one for the username and one for the password, which are highlighted by a red rectangular border. Below the input fields are two buttons: 'Bejelentkezés' (Registration) on the left and 'Vissza' (Back) on the right.

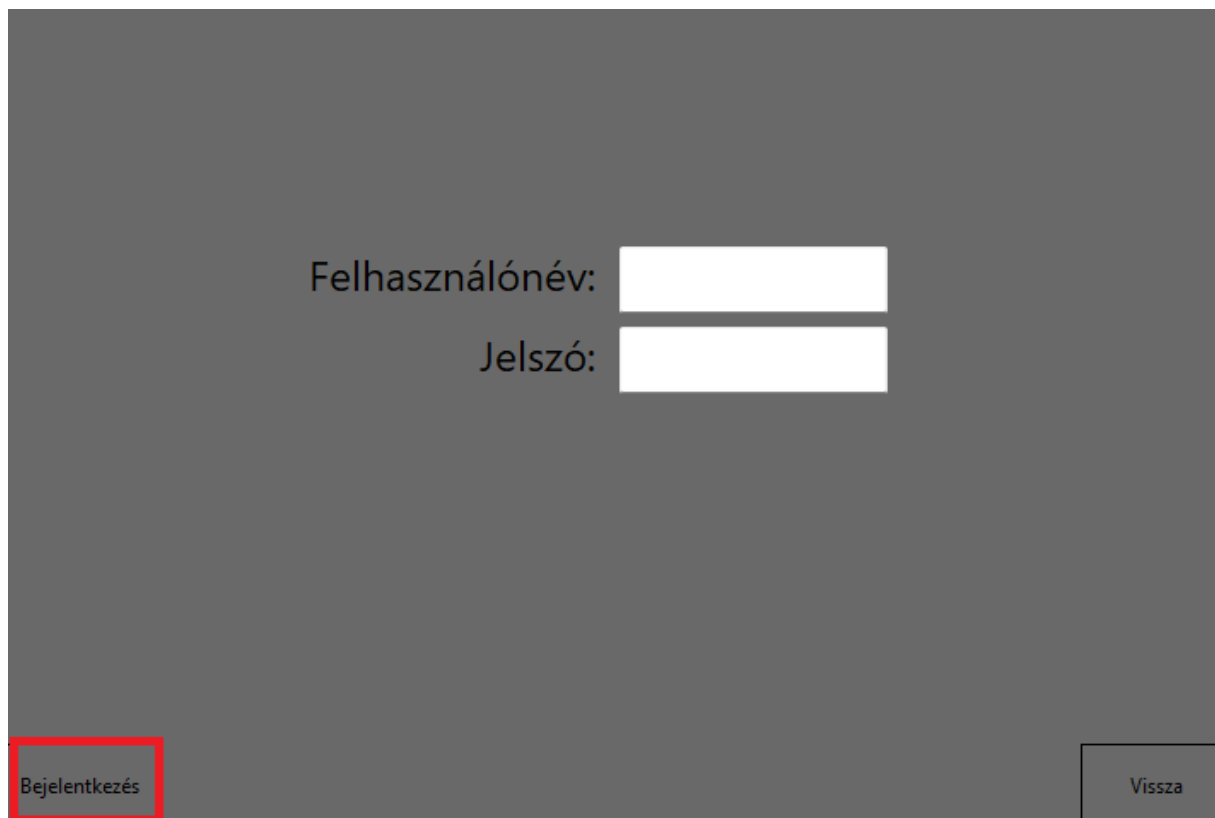
Felhasználónév:	<input type="text"/>
Jelszó:	<input type="password"/>

Bejelentkezés

Vissza

15. ábra Bejelentkezéshez szükséges mezők

A belépést a „Bejelentkezés” gombra kattintva lehet véglegesíteni.

