Bolyai Farkas Elméleti Líceum

Marosvásárhely

Arcade Games Offline



Készítette:

Szövérfi Márk

Monda Antonio

XII.G Matematika-Informatika, intenzív informatika osztály

Felkészítő tanár: Jakab Irma-Tünde

2025

**Tartalomjegyzék**

1. Témaindoklás . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3
2. Rendszer Követelmény . . . . . . . . . . . . . . . . . . . . . . . 4
3. Felhasználói Kézikönyv . . . . . . . . . . . . . . . . . . . . . . . 5
4. Programozói Kézikönyv . . . . . . . . . . . . . . . . . . . . . . 12
5. Fejlesztési lehetőségek . . . . . . . . . . . . . . . . . .. . . . . 22
6. Szakirodalom . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 23
7. **Témaindoklás**

A projektünk célja egy offline működő applikáció megalkotása, amely a klasszikus arcade játékok világát idézi meg, modern köntösben. Az ötletet a csapat tagjainak közös érdeklődése inspirálta, hiszen mindannyian rajongunk a retro videojátékok iránt, és fontosnak tartjuk, hogy ezek a régi, ikonikus játékok ne merüljenek feledésbe. Szerettük volna az offline játéklehetőséget előtérbe helyezni, hogy a játékélmény bárki számára elérhető legyen, internetkapcsolat nélkül is.

A projekt előkészítése során részletesen megvitattuk, hogyan tudnánk a klasszikus játékélményt visszahozni a mai felhasználók számára. Olyan játékokat terveztünk, amelyek egyszerűek és letisztultak, hogy a felhasználók könnyen hozzáférjenek és élvezhessék a nosztalgikus élményeket. Az offline működés mellett kiemelt figyelmet fordítottunk arra, hogy az applikáció interaktív és vizuálisan vonzó legyen.

A fejlesztés során Python programozási nyelvet használtunk, amely lehetővé tette számunkra, hogy rugalmas és hatékony megoldásokat alkalmazzunk. Ez nemcsak technikai ismereteink mélyítésében segített, hanem betekintést nyújtott abba is, hogy milyen kihívásokkal kell szembenéznünk egy applikáció létrehozása során, a tervezéstől egészen a megvalósításig.

Reméljük, hogy az applikációnk felhasználói élménye nemcsak nosztalgikus hangulatot teremt, hanem hozzájárul az arcade játékok kultúrájának megőrzéséhez és népszerűsítéséhez. Kellemes szórakozást kívánunk minden játékosnak!

1. **Rendszer követelmények**

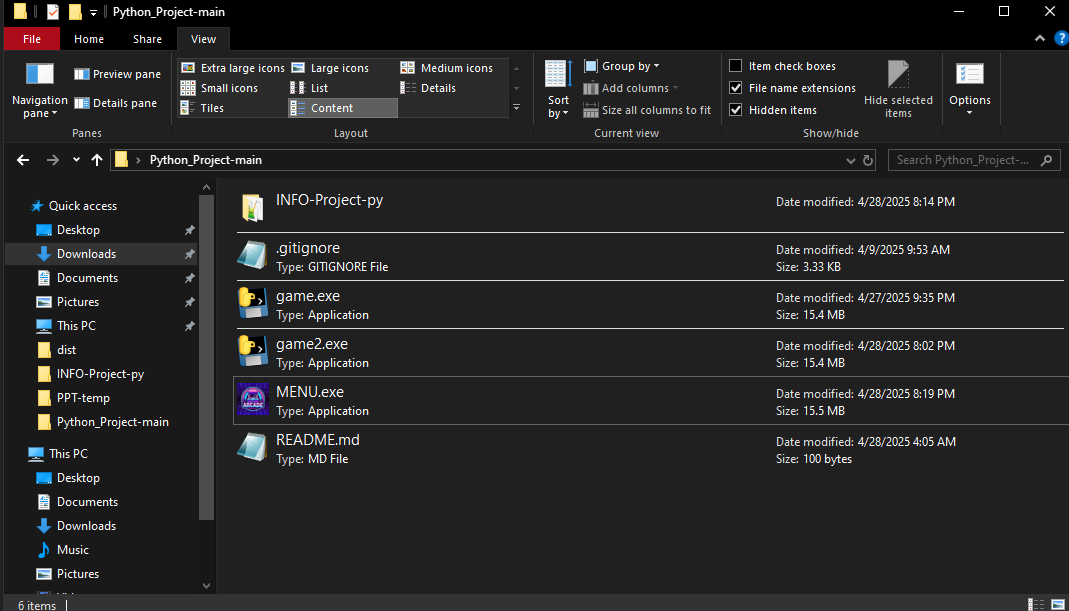
**Hardver igény:**

* Minimum: 2 GB RAM;
* Minimum: 2 magos 64 bites processor
* Minimum: 300 MB szabad tárhely
* Integrált grafika elegendő
* Legalább **14-15 hüvelyk**, Full HD (1920x1080) ajánlott

