

Snake Multiplayer 2018 – Dokumentáció



1. Feladatkiírás

Készíts kígyó játékot, ahol a kígyók, ha megeszik a véletlenszerűen elhelyezett étet, akkor nő a méretük. Ha önmaguknak, vagy egymásnak ütköznek, az a játék végét jelenti.

A program legyen képes:

- egy, illetve két játékos üzemmódra
- a pontok számolására és elmentésére

2. Pontosított Specifikáció

A program célja

SDL könyvtár felhasználásával, tehát grafikusan kivitelezni egy és kétszemélyes, valósídejű kígyós (Snake Multiplayer) játékot, mely rendelkezik eredményjelzővel. A kivitelezés módszerét a program fejlesztése során kell majd kitalálni.

A program használata

A felhasználónak a program indítását követően menüből ki kell választani, hogy egy- vagy kétszemélyes játékot szeretne játszani, lehetősége van a rendelkezésre álló pályákból választani. A kód elkészítése során felhasználásra kerül dinamikus memóriakezelés és láncolt lista használat az adott kígyó méretét és a játéktérben szereplő helyét illetően.

A futás eredménye

A játéknak akkor van vége, ha a kígyó falnak, önmagának, vagy másik kígyónak ütközött. A játék vége után a szerzett pontok eredményjelzőn rögzítésre és megtekintésre kerülnek, mely egy `results.txt` fájlban (Fájlkezelés) lesz megtalálható.


A `results.txt` fájl felépítése:

Eddig játszott játékok +1	Első játékos: Pontszám	Második játékos: Pontszám
Eddig játszott játékok +2	Első játékos: Pontszám	Második játékos: Pontszám
...

3. Programozói Dokumentáció



Megvalósított módszerek áttekintő magyarázata

Code::Blocks – ban a játék Debug módban, a „Build and Run”  gombra kattintva indul.

A program egy C környezetben fejlesztett SDL2 Grafikus program, melyben sor kerül időzítők használatára, eseményvezérelt programműködésre, szövegmegjelenítésre, billentyűzet kezelésre valamint képfájlok beolvasására. A játékban található kígyók, falak, a pályára random kerülő ételek egy irányba láncolt listában vannak eltárolva. A kígyó mozgása úgy lett megvalósítva, hogy a vezérlő tagja (NULL-ra mutató utolsó tag), kapja meg a felhasználótól érkező irányítást, és a többi tag pedig ezt az elmozdulást követi. A program egyik fő működési elve az ütközésellenőrzés megvalósítása. Megvizsgálásra kerül kígyó-fal, kígyó önmagával, kígyó-kígyó és kígyó-étel ütközések ellenőrzése, mely láncolt listák szisztematikus bejárásával valósult meg. A program a segédletként felhasználtató Debugmalloc programmal ellenőrizve lett. Az ellenőrzés során a kódban memóriaszivárgásra utaló hiba nem volt található.

Adatszerkezetek választása

A program működésének több főszereplője van. Az első egy menüválasztó struktúra, mely horizontális és vertikális elmozdulással tárolja a felhasználó által nyílbillentyűkkel léptetett, azaz kiválasztott megfelelő értéket.

```
typedef struct todo{  
    int vertical;    //Choosing Menu Status  
    int horisontal; //Choosing Map  
}todo;
```

Ez a food névre hallgató struktúra a kódban láncolt listaként kerül majd felhasználásra. Célja a pályára kerülő random ételek eltárolása.

```
typedef struct food{  
    int x,y;  
    struct food *next;  
}food;
```

A pályák megrajzolásakor a pálya falai egy irányba láncolt listában vannak eltárolva. A pályát 20x20-as négyzetrácsok alkotják, és kizárólag ezeken lehet fal, étel vagy kígyót alkotó grafikus kijelzés.

```
typedef struct wall{  
    int x0,x1,x2,x3;  
    int y0,y1,y2,y3;  
    struct wall *next; ///One Direction  
}wall;
```

A kígyók irányításáért ez a struktúra felel. Célja bool típusként eltárolni a felhasználó által leütött billentyűt.

```
typedef struct control{
    bool up, left, down, right;
}control;
```

A legfontosabb struktúrák egyike a következő, mely a kígyó testének eltárolásáért felelős. Az adattárolás egy irányba láncolt listában történik. Tartalmazza a tagok elmozdulási irányát és a kirajzolás koordinátáit, melyek egy 20x20-as négyzet csúcsait jelölnék ki.

```
typedef struct snake{
    bool dir_u, dir_d, dir_l, dir_r;
    int x0,x1,x2,x3;
    int y0,y1,y2,y3;
    struct snake *next;
}snake;
```