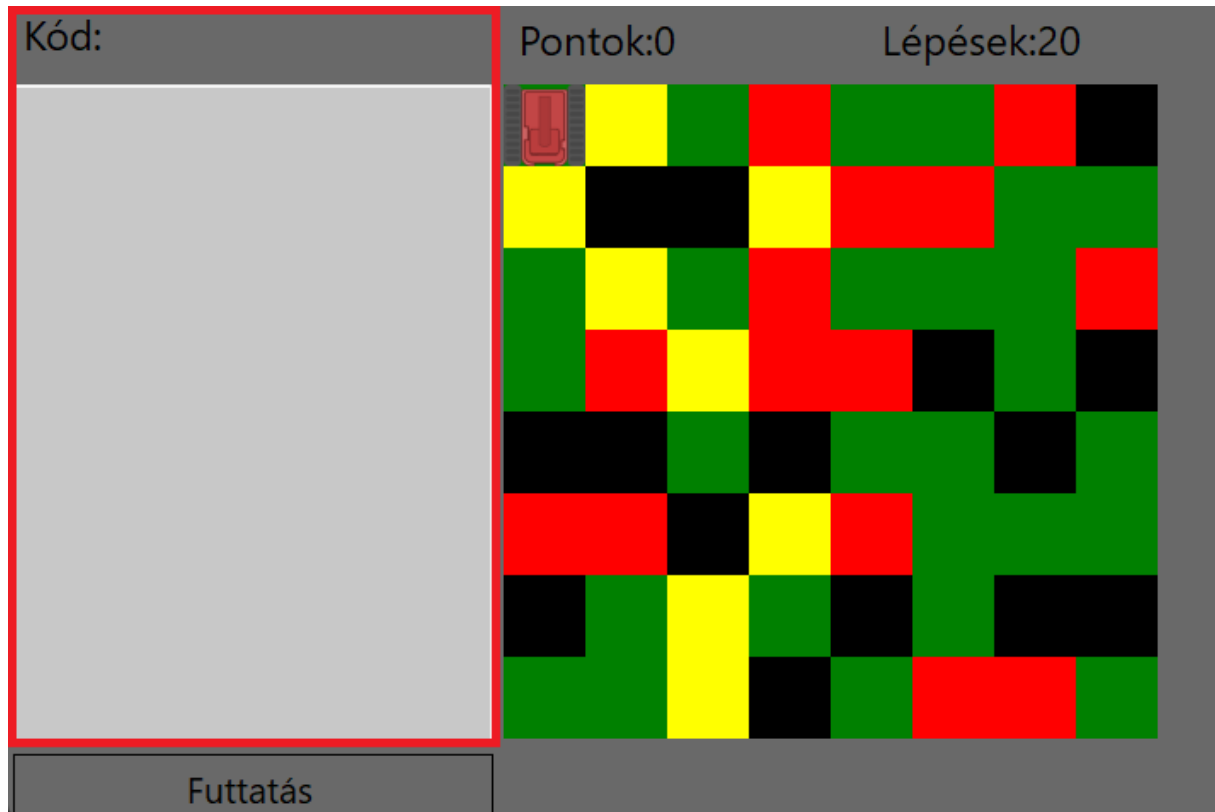


The image shows a login interface on a dark gray background. In the center, there are two white input fields. The first field is preceded by the text 'Felhasználónév:' and the second by 'Jelszó:'. At the bottom left, there is a button labeled 'Bejelentkezés' which is highlighted with a red rectangular border. At the bottom right, there is a button labeled 'Vissza'.