**Szoftver igény:**

* Windows 10/11 vagy bármilyen operációs rendszer ami tud .exe programokat futtatni
* Python 3.9 vagy újabb

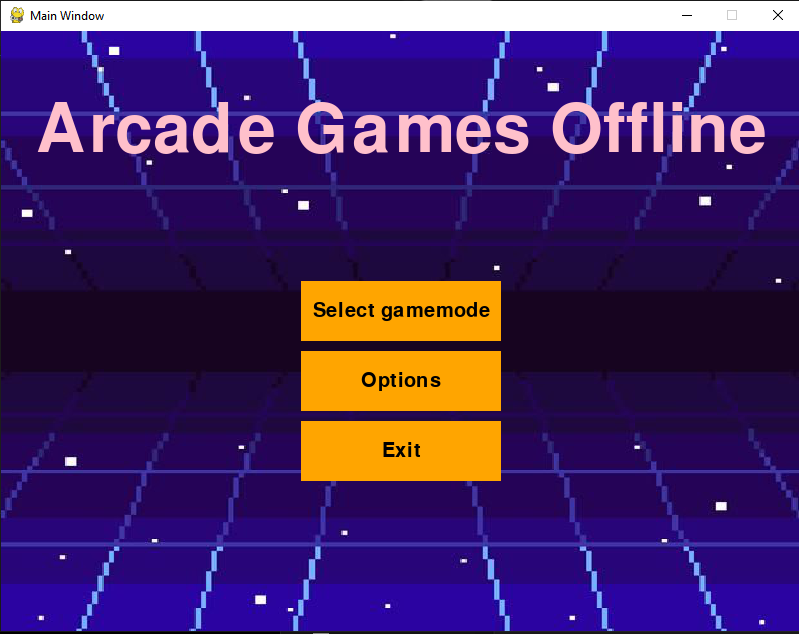
1. **Felhasználói kézikönyv**



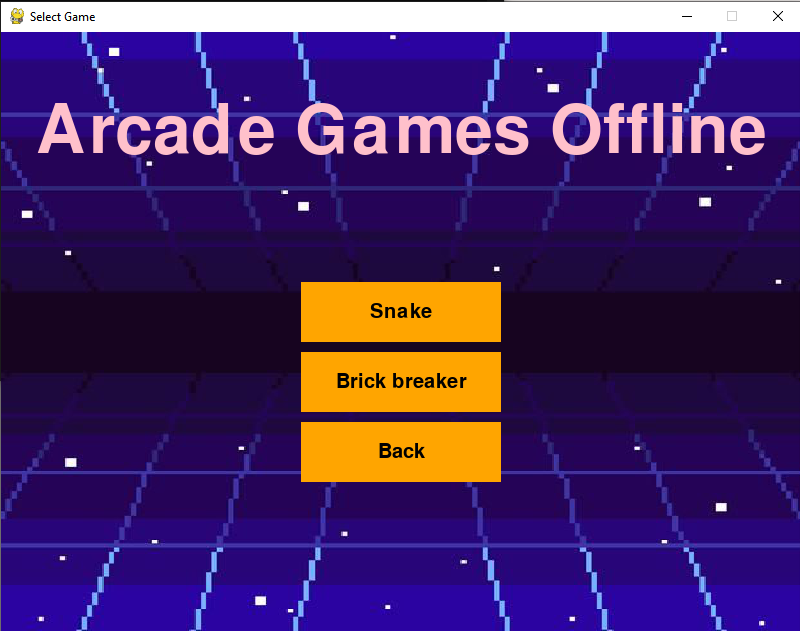
A Pyinstallernek köszönhetően a Python-alkalmazások könnyedén, önállóan futtatható fájlokká alakíthatók, így **minden külső telepítés nélkül működik és használható**! Az elkészült program azonnal futtatható külön konfiguráció nélkül. A játékok a MENU.exe futtatásával érhetőek el.

1. Menü:

A játék főmenüje három alapvető funkciót kínál, amelyek lehetővé teszik a felhasználók számára a játékmód kiválasztását, a beállítások módosítását és a játékból való kilépést.



