Vass Annamária

Snake – Programozói dokumentáció

A játékom az SDL grafikus könyvárat használja megjelenítésre, tehát a kód fordításához szükséges ennek a telepítése. A grafikus megjelenítés során egyetlen ablakot használok, és a megjelenítő segítségével rajzolom ki rá a megfelelő elemeket.

A játék lényegi részét duplán láncolt listával valósítottam meg, így lehetséges, hogy megnő a kígyó ha megeszi az almát. A dinamikus memóriafoglalás és felszabadítás helyességének ellenőrzésére a ***"debugmalloc.h"***-t használtam.

A programban ezen kívül van képbetöltés az SDL_Image.h könyvtár segítségével, valamint file-ból olvasás és abba való írás a dicsőséglista és a help ablakok megjelenítésénél.

A programhoz tartozó file-ok:

- ***"leaderboard.txt"*** – Dicsőséglista: Tartalmazza az eddigi játékosok közül a legjobb 5 nevét, pontszámát. A fájlban egyik sorban van a játékos neve, az utána következő sorban (sorvége jellel elválasztva) pedig a hozzá tartozó pontszám.
- ***"help.txt"*** – Tartalmazza a *Help* meghívásakor megjelenítendő szöveget, amelyben a felhasználó számára röviden le van írva, hogy hogyan működik a játék. A szöveg egyes sorai sorvége jellel vannak elválasztva, a legvégén pedig egy end szócska található, külön sorban, amely jelzi, hogy vége van a kiírandó szövegnek. A program ezt már nem jeleníti meg.
- ***"snake.png***" – A menübe beillesztendő kép
- ***"leaderboard.png***"- A dicsőséglistára illesztendő kép
- ***"gameover.png***"- A létjátékosmód végére illesztendő kép.
- ***"LiberationSerif-Regular"***-A név beolvasásánál használt font.

A program felépítése:

A programomban 6 darab modul található, emellett pedig a ***"main.c"*** és a ***"debugmalloc.h"*** memóriakezelést ellenörző modul.

1. ***"main.c"***- Létrehozza a grafikus megjelenítéshez szükséges ablakot és megjelenítőt. A menüt vezérli, a program indításakor meghívja a menüt megjelenítő függvényt, és kezeli a menügombok lenyomását, például ha a felhasználó rákattint a "*singleplayer"* gombra, akkor meghívja az annak megfelelő függvényt. A *valaszt* logikai változó vizsgálja, hogy a felhasználó választott-e már a menüből, ha igen, akkor az adott művelet befejeztéig ne menged újabb választást. A *quit* változóban tárolódik, hogy fut-e a program, vagy a felhasználó rányomott a kilépés gombra. Kilépés esetén megsemmisíti az ablakot, a megjelenítőt és a betöltött képeket.
2. ***"menu.h"*** és ***"menu.c"***-Tartalmazza az ablak létrehozásához szükséges függvényt, valamint azokat a függvényeket, amiket a menüből választásnál használunk, kivétel a singleplayer és multiplayer függvényeket, mert azokat külön modulba szerveztem a könnyebb áttekinthetőség érdekében.
3. ***"singleplayer.h"*** és ***"singleplayer.c***"-Az egyjátékosmód megjelenítéséhez és vezérléséhez szükséges függvényeket tartalmazza, valamint a használt struktúrákat.
4. ***"multiplayer.h"*** és ***"multiplayer.c"***-A kétjátékosmód megjelenítéséhez és vezérléséhez szükséges függvényeket tartalmazza.
5. ***"commongraphics.h"*** és ***"commongraphics.c"*** – Azokat a függvényeket tartalmazza, amelyeket a singleplayer és multiplayer modulok közösen használnak.
6. ***"apple.h"*** és ***"apple.c"***-A játék közben az alma (étek) kezeléséhez szükséges struktúrák és függvény található benne.
7. ***"snake.c "*** és ***"snake.h"***-A játékbeli kígyóhoz tartozó struktúrák, függvények, amik különböző funkciókat valósítanak meg, pl. átpozícionálják a kígyót a mozgása során.

Az egyes modulok struktúráinak és függvényeinek leírása:

1. "***menu***" modul:
   i) Függvények:
   - void sdl_init(char const *felirat, int szeles, int magas, SDL_Window **pwindow, SDL_Renderer **prenderer)

Ez a függvény inicializálja az SDL ablakot és a megjelnítőt, a paraméterei tartalmazzák az ablak méreteit, valamint a megjelenítendő feliratot az ablak fejlécében.

- void menu_show(SDL_Renderer *renderer, Button *buttons, SDL_Rect *pic_rect, SDL_Texture* pic)

Ez a függvény jeleníti meg a menüt, kirajzolva annak gombjait és a "snake.png" képet amely a menüt díszíti.