16. ábra Bejelentkezést véglegesítő gomb

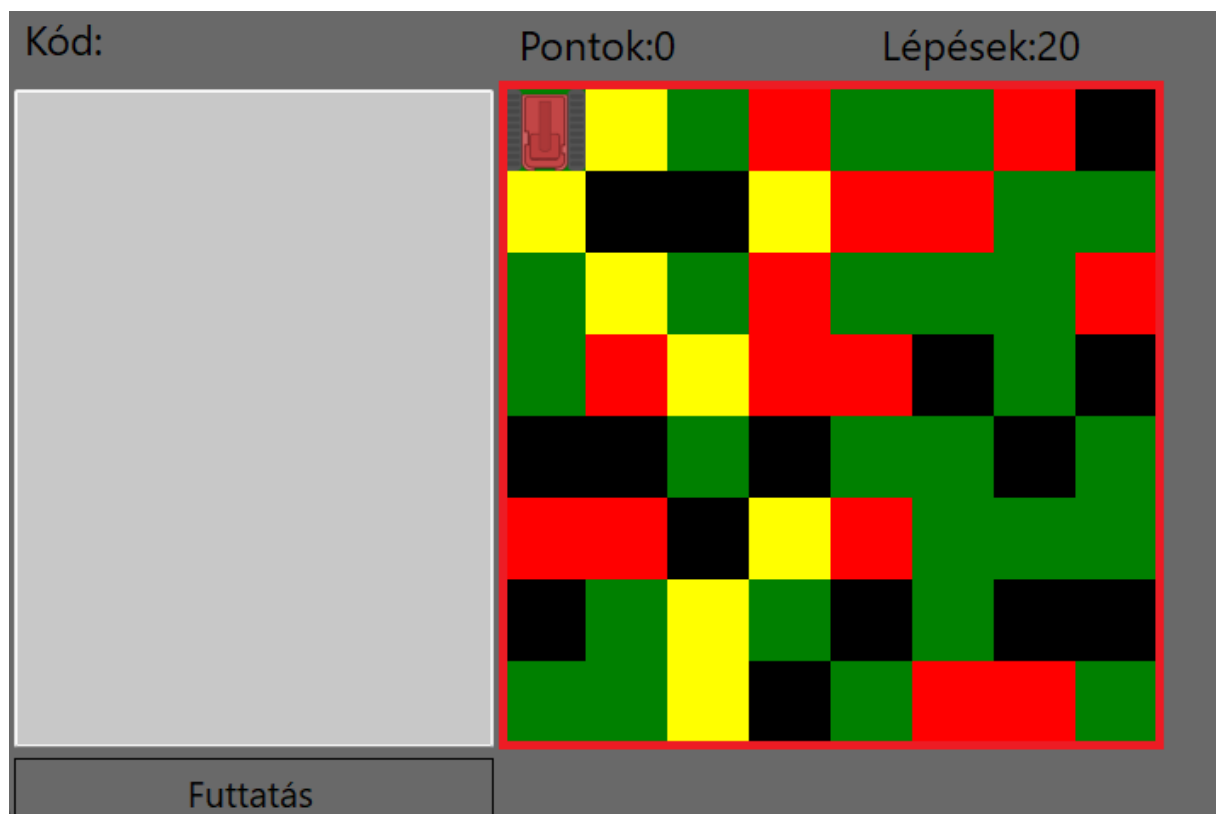
## A Játéktér

A játéktér két főbb elemből áll össze, egy szöveges felületből, ahol parancsokat adhatunk a tankunknak.



17. ábra Játéktér kód mezője

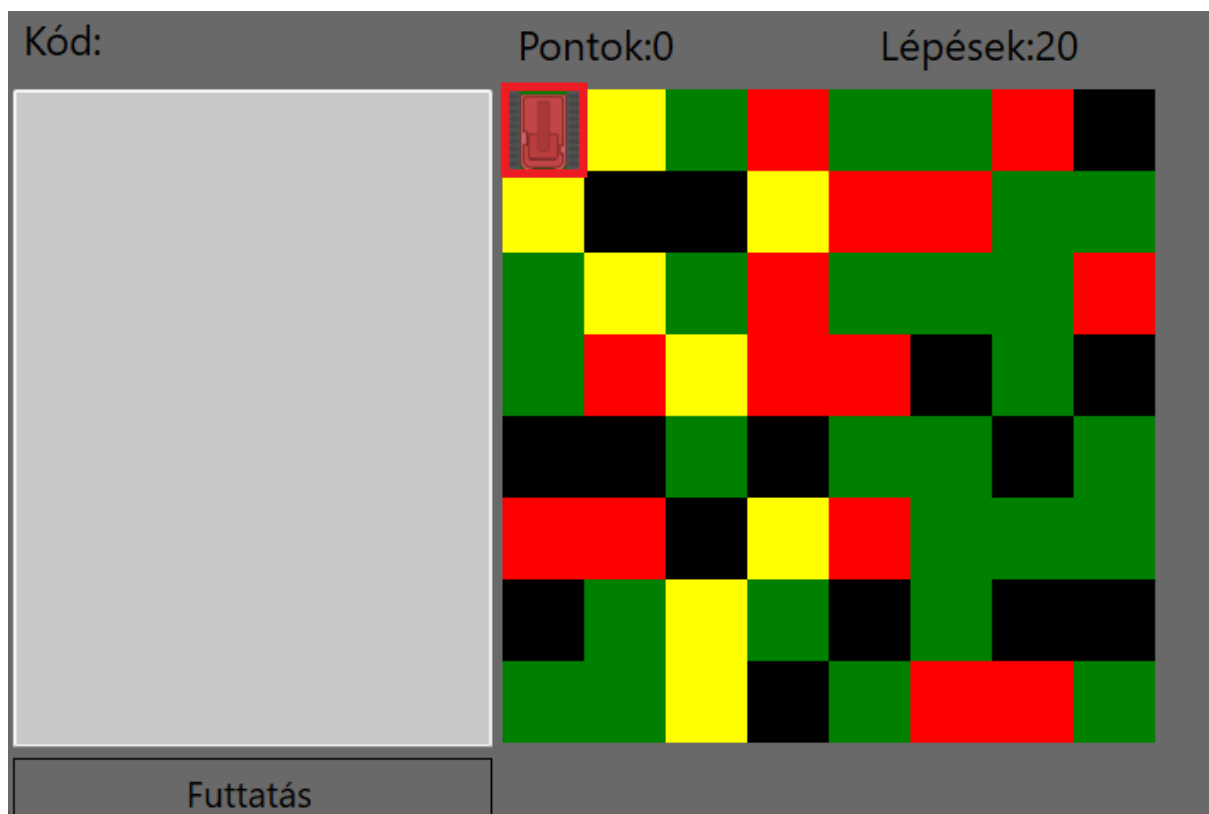
Illetve a térkép, ahol megfigyelhetjük, a parancsok eredményét, és tervet szőhetünk minél több pont megszerzésének reményében.



