Snake játék – Projekt Specifikáció

Szoftverfejlesztés MI támogatással – Házi feladat specifikáció

1. Bevezetés

A házi feladat célja egy klasszikus **Snake** (kígyós) játék megvalósítása Python nyelven, a **pygame** grafikus könyvtár felhasználásával. A játékos egy folyamatosan mozgó kígyót irányít a pályán, amely minden alkalommal hosszabbodik, amikor élelmet fogyaszt. A cél a lehető legmagasabb pontszám elérése anélkül, hogy a kígyó falnak vagy saját testének ütközne.

A rendszer két fő komponensből áll: a **backend**ből, amely a játékszabályokat és az állapotkezelést tartalmazza, valamint a **front-end**ből, amely a pygame alapú grafikus megjelenítésért felel. A projekt során mesterséges intelligencia eszközöket (**ChatGPT, GitHub Copilot**) használok, és ezek alkalmazását a **README** fájlban dokumentálom.

2. Funkcionális követelmények

2.1 Alapfunkciók

- A játék grafikus ablakban indul.
- A kígyó a négy irány egyikébe mozgatható.
- A kígyó minden lépésben egy cellát mozdul.
- A rendszer véletlenszerű pozícióban ételt generál.
- Étel megevésekor nő a pontszám és a kígyó hossza.
- Fal- vagy testütközés esetén a játék véget ér.
- A játék újraindítható a game over képernyőről.

2.2 Speciális funkciók

- Három nehézségi szint különböző sebességekkel.
- A pontszám folyamatosan megjelenik.
- Game over képernyő megjelenítése.
- Kezdőmenü és állapotkezelés.

2.3 Nem funkcionális követelmények

- Stabil, hibamentes működés.
- Moduláris, karbantartható kódszerkezet.
- Egyszerű és áttekinthető grafikus megjelenés.
- Python 3.10+ kompatibilitás.

3. Szerepkörök

A rendszer egyetlen szereplője a Játékos, aki irányítja a kígyót, kiválasztja a nehézségi szintet, és kezeli az újraindítást.

4. Forgatókönyvek

4.1 Játék indítása

- A kezdőmenü megjelenik.
- A játékos kiválasztja a nehézségi szintet.
- A backend inicializálja a játékállapotot.
- Elindul a játék.

4.2 Kígyó mozgása

- A játékos billentyűzettel változtat irányt.
- A backend frissíti a kígyó pozícióját.
- A frontend kirajzolja a mozgást.

4.3 Ételfogyasztás

- Backend ütközésvizsgálatot végez.
- A kígyó hossza nő.
- A pontszám növekszik.
- Új étel jelenik meg.

4.4 Ütközések

- Fal- vagy testütközés → game over.
- A frontend megjeleníti a game over képernyőt.

4.5 Újrakezdés

- A játékos újraindítja a játékot.
- A backend teljesen reseteli az állapotot.

5. Rendszerarchitektúra

5.1 Backend

- SnakeState: játékállapot-kezelés
- Snake modell: a kígyó testének tárolása
- Food modul: ételgenerálás
- Collision modul: ütközésvizsgálat
- Difficulty: sebességprofilok

5.2 Frontend

- Pygame ablak kezelése
- Input-kezelés
- Játékelemek kirajzolása
- Game loop vezérlése
- Menük és képernyők megjelenítése

5.3 Backend–frontend kommunikáció

A frontend minden ciklusban meghívja a backend update() függvényét, majd az ennek eredményeként kapott állapotot kirajzolja. A két komponens így jól elkülönül, de folyamatosan együttműködik.

6. Technológiai indoklás

Python: gyors fejlesztés, könnyen olvasható szintaxis. Pygame: egyszerű 2D grafikai library, ideális oktatási projektekhez. MI-eszközök: gyorsabb fejlesztés, refaktorálás és dokumentációkészítés támogatása.

7. Mappastruktúra

```
/snake_game
 ├── backend/
 │    ├── state.py
 │    ├── snake.py
 │    ├── food.py
 │    ├── collision.py
 │    └── difficulty.py
 ├── frontend/
 │    ├── renderer.py
 │    ├── input_handler.py
 │    └── game_loop.py
 ├── assets/
 ├── main.py
 └── README.md
```

8. Összegzés

A projekt célja egy jól strukturált, backend–frontend elválasztású Snake játék megvalósítása Python és pygame segítségével. A specifikáció részletezi a rendszer összes fontosabb követelményét, a szereplőt, a fő forgatókönyveket, az architektúrát és a technológiai indoklást. A fejlesztés során végig alkalmazom a tárgy által megkövetelt MI-eszközöket, amelyeket a dokumentációban átláthatóan rögzítek.