

## 2. Szoftverkövetelmény-specifikáció (SRS)

### 2.1 Bevezetés

#### Cél:

Ez a dokumentum meghatározza az *Infinite Loopers* nevű 2D-s endless runner játék funkcionális és nem funkcionális követelményeit. Leírja a rendszer célját, működését, fő funkcióit, valamint a fejlesztési korlátokat és jövőbeli bővítési lehetőségeket. Az SRS alapjául szolgál a fejlesztők, tesztelők és dokumentációkészítők számára a projekt teljes életciklusa során.

#### Hatókör:

Az *Infinite Loopers* egy ügyességi játék, amelyben a játékos célja, hogy a karakterét minél tovább életben tartsa, miközben elkerüli az akadályokat és pontokat gyűjt. A játék Pygame könyvtár segítségével készült, és futtatható Windows, Linux és macOS rendszereken. A cél egy egyszerűen kezelhető, gyors és szórakoztató játék létrehozása, amely a reflexeket és reakcióidőt teszi próbára.

### 2.2 Általános leírás

Felhasználók (aktorok):

#### • Játékos:

Az egyetlen aktív felhasználó, aki közvetlenül irányítja a játék menetét.

A játékos felelős a karakter mozgásáért, ugrásaiért, és minden olyan döntésért, amely befolyásolja a túlélési időt és a megszerezhető pontszámot.

A játékos feladatai közé tartozik a menürendszerben való navigáció, az új játék indítása, a szünetmenü használata és a játék újraindítása egy Game Over után.

A játékmechanika egyszerűségének köszönhetően a játékos gyorsan felveheti a ritmust, azonban az egyre növekvő sebesség és a véletlenszerű akadálygenerálás miatt a játék folyamatos odafigyelést igényel.

#### • Fejlesztő / Tesztelő (másodlagos):

A rendszer stabil működését biztosító háttérszereplők.

A fejlesztő feladata a kód módosítása, optimalizálása és bővítése, új grafikai vagy játékelemek hozzáadása, valamint az esetleges logikai hibák javítása.

A tesztelő szerepe a játék funkcióinak rendszeres ellenőrzése, a hibajelenségek reprodukciója és dokumentálása, illetve annak megállapítása, hogy a rendszer a követelményeknek megfelelően működik-e különböző környezetekben és terhelés mellett.

Ezek az aktorok biztosítják a projekt folyamatos fejlődését és a játék minőségének fenntartását.

Környezet:

#### • Operációs rendszer: Windows 10+

A játék fejlesztése és tesztelése elsősorban Windows 10 vagy újabb rendszereken történt, ahol

a Python és a Pygame futtatása teljes mértékben támogatott.

A program ugyanakkor hordozható, így Linux és macOS környezetekben is kompatibilis, de a teljes funkcionalitás és stabil grafikus megjelenítés Windows-on garantált.

- **Programozási nyelv: Python 3.9+**

A projekt Python alapokon működik, amely a gyors prototípus-készítést, az egyszerű hibakeresést és a moduláris kódstruktúra kialakítását teszi lehetővé.

A Python magas szintű szintaxisa és széleskörű könyvtártámogatása ideális választás volt egy oktatási célú játékfejlesztési projekthez.

- **Könyvtár: Pygame (grafika, eseménykezelés, hang)**

A Pygame biztosítja a játék grafikus megjelenítését, a sprite-ok mozgását, az animációkat, valamint a felhasználói bemenetek (billentyűk, zene, hanghatások) valós idejű kezelését.

A könyvtár magas szintű funkcióival könnyen lehet 2D-s játékokat fejleszteni, miközben elegendő rugalmasságot biztosít a testreszabáshoz.

- **Hardver: alapvető CPU és GPU (minimum 60 FPS teljesítményhez), billentyűzet**

A játék futtatásához nem szükséges különleges hardver.

Egy átlagos laptop vagy asztali PC is képes stabilan 60 FPS sebességgel futtatni a játékot.

A billentyűzet az egyetlen szükséges beviteli eszköz, mivel a játékmenet kizárólag két fő bemenetre – ugrás és menüvezérlés – épül.