18. ábra Játéktér térképe

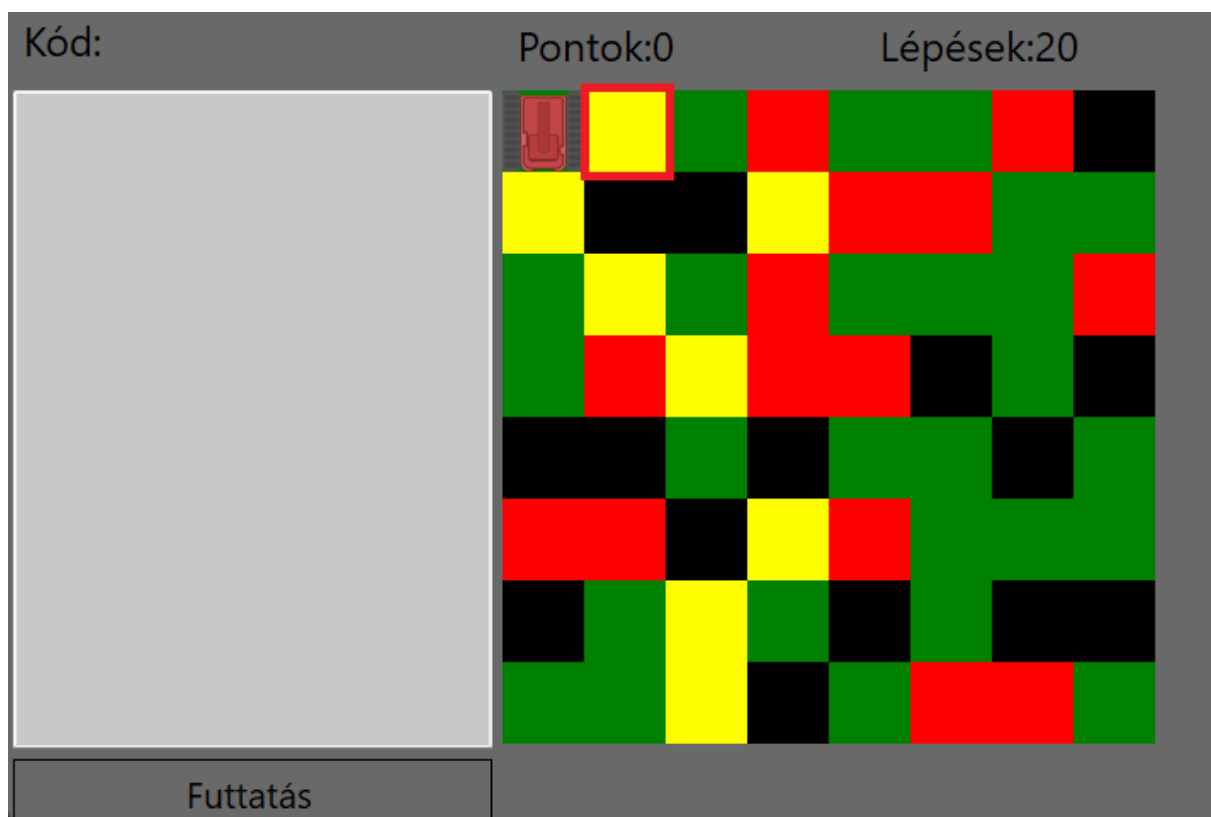
A térképen négy plusz egy elemet különböztethetünk meg.