* Select Gamemode: Ez a gomb lehetővé teszi a felhasználó számára, hogy kiválassza a játék módját**.**

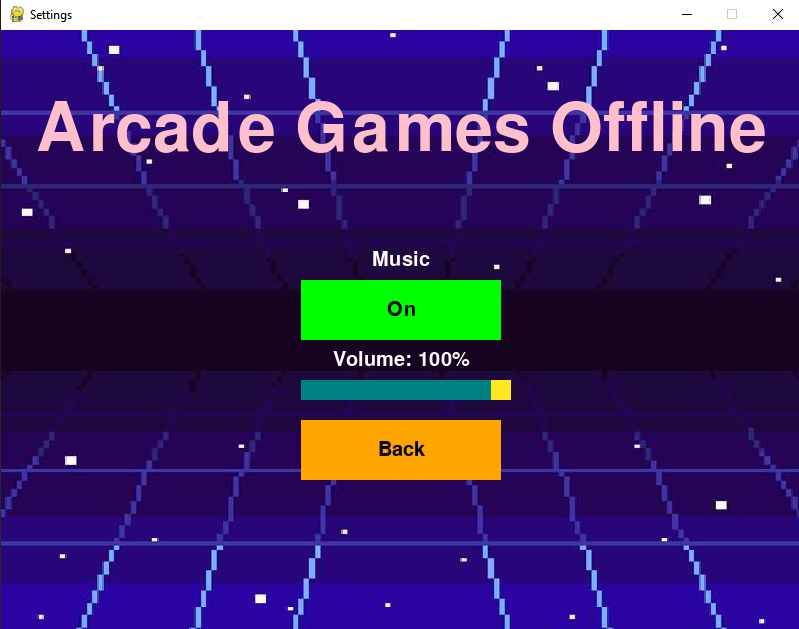
****

A kattintás után egy új képernyő jelenik meg, ahol különböző játékmódok közül választhat:

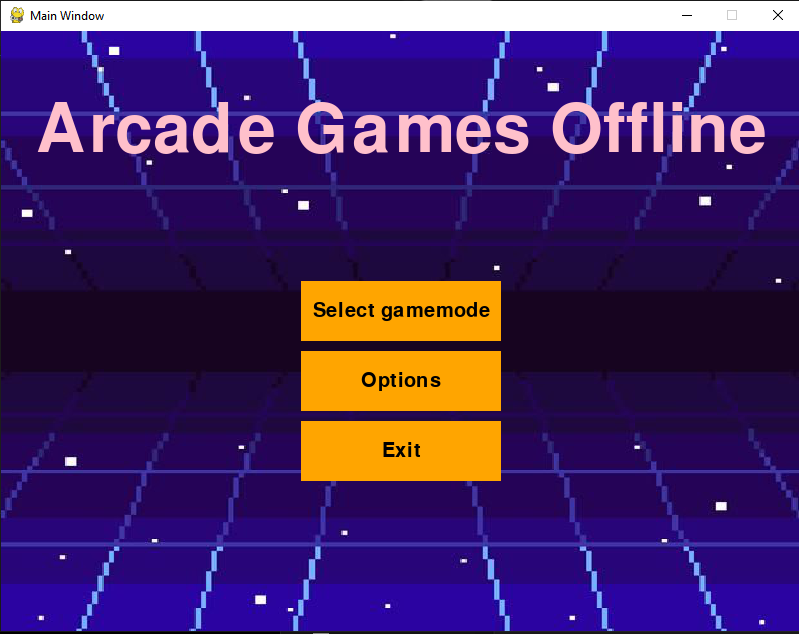
Snake – A klasszikus kígyós játék, ahol a cél minél hosszabbra növeszteni a kígyót anélkül, hogy nekiütközne a saját testének.

Brick Breaker – Egy ügyességi játék, amelyben egy ütővel irányított labdával kell eltörni az összes téglát a pályán.

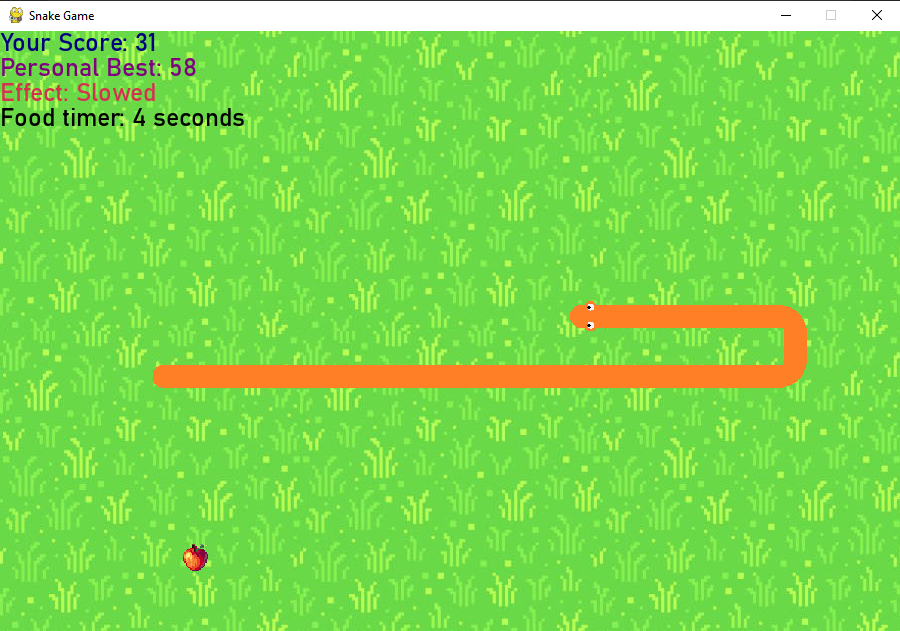
* Az **Options** menü olyan beállításokat tartalmaz, amelyekkel testre szabhatod a felhasználói élményt. Itt lehetőséged van a zene be- és kikapcsolására, valamint a hangerő módosítására egy csúszkával.