- **Kijelző: 1280x720 (ajánlott), fut teljes képernyőn vagy ablakban is**

A játék minimum HD felbontásra lett optimalizálva, de a grafikai elemek méretezése rugalmas, így alacsonyabb és magasabb felbontásokon is megfelelően jelenik meg.

A felhasználó tetszés szerint játszhat teljes képernyős, illetve ablakos módban.

### Főbb korlátok és feltételezések:

- **Internetkapcsolat nem szükséges.**

A játék teljes mértékben offline működik, nincs szükség külső szerverkapcsolatra.

Ez gyors indítást és stabil futást biztosít olyan környezetekben is, ahol korlátozott vagy nincs internetelérés.

- **Nincs külső adatbázis; a High Score helyben mentődik.**

A pontszámok mentésére egyszerű fájlkezelést használ a program.

Ez minimális erőforrást igényel, és biztosítja, hogy a High Score ne vesszen el akkor sem, ha a játékos később offline játszik.

- **A játék nem támogat többjátékos módot.**

A rendszer kifejezetten egyjátékos élményre készült.

A multiplayer funkcionalitás bonyolultabb szinkronizációt és adatkezelést igényelne, amely meghaladja a jelen projekt célkitűzéseit.

- **A felhasználó rendelkezik a Python és Pygame futtatásához szükséges jogszolgáltatásokkal.**

Ez magában foglalja a program telepítését, futtatását és a fájlrendszer elérését (például a High Score mentéséhez).

Feltételezzük továbbá, hogy a felhasználó alapszintű számítógép-kezelési ismeretekkel rendelkezik.

## 2.3 Rendszerfunkciók (funkcionális követelmények)

### 1. Játék indítása

**Leírás:**

A játékos a főmenüből elindíthatja az új játékot.

Ez a folyamat aktiválja a játék inicializálási szakaszát, amely létrehozza a szükséges objektumokat, betölti a grafikai és hangfájlokat, majd elindítja a játékszálát. A játékindítás egyben az alapállapotba helyezi a karaktert és az akadálygeneráló rendszert.

**Bemenet:**

Egérkattintás vagy Enter billentyű.

A rendszer mindkét inputot felismeri, és az eseménykezelő modul továbbítja a játékindítási folyamatnak.

**Kimenet:**

A játék futása elkezdődik, a karakter mozgásba lendül.

A fő játékhurok frissítési ciklusai (FPS ciklusok) aktiválódnak, a háttér animáció és az akadályok generálása elindul.

**Előfeltétel:**

A program sikeresen elindult.

A Pygame keretrendszer megfelelően inicializálódott, az erőforrások betöltése nem okozott hibát.

**Utófeltétel:**

A játék fő ciklusa aktív.

A játékos átkerül a játékmenet állapotába (Game State: Running), készen az interakciókra.

**Kiegészítés:**

A folyamat célja, hogy a játékos minimális várakozás mellett léphessen be a játékmenetbe. A játékindítás stabil működése meghatározó a felhasználói élmény szempontjából, mert ha az inicializálás során hiba történik, az megakadályozza az egész játékmenetet. Emiatt a rendszer több biztonsági ellenőrzést is végez, például ellenőrzi a szükséges fájlok meglétét és a grafikus ablak sikeres megnyitását.

### 2. Karakter irányítása

**Leírás:**

A játékos a karaktert billentyűzettel irányítja.

Az irányítás egyszerű, egyetlen mozdulatra épül: az ugrásra. A minimalista vezérlés biztosítja, hogy a játék könnyen elsajátítható legyen, ugyanakkor reflexet igénylő kihívást biztosítson.

**Bemenet:**

Space vagy ↑ – ugrás

Mindkét gomb ugyanazt a műveletet váltja ki, így a játék akadálymentes különböző irányítási szokások esetén is.

**Folyamat:**

A karakter animációval ugrik, és elkerüli az akadályokat.

A játékmotor kezeli a gravitációt, az ugrási ívet, a sebességet és a leérkezést. A fizikai modell

garantálja, hogy az ugrás időzítése és magassága minden esetben kiszámítható és reális legyen.

**Kimenet:**

Az aktuális pozíció frissül.

A rendszer minden képkockában újraszámolja a karakter helyzetét.

**Utófeltétel:**

Ütközés esetén Game Over állapot.