- void Leaderboard_show(SDL_Renderer *renderer,int score ,SDL_Texture *leaderboard_img)

Ez a függvény jeleníti meg a dicsőséglistát, azáltal, hogy a ***"leaderboard.txt"*** fájlból kiolvassa azt, és kiírja a képernyőre. Ha a *score* paraméternek -1-et adunk, akkor tudjuk, hogy a menüből volt meghívva a függvény, ellenkező esetben pedig a *singleplayer* mód végén, ilyenkor a függvény ki is írja az aktuális játékos pontszámát is. A *"leaderboard_img"* tartalmazza a betöltött ***"leaderboard.png"*** képet, amely a dicsőséglistát díszíti.

- void help_show(SDL_Renderer *renderer)

Ez a függvény jeleníti meg a "*Help"* menügomb lenyomása után a felhasználó számára a rövid útmutatót a játékhoz. A program ezt a ***"help.txt"*** nevű fájlból olvassa ki, az 'end' végszóig.

2. ***"singleplayer"*** modul:

   i) Struktúrák/felsorolt típusok:

- A "Game_mode" felsorolt típusban a játékmód lehetséges változatai találhatóak az egyjátékos módon belül: *freemode*-a játékban nincsenek falak, a kígyó a képernyő egyifeléről át tud menni a másikra; *wallmode*-a pálya körul fal van.

```
typedef enum Game_mode
{
    freemode,wallmode
} Game_mode;
```

- A "Player" struktúrában összetartozó adatok találhatóak, egy játékos neve és pontszáma. Ezt a struktúrát a dicsőséglistánál használom, így könnyen kezelhetem egyben a játékosok adatait.

```
typedef struct Player
{
   char name[50];
   int score;
} Player;
```

ii) Függvények:

- void color_picker_show(SDL_Renderer *renderer,Button* colors,Button* gamemodes,Button* startbutton)

Az egyjátékosmód elején található szín- illetve játékmód('*free*' vagy '*wall*') választót kirajzoló függvény. Paraméterül a megjelnítőt és a különböző gombokat kapja, amiket kirajzol. A szín- és játékmód választás egészen addig lehetséges, amíg a felhasználó el nem indítja a játékot a *start* gombbal.

- void color_picking_show(SDL_Renderer *renderer,Button *colors, Button* clicked)

Ez a függvény jeleníti meg az éppen kijelölt színválasztást. Ha a felhasználó rákattint valamelyik gombra, akkor a függvény szín radiobutton-ját feketére színezi, hogy látható legyen mi van éppen kiválasztva. Paraméterként megkapja a gombok tömbjét, és azt a gombot amelyikre rákattintottak.

- void gamemode_picking(SDL_Renderer *renderer,Button *modes, Button* clicked)

Ez a függvény jeleníti meg az éppen kijelölt játékmódválasztást, a kiválasztott mód radiobutton-ját feketére színezi.

- void singleplayer_show(SDL_Window *window, SDL_Renderer *renderer, Button* singleplayer_buttons , Snake snake, Apple apple)

Az egyjátékosmód játékpályáját és a játékot irányító gombokat kirajzoló függvény. Emellett ez a függvény jeleníti meg kezdetben kígyót és az almát is a képernyőn, ezeket az elemeket kapja meg paraméterül.

- void wall_show(SDL_Renderer *renderer)

Ez a függvény "*wall*" játékmód esetén kirajzolja a falat a pálya köré.

- void TTF_Font_init(TTF_Font **pfont, char const *tipus)

Ez a függvény inicializálja a játékos nevének beírásához a fontot.

- bool input_text(char *dest, size_t hossz, SDL_Rect teglalap, SDL_Color hatter, SDL_Color szoveg, TTF_Font *font, SDL_Renderer *renderer)

Ez a függvény teszi lehetővé, hogy a játékos beírhassa a nevét, amiután vége a játéknak.

Forrás: infoc

- void end_of_game_singleplayer(SDL_Renderer *renderer,int score)

Ez a függvény hívódik meg, amikor a játékos meghal, itt kérődik be a játékos neve. A függvény a ***"leaderboard.txt"*** fájlból kiolvassa az eddigi legjobb 5 játékost, megnézi, hogy az éppen most végzett játékosnak fel kell-e kerülnie a listára, ha igen beszúrja a megfelelő helyre, majd frissíti a ***"leaderboard.txt"*** fájlt. A legvégén meghívja a dicsőséglistát megjelenítő függvényt, amelyben írja a játékos pontszámát is a *'Your score'* felírat mellett, ezért kell paraméterül megadni a score változó értékét, amely a játékos pontszámát tárolja.