* **Exit: Az Exit gomb megnyomásával a felhasználó kiléphet az alkalmazásból.**



1. Snake

****

Ebben a speciális Snake játékban a játékos irányítja a kígyót, amely nem ütközik a falakba, hanem át tud rajtuk haladni, és körbemenni a pályán. A cél továbbra is az, hogy minél hosszabbra nőjön, miközben háromféle ételt fogyaszt: banán, alma és szilva.

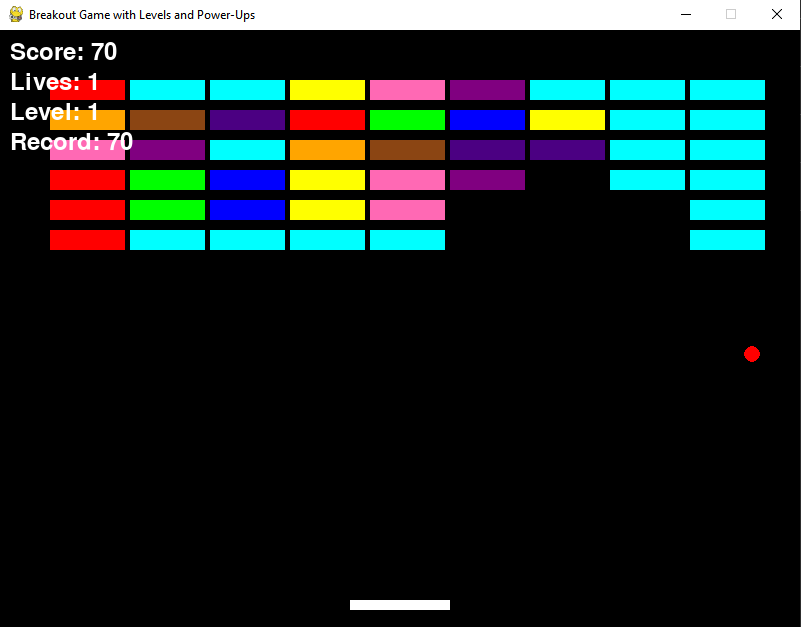
Az **alma** a leggyakoribb 50% eséllyel jelenik meg a pályán és nincs különleges hatása.

A **szilva** a második leggyakoribb étel amelyre pontosan 30% esély van és könnyít a játékon ugyanis 3 másodpercig felezi a haladási sebességet.

A **banán** a legritkább gyümölcs amit 20% eséllyel lehet kapni de értéke miatt nehezíti a játékot ugyanis 3 másodpercig kétszeresére növeli a játék sebességét. Ezeket az “Effect:” tartja számon.

Az ételek változatossága miatt minden gyümölcs csak 5 másodpercig létezik, amit jelez is e utána új gyümölcs jelenik meg. Ezt a „Food timer” jelzi, ezzel segítve a játékosnak tisztában lenni a hátralevő idővel. A játék a **maximális pontszámot** az elfogyasztott ételek alapján számolja.

1. Block breaker



A **Block Breaker** egy klasszikus ügyességi játék, amelyben a játékos egy mozgatható ütővel irányítja a labdát, és célja, hogy minden téglát eltörjön a pályán.

A játék kezdetén a labda elindul, és a falaknak vagy blokkoknak ütközve visszapattan. A játékos balra és jobbra mozgatva az ütőt gondoskodik arról, hogy a labda ne essen le a képernyő aljára, különben életet veszít.

A pontszámítás a játékos teljesítményét tükrözi: minden eltört blokk után pont jár, és a játékos célja, hogy a lehető legmagasabb pontszámot érje el. A játék véget ér, ha minden blokk eltört, vagy ha a játékos elveszíti az összes életét.

A Képernyőn megjelenik az elért pontszám, „Score”, a hátralevő életek “Lives”, a szint, hogy hol tart a játékos “Level” és a “Record” ami a játékos legnagyobb valaha elért pontszámát jelzi.