A kollíziókezelő modul észleli az akadályokkal való érintkezést és kiváltja a játék végét.

**Kiegészítés:**

A karakter irányítása a játék egyik legkritikusabb pontja. A bemenetek valós idejű feldolgozására van szükség, ezért a rendszer az eseményeket azonnal regisztrálja. A hibamentes kollíziókezelés és a lag-mentes mozgás kulcsfontosságú a játékelmény szempontjából. Mivel a játék sebessége fokozatosan növekszik, az ugrások időzítése egyre nehezebbé válik, ezzel növelve a kihívást.

### 3. Pontszám és rekordrendszer

**Leírás:**

A rendszer méri a megtett távolságot és a gyűjtött pontokat.

A pontszám real time frissül, és a HUD folyamatosan mutatja a játékos aktuális eredményét.

**Folyamat:**

A játékos pontot kap idő és túlélés alapján.

Minden megtett távolság, illetve túlélési másodperc hozzájárul az összpontszámhoz.

**Kimenet:**

Pontszám és High Score megjelenítése a HUD-on.

A játék végén összehasonlításra kerül a korábbi rekorddal.

**Utófeltétel:**

A High Score helyileg mentődik a fájlba.

Ez biztosítja az eredmények tartósságát internet nélkül is.

**Kiegészítés:**

A pontszámítás motivációt biztosít a játékos számára, hogy újra és újra próbálkozzon. A High Score fájl helyi tárolása egyszerű és gyors megoldás, amely nem igényel adatbázist vagy internetkapcsolatot. A rendszer védetten kezeli a fájlmegnyitást, így hiba esetén sem omlik össze a játék.

### 4. Menükezelés

**Leírás:**

A játék főmenüt, szünetmenüt és Game Over képernyőt tartalmaz.

A menü feladata, hogy átlátható módon vezesse a játékost a különböző funkciók között.

**Főmenü:**

új játék, beállítások, kilépés

A játékos itt tudja elindítani vagy személyre szabni a játékot.

**Pause menü:**

folytatás, újratekés, főmenü

A játékos ideiglenesen felfüggesztheti a futást.

**Game Over:**

pontszám kijelzése, újratekés, főmenü

A rendszer teljes körű visszajelzést ad a futás eredményéről.

**Bemenet:**

egérkattintás vagy Enter billentyű

**Kiegészítés:**

A menürendszer a játék vezérlési logikájának alapja. A grafikus felület könnyen értelmezhető, a menüpontok nagy méretűek és jól láthatóak. A menürendszer folyamatos visszajelzést ad az interakciókról (hangok, vizuális kiemelések). A Game Over képernyő különösen fontos, mert ez biztosítja a felhasználó számára, hogy könnyen újratekeshesse a játékot.

## 5. Hangeffektek és háttérzene

**Leírás:**

Minden eseményhez megfelelő hangeffekt tartozik (ugrás, ütközés, menü).

**Előfeltétel:**

A sounds/ mappa megfelelő helyen van.