- void singleplayer_mode(SDL_Renderer *renderer)

Ez a függvény vezérli az egyjátékos módot, meghívja a szín- és játékmód megjelenítő és választó függvényeket, eltárolja a választást és az alapján indítja el a játékot. Kiírja a kígyó aktuális pontszámát, időzítővel mozgatja a kígyót a pályán, és figyeli, hogy meghalt-e a játékos, vagy esetleg megnyomta-e valamelyik játékgombot, valamint érzékeli a billentyűzeten található nyilak megnyomását, ezáltal beállítja, hogy melyik irányba mozogjon a kígyó. Itt számolódik a játékos pontszáma is.

3. ***"multiplayer"*** modul:

ii) Függvények:

- void multiplayer_show(SDL_Renderer *renderer,Button* multiplayer_buttons, Snake snake1, Snake snake2, Apple apple)

Ez a függvény jeleníti meg kétjátékos módban a játékpályát, valamint a játékhoz tartozó gombokat. Kirajzolja a pályára az almát és a kígyót is, a *'beginning_multiplayer'* kezdőállapotuk alapján.

- void end_of_game_multiplayer(SDL_Renderer *renderer,int score1, int score2)

Ez a függvény hívódik meg amikor az egyik játékos meghal, kiírja, hogy vége a játéknak a ***"gameover.jpg"*** megjelenítésével, majd megjeleníti a nyertes játékos nevét, és vár a kilépésre, ami a *close* gomb megnyomásával történik meg.

- void multiplayer_mode(SDL_Renderer *renderer)

Ez a függvény irányítja tulajdonképpen a multiplayer módot, meghívja a pályát és a gombokat kirakójzoló függvényeket. Érzékeli a játékgombok lenyomását, és azt is, ha a játékosok a billentyűzet segítségével megváltoztatják a kígyóik haladási irányát. Egy időzítő segítségével folyamatosan mozgatja a kígyókat a nekik megfelelő irányba. Érzékeli ha egyik játékos kígyója saját magának ütközik, vagy ha egymásnak ütköznek a kígyók, ilyenkor pedig meghívja a játék végét megjelenítő függvényt. Ha az egyik kígyó saját magának ütközik, annak a pontja nullázódik, ezzel jelezve az *end_of_game_multiplayer* függvénynek, hogy ő a vesztes játékos.

4. ***"commongraphics"*** modul:

i) Struktúrák/felsorolt típusok:

- typedef enum zone {ZONE_X=50,ZONE_Y=50,ZONE_H=500,ZONE_W=900} zone;

A játékpálya méreteit és pozícióját tartalmazó felsorolt típus.

- typedef enum Button_Type

  {

singleplayer, multiplayer, help, leaderboard,close, newgame, pause, exitgame, none, free_b,wall_b,start
} Button_Type;

A gombok fajtájit tartalmazó felsorolt típus.

- typedef struct Button
  {
  SDL_Rect rect;
  Color color;
  } Button;

Egy gomb adatait tartalmazza: az alapjául szolgáló téglalapot, a színét.

ii) Függvények:

- Uint32 idozit(Uint32 ms, void *param)

A singleplayer és multiplayer módokban használt SDL időzítő.

- bool Button_Click(Button *button, SDL_Event *ev)

Ez a logikai függvény megmondja, hogy egy gombra történt az egérkattintás (true), vagy nem (false).

- void Button_Draw(SDL_Renderer *renderer, Button *button, Button_Type type)

Ez a függvény megrajzol egy gombot, és kiírja a típusának megfelelő szöveget rá.

- void score_show(SDL_Renderer *renderer,int score,int x)

Ez a függvény kiírja a sztringgé alakított ponszámot a képernyőre, a megadott *x* koordinátájú helyre. Az előző pontszámot kitörli a képernyőről.

- void gamezone_show(SDL_Renderer *renderer,Button* singleplayer_buttons)

Ez a függvény kirajzolja a játékpályát és a hozzá tartozó gombokat.

- void snake_redraw(SDL_Renderer *renderer,Snake *snake)

Ez a függvény rajzolja újra a kígyót a pályán, azáltal, hogy kitörli a régi helyéről, majd az új pozíciójában kirajzolja azt a farkával együtt, utóbbit egy ciklus segítségével, amely végigmegy a láncolt lista összes elemén és új pozíciójukra rajzolja ki őket.

5. ***"apple"*** modul:

   i) Struktúrák/felsorolt típusok:

- typedef struct Point
  {
    int x,y;
  } Point;

Egy pont x és y koordinátáit tartalmazza.

- typedef struct Apple
  {
    Point position;
    int r;
  } Apple;

Az alma adatait tartalmazza: a pozícióját és a sugarát.

   ii) Függvények:

- void reposition_apple(SDL_Renderer *renderer, Apple *apple)

Ez a függvény kitörli az almát a régi helyéről és új, random helyre rakja az almát, miután a kígyó megette azt.