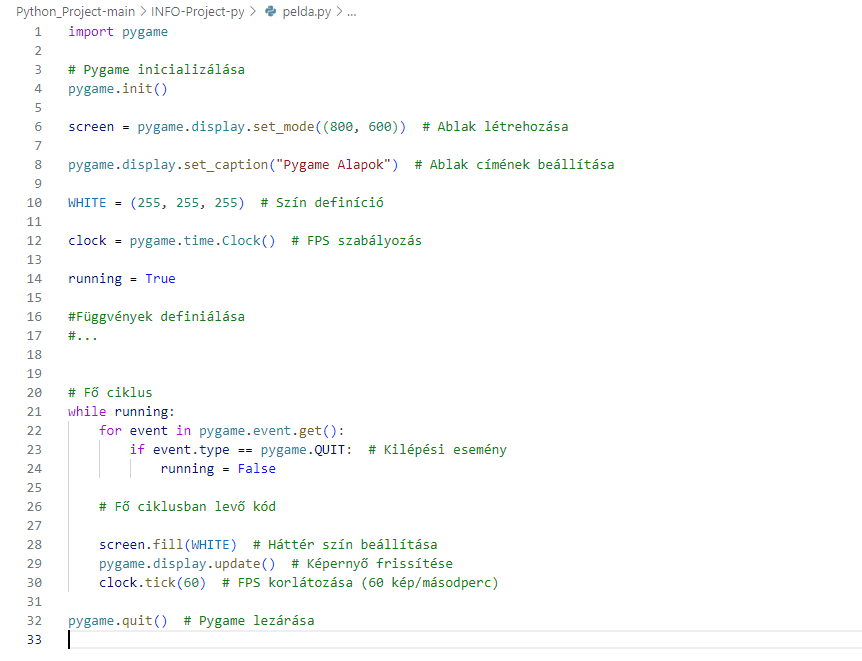
1. Programozói kézikönyv

A játék fejlesztése során „**Python”** volt az alapvető programozási nyelv, amelyben minden logikát, számítást és folyamatot kezeltünk. Segítségével meghatároztuk a játékelemek viselkedését, mint például a kígyó mozgását a „**Snake”** játékban, vagy a labda fizikai interakcióit a „**Block Breaker”** esetében. A Pythonban írt függvények biztosították a főbb funkcionalitást és a logikai részleteket.

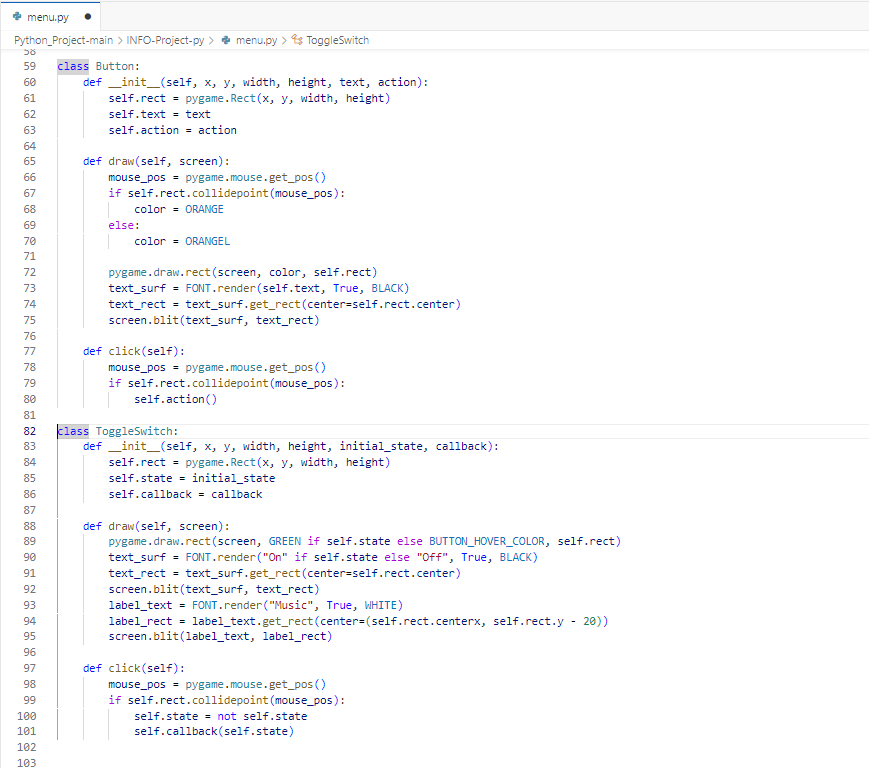
A grafikai megjelenítéshez és interaktivitáshoz „**Pygame**-et” használtunk, egy grafikus importálható ingyenes könyvtár, amely lehetővé tette, hogy megjelenítsük az objektumokat a képernyőn, kezeljük a felhasználói bemenetet, és animáljuk a játék elemeit.