1. A játékos, akivel pontokat szerezhethetünk.



19. ábra A játékos

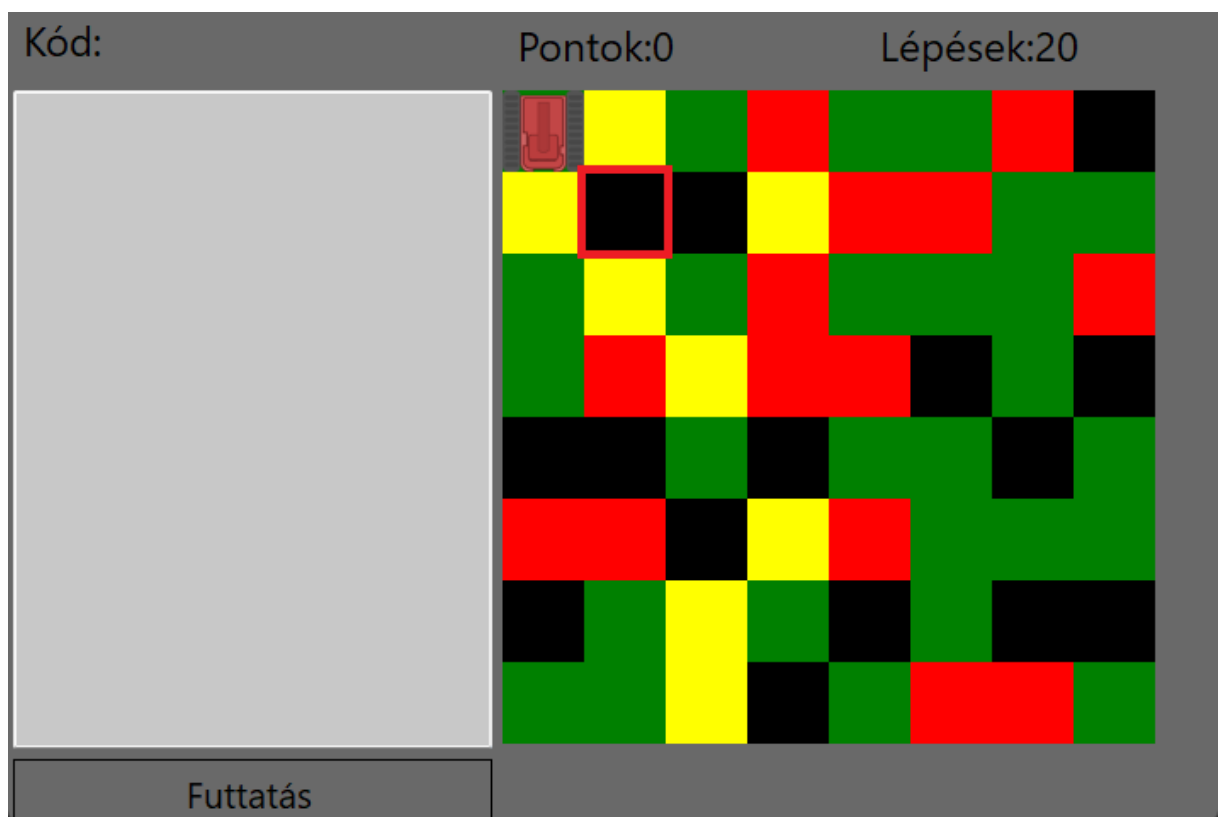
2. A összegyűjtendő pontok, amiket a sárga szín jelöl.