A program működését vezérlő fő függvények

A main() függvényből meghívott start menüt elindító függvény.

```
sdl_start_menu();
```

A controls menüpont kirajzolásához szükséges függvény.

```
void show_controls();
```

Az értékjelző kirajzolásához szükséges függvény.

```
void show_scoreboard();
```

A játékon belüli, menü újrarajzolásához szükséges függvény

```
void draw_all();
```

A resulst.txt tartalmát beolvasó és tömbbe helyező függvény.

```
void linestoaarray(char* filename, int lines, int* tomb);
```

Ebből a függvényből hívódik meg a játékmenetért felelős függvények jelentős része.

```
void map0(int users);
```

Ez a függvény rajzolja ki a pályákat. Három másik hasonló nevezéktannal ellátott testvérével a pályát rajzolják ki.

```
wall *draw_map0();
```

A falat egy irányú láncolt listába helyező függvény.

```
wall *draw_map_injection(wall *begin, wall adat);
```

A pályát kirajzoló függvény, mely láncolt listából építkezik.

```
void generate_map0(wall *wall0);
```

Az étel random elhelyezéseért felelős függvény, mely kizárólag úgy helyez el random ételt, hogy az a 20x20-as rádspontokra kerüljön és ne ütközzék aktuálisan falba, vagy kígyóba.

```
food* generate_food(wall *wall0, snake *snake_one);
```

Ütközést vizsgáló függvény kígyóval és fallal.

```
bool collision_detection_wall(snake *snake_one, wall *wall0);
```

Egyjátékos mód esetén a results.txt fájlba írja a játék végeredményét. Párja kétjátékos mód esetén teszi ugyanezt.

```
void single_player_scoretofile(snake *snake_one);
```

A legfontosabb függvények egyike. Ez lépteti előre folyamatosan a kígyókat, adja át a tagoknak az előző tag koordinátáit, mellyel megvalósítja a léptetést. A boss változó vezér módjára adja meg a kígyó irányát.

```
void control_snake(snake *snake_one, control boss);
```

4. Felhasználói Dokumentáció

Ez a „*Snake Multiplayer*” játék arra való, mint ahogy a neve is mutatja, hogy egyedül vagy partnerrel szórakoztatóan lehessen időt eltölteni.

A program indítását követően a nyílbillentyűk segítségével lépkedhetünk a menüben. A fel és le gombokkal választhatunk „*Single Player*” illetve „*Multi Player*” üzemmód között, megtekinthetjük a „*Scoreboard*”-ot, és a „*Controls*” menüpont alatt tesztelhetjük az irányítást. Az épp aktívan kiválasztott menüpont piros szövegháttérrel rendelkezik, észrevehető hogy a menüben való léptetéssel a piros színezés változik.

A balra és jobbra nyílbillentyűkkel van lehetőség a pálya választására. Minden pályához különböző háttér és pályafelépítés tartozik. A menüből való kilépésre az F1 billentyű megnyomásával van lehetőség.

Amennyiben a balra és jobbra mutató nyílbillentyűk segítségével kiválasztásra került az adott pálya, illetve a fel és le nyílbillentyűkkel vagy a „*Single Player*” vagy a „*Multi Player*” menüpontok egyikén vagyunk, úgy értelemszerűen egy Enter lenyomásával elindul a játék.

A játéknak akkor van vége, ha a kígyó saját magának, falnak vagy többszemélyes játék esetén a másik kígyónak ütközött. A játék közben kilépésre való lehetőséget szintén az F1 gomb megnyomása szolgáltatja. Ha a játék véget ért, akkor megjelenik a „*Scoreboard*” menüpont által is elérhető eredményjelző. A „*Scoreboard*” menüpontból az F1 gomb megnyomásával van lehetőség.

Az eredményjelzőbe a results.txt fájlból kerülnek beolvasásra az adatok, ahol rendre szerepel a lejátszott játékok száma, valamint hogy az adott kígyó mennyi random elhelyezett ételt szerzett meg. Ha csak egy játékos üzemmódban játszottunk, akkor a második játékos pontszáma automatikusan 0.

A „*Controls*” menüpontban észrevehető, hogy az első és egyben bal oldali játékos a W-A-S-D billentyűk, míg a jobb oldali játékos a 8-4-5-6 numerikus billentyűk segítségével mozdul el a pályán. A kígyók első alapértelmezett elmozdulása a képernyő felső része felé történik. Ahogy a kígyó feje ételhez ért és megszerezte azt, úgy testmérete növekszik. A „*Controls*” menüpontból való kilépésre az F1 billentyű megnyomásával van lehetőség.

Lévai Gábor
2018 November 8,
Budapest