A „Pygame” segített abban, hogy a játék folyamatosan frissüljön és sima mozgásokat hozzunk létre a képkockák kiszámításával. Például a „Snake” játékban a kígyó mozgását irányítottuk, míg a „Block Breaker” esetében a labda visszapattanását és az ütközések érzékelését oldottuk meg.

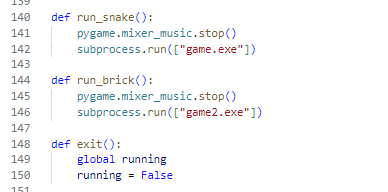
Pygame szintaxisa:



A menü működése:



Classok segítségével megoldottuk a funkcionalitást pélául a gomboknak. Hasonlít a szintaxisa és funkcionalitásában is a struktúrákra de van különbség a kettő között. A Pythonban nem létezik alapból struktúra ezért classokat használtunk.



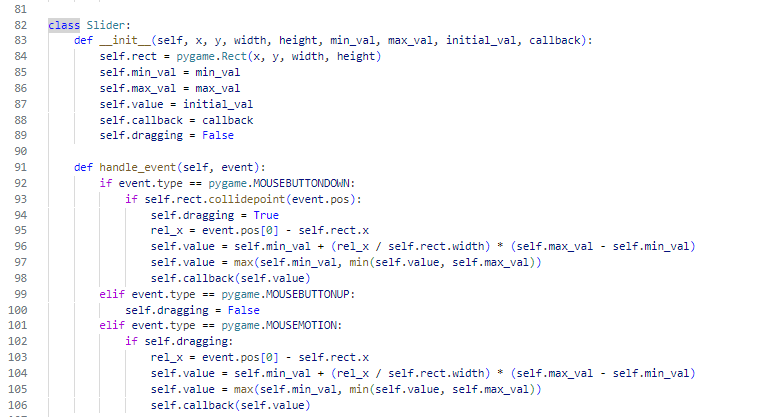
Itt látható a játék meghívásának működése a subprocess.run() függvény segítségével.



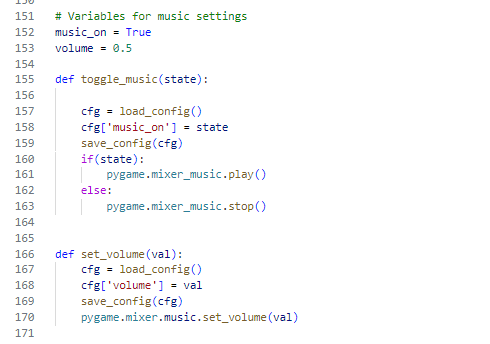
A pygame szintaxis alapján, ez a menü fő függvénye ami segítségével a felhasználó navigálhat a menüpontok között.

A zene és a config:

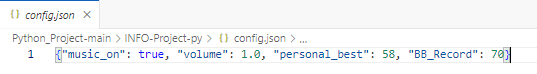
Mindegyik játékban zene játszódik le amit a menüben lehet ki/be illetve a hangerejét is állítani a sliderrel:



Két függvény amivel tároljuk a zenebeállításokat:



Ezeket a beállításokat egy “Config.json” fileba mentjük ki a könnyed tárolhatóság érdekében:



Ezeken kívül ide tároljuk le a rekordszámlálókat a játékokhoz hogy ne vesztődjenek el a játék bezárása után.

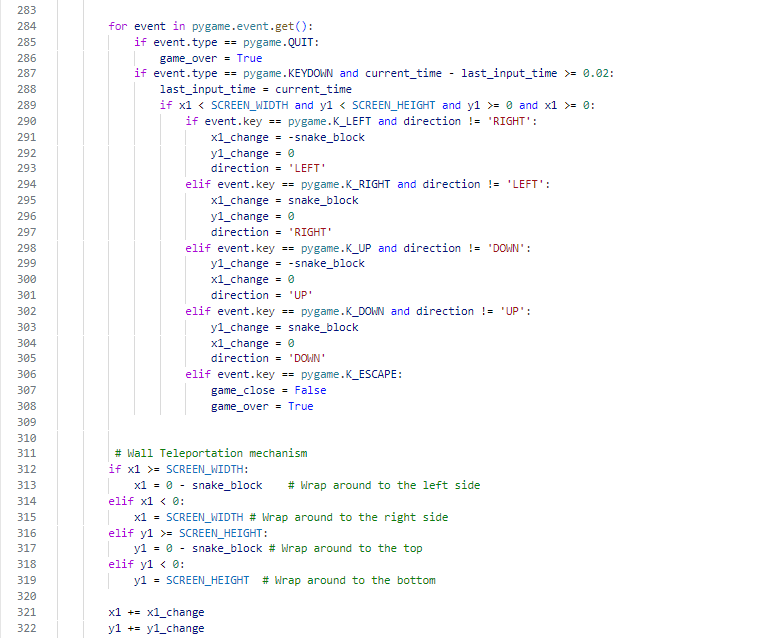
A kígyó grafikai megoldása:



Megtörténik az importálás, utána következik a logika a koordináta rendszer alapján.