20. ábra A pontok

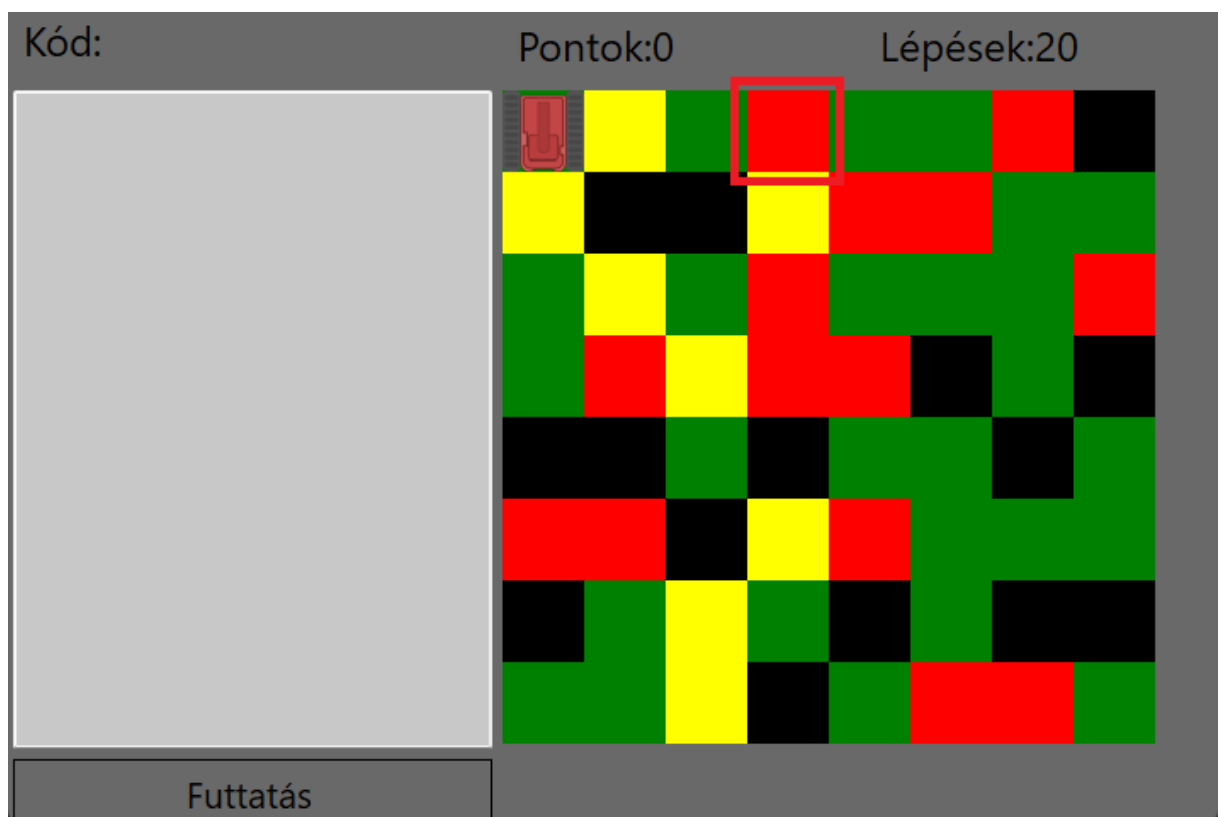


3. Akadályok ezeket a fekete szín jelöli.



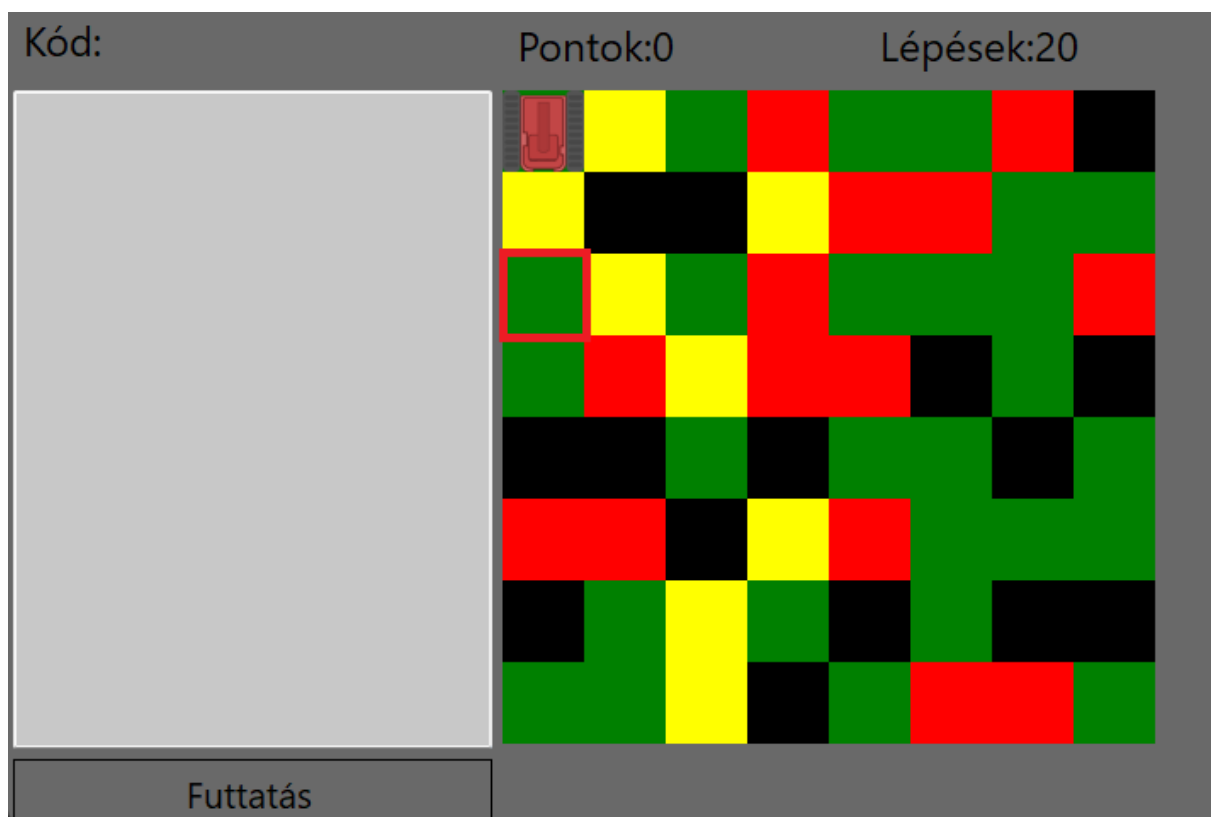
21. ábra Az akadályok

4. Aknák, amiket a piros szín jelöl.



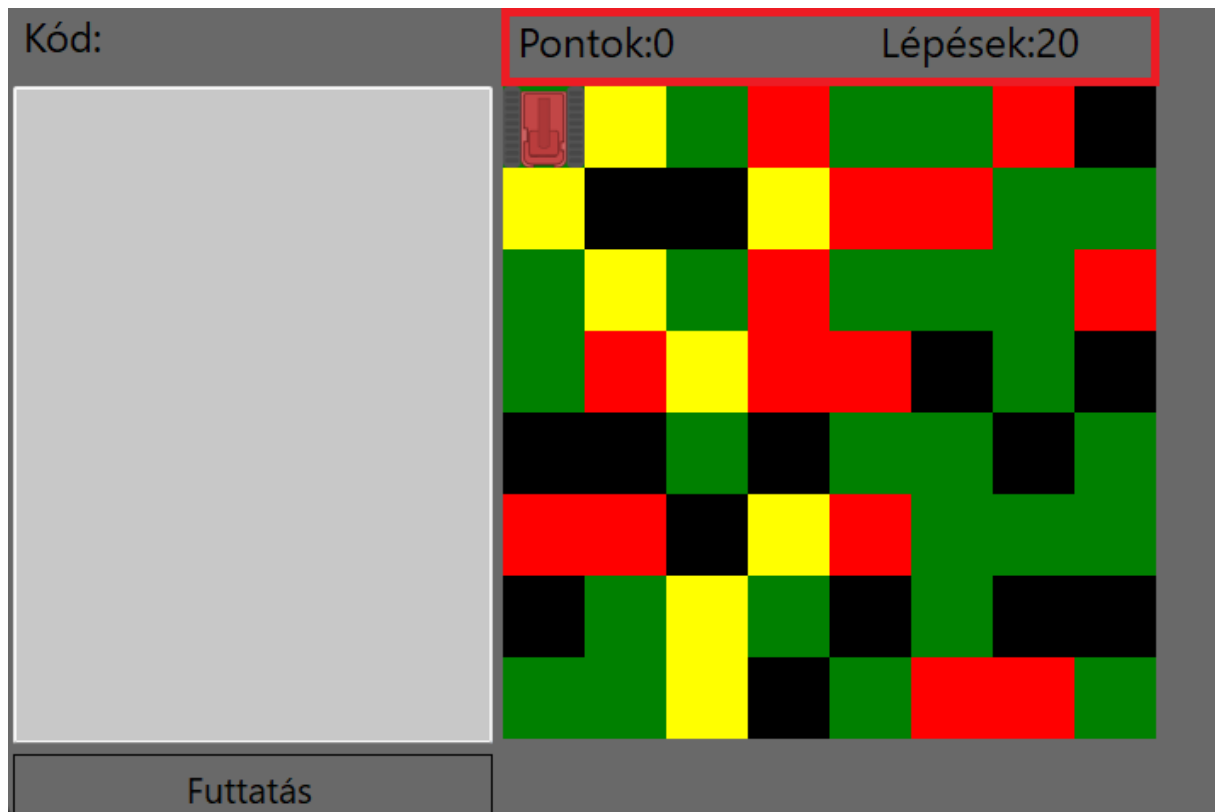
22. ábra Az aknák

5. Talaj, melynek színe zöld.



23. ábra A talaj

A játéktér felett a már megtalált pontoka, illetve a hátralévő lépéseinket figyelhetjük meg.



24. ábra Pontok és Lépések

A játék menete

Amikor a játéktérhez érünk egy véletlen, a program által generált térképpel fogunk találkozni. Feladatunk egyszerű, minél több pont begyűjtése a lehető legkevesebb lépésből. Amennyiben nem sikerül az összes pontot megszerezni, lehetőség van egy új világ indítására.

A tankokat a következő parancsokkal lehet irányítani:

- Jobbra(lépés);
- Balra(lépés);
- Fel(lépés);
- Le(lépés);

Ezek a parancsok a térképen a megfelelő irányba léptetik a tankot. Több parancsot is összerakhatunk pontosvesszőkkel elválasztva.

## Hardware és Software igény

Az alkalmazásunk működéséhez szükséges egy Windows operációs rendszeren futó számítógép. Az alkalmazást .NET Framework használatával fejlesztettem, így szükséges a számítógépen telepítve lennie a 6.0 verziójának.