6. ***"snake"*** modul:

   i) Struktúrák/felsorolt típusok:

- typedef struct Color
  {
    Uint8 r,g,b,a;
  } Color;

A színek könnyebb kezeléséért létrejött struktúra, amleyben az *r,g,b,a* komponensek találhatóak

- typedef struct Size
  {
    int w, h;
  } Size;

Egy téglalap szélességét és magasságát tartalmazó struktúra.

- typedef struct Tail
  {
    Color color;
    Point position;
    struct Tail *next,*pervious;
  } Tail;

A kígyó farkának adatait tartalmazó struktúra, amelyben eltárolódik hogy a kígyó farkát alkotó négyzetek éppen melyik pozícióban vannak, valamint a színük. A struktúra utolsó adattagja tartalmazza az adott farokelem mellett, a láncolt listában kétoldalt található farokelemekre mutató pointereket.

- typedef struct Snake
  {
    Direction dir;
    Point position;
    Size ssize;
    Color color;
    Tail *tail_first, *tail_last;
  } Snake;

A kígyó tulajdonságait tartalmazó struktúra: irány, helyének koordinátái, mérete, színe. A struktúra utolsó tagja a kígyó farkát tartalmazó duplán láncolt lista elejére és végére mutató pointerek, tehát ez a típus tulajdonképpen a "kígyó feje", amit a játék során követ a farka is.

- typedef enum Direction
  {
  up,down,right,left
  } Direction;

Az irányokat tartalmazó felsorolt típus, mivel az iránynak más értékei nem lehetnek.

- enum {V=10}

A sebességet megadó felsorolt típus.

ii) Függvények:

- void beginning(Snake *snake, Apple *apple)

Ez a függvény beállítja a kezdőállapotot: a kígyó alaphelyzetét, irányát, színét, az almát elhelyezi egy random helyre, kiírja a kezdeti pontszámot. Előállítja a kígyó kezdeti hosszúságú farkát, dinamikusan foglal memóriát, létrejön a láncolt lista, amely a kígyó farka.

- void beginning_multiplayer(Snake *snake1, Snake *snake2, Apple *apple)

Ez a függvény a multiplayer játékmódhoz állítja elő a kezdőállapotot, a két kígyó színét, helyzetét, irányát, az alma random kezdeti helyét, valamint mindkét kígyóhoz létrehoz egy-egy láncolt listát ami a kígyók farkait tartalmazza.

- void add_tail(Snake *snake)

A kígyó farkát tartalmazó duplán láncolt lista végéhez hozzáfűz mégegy farokelemet. A játék végén a hívónak fel kell szabadítania a dinamikusan lefoglalt memóriát!

- void free_snake(Snake *snake)

Amikor a játékos meghal, vagy kilépünk a játékból, ez a függvény felszabadítja a kígyó farkának dinamikusan lefoglalt láncolt listát.

- void left_screen(Snake *snake)

Ez a függvény ellenőrzi, hogy a kígyó kiment-e a játékterületről, és ha igen akkor a pálya ellentétes oldalára helyezi a kígyót (a 'free' modeban használjuk, amikor nincsen fal a pálya körül.

- bool snake_collides_with_wall(Snake *snake)

Ez a logikai típusú függvény ellenőrzi, hogy a kígyó nekiütközött-e a falnak a *'wall'* játékmódban. Ha a kígyó nekimegy a falnak a függvény true-val tér vissza, ellenkező esetben false-al.

- void snake_move(SDL_Renderer *renderer,Snake *snake)

Ez a függvény mozgatja egy sebességnyi egységgel (*V*) a kígyót a megadott irányba, és használja a *left_screen* függvényt, amelyik vissszarakja a kígyót a játékterületre, ha elhagyja azt. A farkának mozgását úgy oldja meg, hogy az egyes farokelemeket mindig az előző farokelem helyére rakja egy ciklus segítségével.

- double distance(double x1, double y1, double x2, double y2)

Ez a függvény kiszámolja két pont között a távolságot.

- bool snake_eats(Snake *snake, Apple *apple)

Ez a logikai típusú függvény visszatéríti, hogy a kígyó megette-e az almát, azaz ütközik-e vele a képernyőn. Ha a kígyó megette az almát, true értékkel tér vissza, ellenben false-al.

- bool death1(Snake *snake)

Ez a logikai típusú függvény igazat térít vissza, ha a kígyó a saját farkának ütközik neki, másképp hamisat.

- bool death2(Snake *snake1, Snake *snake2)

Ez a függvény igazat térít vissza, ha a kétjátékosmódban a két kígyó egymásnak ütközik, ellenben hamissal tér vissza a hívóhoz.