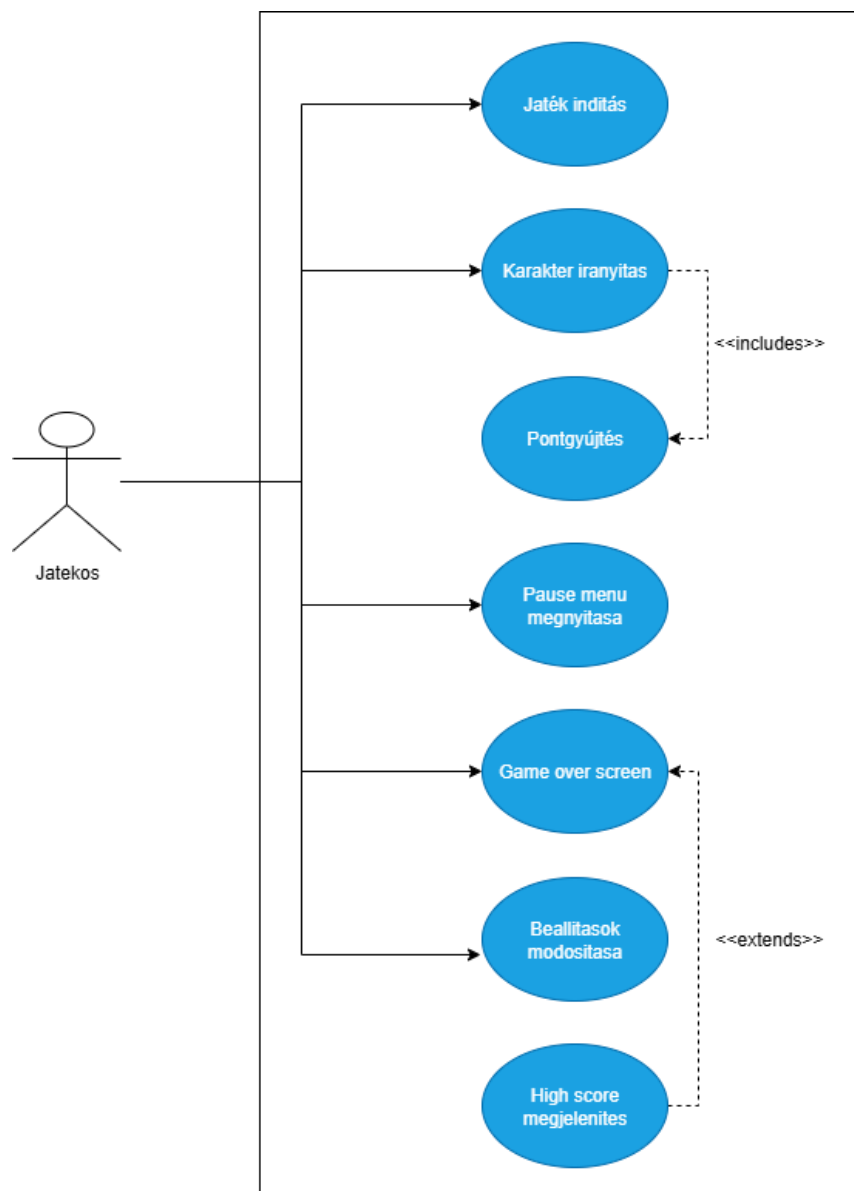
**Utófeltétel:**

A hang lejátszódik az eseményhez kötve.

**Kiegészítés:**

A hanghatások jelentősen hozzájárulnak a játékelményhez. A Pygame mixer modul segítségével a hangok minimális késleltetéssel szólnak meg. A háttérzene fenntartja a ritmust és feszültséget, míg az ütközési hangok egyértelmű visszajelzést adnak a játék végéről. Fontos, hogy a hangkezelés ne okozzon FPS-esést, ezért optimalizált formátumokat használunk (pl. OGG, WAV).

## 2.4 Use Case diagram és leírások



## 2.5 Nem funkcionális követelmények

- **Teljesítmény:** Legalább 60 FPS átlagos képkockaszám standard hardveren.

A játék optimális működéséhez szükséges, hogy a felhasználó gépe stabilan képes legyen legalább 60 FPS mellett futtatni az alkalmazást.

Ez a sebesség biztosítja az animációk folyamatosságát, az akadályok megfelelő időzítését, valamint az irányításra adott azonnali reakciót.

A Pygame motor belső ciklusvezérlést (clock) használ az FPS limitálására, így a játék futása állandó és kiszámítható marad.

A fejlesztés során kiemelt figyelmet kaptak az optimalizált sprite-kezelések, a minimalizált háttérképfriessítések és a felesleges számítások csökkentése, hogy gyengébb hardveren is stabil teljesítmény érhető legyen el.

A 60 FPS-es teljesítmény kötelező elem, mert a gyors játéktempó miatt a késés, akadás vagy FPS-esés közvetlenül befolyásolná a játékos élményét és az irányítás pontosságát.

- **Használhatóság:** Egyszerű, billentyűzetről irányítható kezelőfelület, vizuálisan letisztult design.

A használhatóságra nagy hangsúlyt fektettünk a fejlesztés során. A kezelőfelület minimalista kialakítása célja, hogy minden funkció könnyen értelmezhető és azonnal elérhető legyen.