Az alkalmazás működéséhez szükséges az egér, billentyűzet vagy egyéb input eszköz használata, amelyek segítségével a felhasználó interakcióba léphet az alkalmazással.

## Fejlesztési lehetőségek

Ezek közül a funkciók közül többet meg szerettem volna valósítani, de az idő és a komplexitás miatt ezt később tervezem megvalósítani.

### Lövés

Eredetileg szerettem volna létrehozni egy lövés funkciót, ami egy másik lehetőséghez, a többjátékos módhoz is csatlakozott volna.

### Többjátékosmód

A játék lehetne egy turn base többjátékos élmény, ahol először az egyik majd a másik játékos ír kódot és lép.

### Pályaválasztás

Mivel a véletlenszerűen generált pályákat eltárolom, így igen könnyen lehetne pálya-választót beépíteni, ahol a már régebben létrehozott pályákat lehet újrajátszani. Ugyanitt lehetne őket pontozni amire az adatbázis fel van készítve.

## Összegzés

Nagyon szívesen dolgoztam ezen a szakdolgozaton és sokat tanultam közben. Mikor már mélyen a munkálatokban voltam, akkor döbbentem rá, hogy egy PHP nyelven íródott szoftverrel jobban jártam volna, de mivel azt még csak egy éve ismerem nem voltam benne annyira magabiztos. Élmény volt a játékot megépíteni és utána látni a mosolyt a gyerekeken, akik azt kipróbálták. És ahogy lezárom ezt a szakdolgozatot, lassan ebben az iskolában eltöltött éveim is lezárulnak, és ezzel a pár szóval szeretném megköszönni a mérhetetlenül sok tudást és élményt, amit ezen falak között szerezhettem, zseniális türelmes tanárokkal.

1. ábra Adatbázis UML ábra .....	11
2. ábra Osztálydiagramm.....	12
3. ábra Main Program Class elemei .....	13
4. ábra FormMain elemei .....	14
5. ábra Login-Registerl elemei .....	15
6. ábra Register elemei .....	16
7. ábra Login elemei.....	18
8. ábra UCMain metódusai.....	20
9. ábra UCMain mezői .....	20
10. ábra Regisztrálás gomb .....	30
11. ábra Regisztráláshoz szükséges mezők .....	31
12. ábra Színválasztás .....	32
13. ábra Regisztrálás véglegesítése .....	33
14. ábra Bejelentkezés gomb.....	34
15. ábra Bejelentkezéshez szükséges mezők .....	35
16. ábra Bejelentkezést véglegesítő gomb .....	36
17. ábra Játéktér kód megjöje.....	37
18. ábra Játéktér térképe.....	38
19. ábra A játékos.....	39
20. ábra A pontok .....	40
21. ábra Az akadályok .....	41
22. ábra Az aknák.....	42
23. ábra A talaj .....	43
24. ábra Pontok és Lépések .....	44



## Irodalomjegyzék

AI, O. ( dátum nélkül.). *ChatGTP-3.5*. Forrás: <https://chat.openai.com/>

Cash, J. (2018. május 18). *Scratch vs. Swift Playgrounds*. Forrás: [makelearn.org:  
https://makelearn.org/2018/05/29/scratch-vs-swift-playgrounds/](https://makelearn.org/2018/05/29/scratch-vs-swift-playgrounds/)

ELTE IK, P. N. (2010). *elte.hu*. Forrás: A Python programozási nyelv:  
<http://nyelvek.inf.elte.hu/leirasok/Python/index.php?chapter=2>

ELTE IK, P. N. (2010). *VisualBasic*. Forrás: [elte.hu:  
http://nyelvek.inf.elte.hu/leirasok/VisualBasic/index.php?chapter=1](http://nyelvek.inf.elte.hu/leirasok/VisualBasic/index.php?chapter=1)

Kiadó, A. (2016). *A LOGO PROGRAMOZÁSI NYELV*. Forrás: [psg.hu:  
http://www.psg.hu/seged/9b/LOGO/comlogo.pdf](http://www.psg.hu/seged/9b/LOGO/comlogo.pdf)

Róbert, K. ( dátum nélkül.). *Programozási nyelvek Java*. Forrás: [elte.hu:  
https://kitlei.web.elte.hu/segedanyagok/foiak/java/hu-java-bsc/01alapok.pdf](https://kitlei.web.elte.hu/segedanyagok/foiak/java/hu-java-bsc/01alapok.pdf)

Tamás, K. ( dátum nélkül.). *Programozási nyelvek I. (Ada)*. Forrás: [elte.hu:  
http://kto.web.elte.hu/hu/oktatas/ada/eloadas/01.pdf](http://kto.web.elte.hu/hu/oktatas/ada/eloadas/01.pdf)