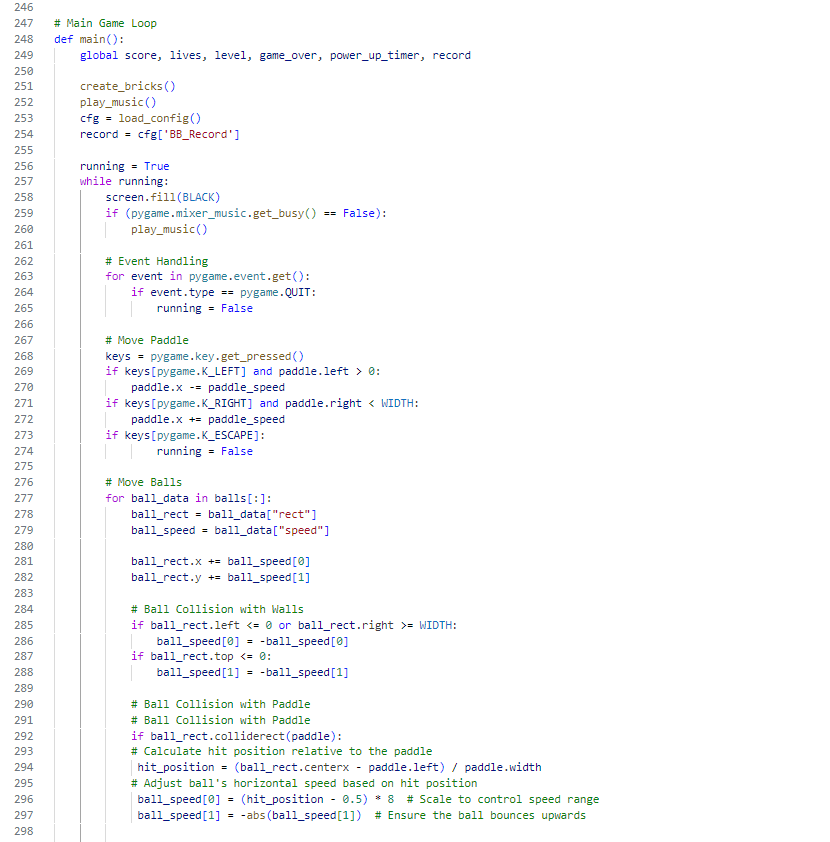


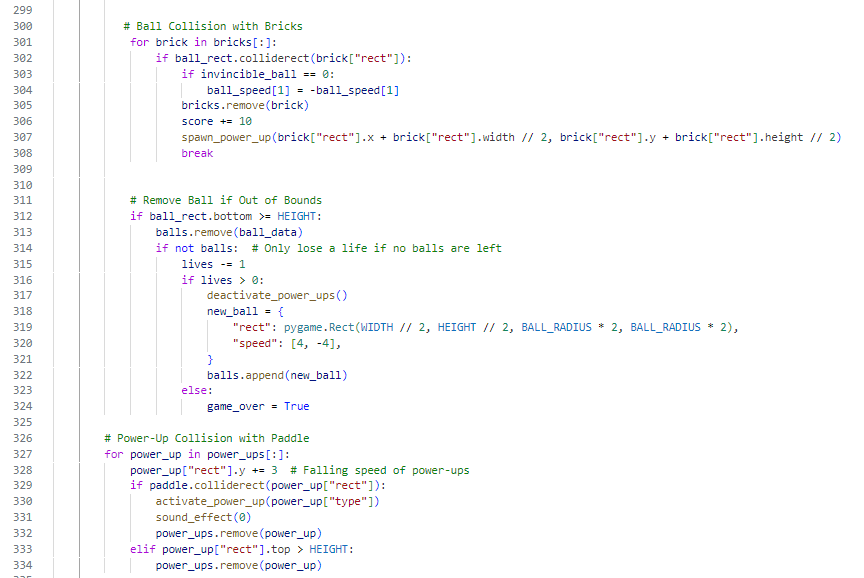
A kígyó mozgási mechanizmusa:



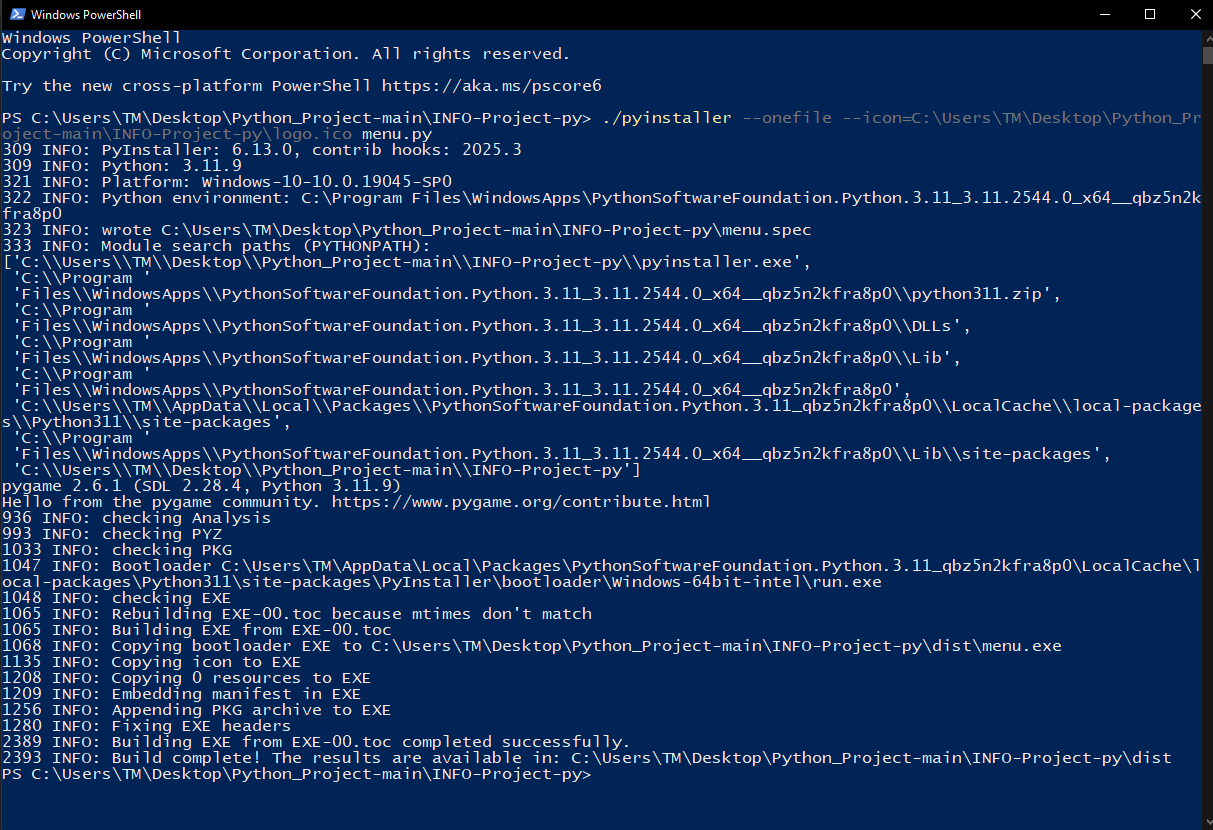
A játék mozgását az FPS-hez kötöttük ami jelen esetben 8 FPS.

A Block breaker fizikája:





A Block breaker esetében a játék 60 fps-en fut, a labda sebességét külön számoljuk.



Végül, amikor készen állt a játék, a **PyInstaller** segítségével csomagoltuk az alkalmazást önálló futtatható fájlba, hogy a felhasználók külső telepítések nélkül játszhassanak. Ez biztosította, így a végleges verzió zökkenőmentesen futtatható lett külön konfigurálás nélkül.

1. Fejlesztési lehetőségek

**Snake-be új játékmódokat lehet beleépíteni** mint például időkorlátos mód, ahol a játékosnak adott időn belül minél hosszabbra kell növelnie a kígyót.

**Ellenfél AI:** Egy másik, automatikusan mozgó kígyó, amely versenyez vagy akadályozza a játékost.

**Dinamikus pályaelemek**: Falak, portálok, vagy mozgó akadályok, amelyek új kihívásokat teremtenek.

**Vizuális Bug kijavítása**: A program jelenlegi verziójában egy ismert vizuális hiba található, amely nagyon szélsőséges estben rekreálható. A további verziókba kijavítás implementálása.

**Block Breaker-be lehetne tenni új blokktípusokat**: Olyan blokkok, amelyek visszalőnek, mozognak vagy időzítve eltűnnek.

**Erősebb bónuszok**: Új ütőméretek, mágneses labda, vagy kettős sebesség.

**Pályakészítő**: Egy szerkesztő, ahol a játékosok saját pályákat tervezhetnek és játszhatnak.

**Többjátékos mód**: Egy kooperatív verzió, ahol két játékos közösen játszik, vagy egy versengő mód, ahol az egyik játékos blokkokat hoz létre, míg a másik eltörli őket.

1. **Szakirodalom**

<https://www.python.org/>

<https://pyinstaller.org/en/stable/index.html>

<https://www.pygame.org/docs/>

<https://docs.python.org/3/tutorial/index.html>

<https://www.w3schools.com/python/default.asp>

<https://pixabay.com/music/>