A játék irányítása szándékosan egyszerű (egy gomb – ugrás), így kezdő felhasználók számára is egyértelmű.

A menük nagy, jól olvasható gombokkal működnek, és vizuális visszajelzéseket (kiemelés, hanghatás) adnak a felhasználónak.

A válaszidő az interakciókra gyors és kiszámítható, ami csökkenti a felhasználói hibák számát és növeli az élményt.

A grafikai megjelenítés letisztult, a színeket és a kontrasztokat úgy választottuk meg, hogy hosszabb játék során se fárasza a szemet, és minden fontos elem azonnal felismerhető legyen.

- **Megbízhatóság:** A játék futása közben nem omlik össze hibás bevitel vagy hosszú futás esetén.

A megbízhatóság az egyik legfontosabb nem funkcionális követelmény, mivel a játék folyamatos működését nem befolyásolhatják véletlen vagy hibás bemenetek.

A Pygame eseménykezelő mechanizmusa biztosítja, hogy minden billentyűlenyomás biztonságos módon legyen feldolgozva, így a program hibás bemenet esetén sem omlik össze.

A fejlesztés során hosszú futási tesztek végeztünk, hogy kizárjuk a memóriafolyást, a frame-idő növekedést vagy a rendszer válaszkészségének csökkenését.

Az akadályok generálása és mozgatása is stabilan működik több órás futás után, mivel ezek a folyamatok optimalizált ciklusokban történnek.

Külön figyelmet kapott a hibakezelés (try-except blokkok), amelyek gondoskodnak róla, hogy egy-egy hirtelen bekövetkező grafikai vagy hanghiba se okozzon összeomlást.

- **Karbantarthatóság:** A kód jól strukturált, modulokra bontott (main loop, player, obstacle, UI).

A projekt kódját több különálló modulba szerveztük, így a rendszer átlátható, könnyen javítható és bővíthető.

A **main** modul tartalmazza a játékhurkot és az állapotkezelést,

a **player** modul a karakter viselkedését és fizikai logikáját,  
az **obstacles** modul az akadályok létrehozását és mozgatását,  
a **UI** modul pedig a menük és felhasználói elemek kirajzolását végzi.  
Ez a moduláris felépítés lehetővé teszi, hogy egy fejlesztő gyorsan megtalálja és módosítsa a szükséges logikai egységeket anélkül, hogy a teljes projektet át kellene néznie.  
A kód dokumentálása, a beszédes változónevek és a következetes formázás még tovább növelik a karbantarthatóságot.

- **Biztonság:** A pontszámok mentése lokális fájlban történik, adatvesztés minimalizálva.

A játék nem igényel bonyolult adatkezelést vagy online kapcsolatot, ezért a pontszámok mentése egyszerű, helyi szövegfájlba történik.

A rendszer ellenőrzi, hogy a High Score fájl létezik-e, és ha nem, automatikusan létrehozza, így biztosítva, hogy az eredmények ne vesszenek el.

A fájlírás védett környezetben történik (try-except blokkokkal), így akkor sem omlik össze a program, ha az írási jogok korlátozottak vagy a fájl átmenetileg nem érhető el.

A játék semmilyen felhasználói adatot nem gyűjt, nincs adat továbbítás vagy tárolás, így a biztonság kockázat minimális.

A helyi mentés gyors, stabil és offline környezetben is működik.

- **Kompatibilitás:** Minden Python 3.9+ és Pygame-kompatibilis környezetben fut.

A játék fejlesztésének egyik célja az volt, hogy a lehető legtöbb platformon futtatható legyen. A Python és a Pygame széleskörű kompatibilitása miatt a játék működik Windows, Linux és macOS rendszereken egyaránt.

Nincsenek platformfüggő beállítások vagy könyvtárak a kódban, így a játék könnyen hordozható.

Amennyiben a szükséges függőségek telepítve vannak (Python + Pygame), a játék külön konfiguráció nélkül futtatható bármilyen eszközön.

Ez különösen előnyös oktatási környezetben, ahol a hallgatók különböző operációs rendszereket használnak.

## 2.6 Korlátok és függőségek

- **Hardver:** Alap szintű GPU és CPU szükséges.

A játék futtatásához nincs szükség kifejezetten nagy számítási teljesítményre, mivel a Pygame motor elsősorban CPU-alapon kezeli a grafikus megjelenítést.

A 2D sprite-ok, háttérretegek és animációk feldolgozása egy átlagos laptop vagy régebbi asztali gép számára is könnyedén kezelhető.

A CPU szerepe a játék fő logikájának futtatása, az ütközésérzékelés, az akadálygenerálás és az időzítések feldolgozása.

A GPU minimális terhelést kap, mivel a játék nem használ 3D gyorsítást vagy nagy felbontású textúrákat.

Ezzel együtt feltételezés, hogy a felhasználónak stabil hardverkönyezete van, amely képes tartós 60 FPS megjelenítésére anélkül, hogy túlmelegedést vagy teljesítménycsökkenést okozna.



- Szoftver: Python 3.9+, Pygame 2.0+

A rendszer működésének alapja a Python modern verziója, amely biztosítja az erőforrások hatékony kezelését, a funkcionális modulok támogatását és a gyors hibakeresést.

A Python 3.9+ azoknak a könyvtáraknak és funkcióknak ad stabil környezetet, amelyek nélkülözhetetlenek a játék futtatásához, például:

- korszerű memória-kezelés
- új szintaktikai lehetőségek
- optimalizált interpreter-teljesítmény

A Pygame 2.0 vagy újabb verziója elengedhetetlen a grafikus, hang- és eseménykezelési funkciókhoz.

A régebbi Pygame verziók hiányos kompatibilitása vagy elavult API-ja akadályozná a játék stabil működését, ezért a projekt kifejezetten az újabb verziókra épült.

- Függőségek:

A játék több Python-modulra is támaszkodik, amelyek külön feladatokat látnak el a futás során.

*o pygame – grafika, hang, eseménykezelés*

Ez a könyvtár felelős a megjelenítésért, a játéktér kirajzolásáért, az animációk futtatásáért, a hangok lejátszásáért, valamint a valós idejű események (billentyűlenyomások, kilépési parancsok) feldolgozásáért.

A Pygame teszi lehetővé a játékhurok működését, amely minden képkockán frissíti a játék állapotát.

*o os, sys, random – alap Python modulok*

Ezek a standard könyvtárak beépített Python-modulok, így nem igényelnek külön telepítést.

- **os**: fájlmenedzsment, könyvtárhozzáférés, High Score fájl kezelése
- **sys**: a program rendszerparamétereinek és kilépési folyamatait kezeli
- **random**: véletlenszerű akadálygenerálás biztosítása a játékmenet kiszámíthatatlanságához és változatosságához

A modulok használata biztosítja, hogy a játék könnyen bővíthető és karbantartható maradjon.

- Külső erőforrások:

A játék megfelelő vizuális és hangeleinek betöltéséhez szükség van bizonyos fájlokra.

*o Grafikai elemek a graphics/ mappában*

Ebben a mappában található a játék vizuális komponensei, köztük:

- karakter sprite-ok
- akadályok grafikái
- háttérképek és animált rétegek
- menü és UI elemek

A képek optimalizált formátumban (jellemzően PNG) kerültek elhelyezésre, hogy betöltésük gyors legyen, és ne terheljék túl a memóriát vagy a CPU-t.

A grafikai elemek hiánya vagy hibája esetén a játék nem képes megjeleníteni a pályát, ezért ezek megléte kritikus követelmény.

Ebbe a könyvtárba kerültek az alábbi hangfájlok:

- ugrás hang
- ütközési effekt
- menü interakció hang
- háttérzene

A játék a Pygame mixer modulját használja a hangok lejátszásához.

A hangfájlok egységes formátuma (WAV vagy OGG) biztosítja a gyors beolvasást és a késleltetésmentes lejátszást.

A hangok megfelelő működése hozzájárul a játékelményhez, így ezek hiánya vagy sérülése a játékelmény romlását eredményezné.

## **2.7 Feltételezések és jövőbeli fejlesztések**

### **Feltételezések:**

- A felhasználó rendelkezik működő Python környezettel.

A játék futtatásához elengedhetetlen, hogy a felhasználó olyan operációs rendszert használjon, amelyen a Python telepítve és megfelelően konfigurálva van.

Ez magában foglalja a Python értelmező helyes elérési útjának beállítását (PATH), valamint a szükséges könyvtárak – különösen a Pygame – telepítését.

Feltételezzük továbbá, hogy a felhasználó képes futtatni a `.py` fájlokat, illetve rendelkezik alapvető hibakezelési ismeretekkel, például felismeri, ha egy modul hiányzik vagy környezeti probléma merül fel.

Ez a követelmény minimális technikai hozzáértést igényel a játék futtatójától, de az oktatási jelleg miatt nem okoz jelentős nehézséget.

- A játékos ismeri az alapvető billentyűparancsokat.

A játék irányítása egyszerű, de feltételezi, hogy a felhasználó tudja, hogyan működik a billentyűzet és annak alapvető funkciói.

Mivel a játék egyetlen fő irányítási mechanizmusa az ugrás (Space vagy ↑), a játékosnak tisztában kell lennie azzal, hogyan adjon be gyors, pontos inputokat.

A menürendszer navigálása szintén ismer olyan alapl műveleteket, mint az Enter és az egérgattintás.

A játékmechanikát úgy alakítottuk ki, hogy ne terhelje túl a felhasználót, de a gyors reakcióidő miatt fontos a billentyűzet magabiztos használata.

- A rendszer képes kezelni a billentyűzetes bemenetet valós időben.

Ennek technikai feltétele, hogy az operációs rendszer és a Pygame között a bemeneti események kezelése késleltetés nélkül történjen.

A játék valós idejű inputot igényel, különösen akkor, amikor az akadályok gyors egymásutánban jelennek meg, és az ugrások időzítése kritikus.

A rendszer feltételezi, hogy a bemeneti eszköz (billentyűzet) megfelelően működik, nincs beviteli késés (input lag), és a Pygame képes minden eseményt rögzíteni a játékhurok minden egyes ciklusában.

Ez garantálja a precíz irányítást és a játékelmény pontosságát.

## Jövőbeli fejlesztések:

- Több pályatípus és akadályvariáció.

A jelenlegi rendszer egyetlen pályatípust támogat, amely véletlenszerűen generált akadályokkal dolgozik.

A jövőben több környezet, háttérkép, akadálymodell és animáció adható a játékhoz, amelyek növelik a változatosságot és új kihívásokat teremtenek.

Lehetőség nyílik tematikus pályák (pl. sivatag, űr, város, erdő) bevezetésére, külön egyedi akadályokkal, mozgó platformokkal vagy interaktív elemekkel.

Ez nemcsak látványosabbá tenné a játékot, hanem új játékmenet-mintákat is engedne beépíteni.

- Teljes képernyős mód és mobilverzió támogatás.

A Pygame támogatja a fullscreen módot, azonban ennek implementálása nagyobb grafikai és felhasználói felület módosítást igényel.

A mobilverzió elkészítése külön technológiát igényel (pl. Kivy, pygame-ce for Android), amely lehetővé tenné érintőképernyős irányítás bevezetését.

A mobilplatformra való portolás a projekt jelentős bővítése lenne, amely megnövelné a célközönséget és a játék hozzáférhetőségét.

Ez új inputkezelési, optimalizálási és UI áttervezési feladatokat is jelentene.

- Power-up rendszer (ideiglenes képességek, pl. pajzs, dupla ugrás).

A power-up rendszer olyan játékelemeket tartalmazna, amelyek ideiglenesen megváltoztatják a karakter képességeit.

Példák:

- **Pajzs:** egy ütközést túlél a játékos
- **Dupla ugrás:** két egymást követő ugrást hajthat végre
- **Sebességlassítás:** átmenetileg lassítja a játékmenetet
- **Pontszorzó:** bizonyos ideig többszörös pontszámot ad

Ezek a funkciók jelentősen növelnék a játék újrajátszhatóságát és taktikai mélységet adnának a játékmenethez.

A rendszer implementálása új grafikai elemeket, logikai modulokat és kollíziókezelési kiterjesztést igényel.

## 2.8 Hivatkozások

- Infinite Loopers – Szótár
- Pygame hivatalos dokumentáció: <https://www.pygame.org/docs/>
- Python hivatalos oldal: <https://www.python.org/>