Tetris

A tetris játékban a játékos az általa megadott méretű pályán játszik.
A játékban a játékos egy véletlenszérű alakzatot kell, hogy irányítson, és az a célja, hogy a sorokat feltöltse ezekkel az alakzatokkal azért, hogy azok eltűnjenek, és cserébe pontokat kapjon. Ezek az alakzatok a pálya tetejéről esnek lefele, addig amíg képes a pálya tetejére generálni egy alakzatot anélkül, hogy bele ütközne egy másik alakzatba.
A játék részén kívül tartalmaz, még egy menüt, egy beállításokat, és egy dicsőséglistát.

A tetrishez használt adatszerkezetek

A menü és a különböző nézetek közötti navigációhoz a „Nezet” adatszerkezetet lehet használni. Ennek 5 egyéni értéke van ezek pedig a: „Menu”, „Jateknezet”, „Dicsoseglista”, „Beallitasok”, „Jatek_utani_nezet”.
Ezeken a nézeteken különböző feliratokkal/gombokkal is találkozik a felhasználó, ezeknek a megjelenítéséhez, pedig „Gombok” adatszerkezet szükséges. Ezeknek mindegyike egy-egy szöveget tartalmaz, amelyet meg kell, hogy jelenítsenek.
A játékban előforduló alakzatok eltárolásához, pedig az „Alak” adatszerkezet lett létrehozva. Egy-egy alak a hozzá tartozó 4 kockának a pozícióját, és az alakhoz tartozó szín számát tárolja el.
A játékosok eredményeinek a tárolásához „Eredmenyek” adatszerkezet lett létrehozva. Egy-egy eredmenyek típusú változó egy játékosnak a nevét tartalmazza, és az általa elért pontszámot.

A program működését vezérlő függvények

void menu_megjelenites(SDL_Renderer *renderer,SDL_Event ev,int *kijelolt_gomb,Nezet *jelenlegi_nezet, int kepernyomagassaga, int kepernyoszelesseg)

Ez a függvény a menü megjelenítéséért, és a rajta elhelyezkedő gombok közötti navigálásért felelős.

void jatek(SDL_Renderer *renderer, SDL_Event ev,Nezet *Jelenlegi_nezet, int kepernyomagassaga, int kepernyoszelesseg,int *pontozas, int szelesseg, int magassag)

A jatek függvény a programnak a játékkal foglalkozó függvényeit gyűjti össze, és azoknak a segítségével gondoskodik a Tetris megjelenítéséről, és az alakzatok irányításáról.

void jatek_utani_nezet_megjelenitese(SDL_Renderer *renderer,SDL_Event ev,Nezet *Jelenlegi_nezet, int *Jelenlegi_gomb,char **nev, int pontozas, int kepernyomagassaga, int kepernyoszelesseg)

Ez a függvény „Jatek_utani_nezet” nézet. Itt felajánlja a játékosnak, hogy mentse el az eredményét.

void Dicsoseglista_megjelenitese(SDL_Renderer *renderer,SDL_Event ev,Nezet *jelenlegi_nezet, int kepernyomagassaga, int kepernyoszelesseg)

Ez a függvény dicsőséglista megjelenítéséért felel. Itt írja ki az 5 legjobb elmentett eredményt.

void Beallitasok_megjelenites(SDL_Renderer *renderer,SDL_Event ev,int *kijelolt_gomb,Nezet *jelenlegi_nezet,char **nev,int kepernyomagassaga, int kepernyoszelesseg, int *szelesseg, int *magassag)

Ez a függvény a „Beallitasok” nézet. Ez gondoskodik a beállítások megjelenítéséről és a gombok közötti navigációról. Ennek segítségével lehet beállítani a játékon belüli pálya méretét.

A hatter_betoltese típus

void Hatter_kezelese(SDL_Renderer *renderer)

Ez a függvény felelős a függvényben meghatározott kép betöltéséért, és annak megjelenítéséért.

A gomb_nezet_strukturai típus

void Gombok_kiirasa(SDL_Renderer *renderer,char *tombelem,bool kivalasztott,int kezdo_magassag, int kezdo_szelesseg, int szelesseg_offsett, int magassag_offsett, int listas, int hanyadik_elem)

Ez a függvény felelős a gombok szövegeinek a kiírásáért, megjelenítéséért. Megkapja, hogy a szöveget honnan kezdje el kirajzolni, hogy azt el kell-e csúsztatni vízszintesen vagy függőlegesen, és hogy a kiírás az egymás alá fog-e következni.

Gombok Gombba_alakitas(char *szoveg)

A paraméterként kapott szövegnek dinamikusan lefoglal egy memória területet, és egy gombbá alakítja, majd visszaadja azt.

void Gombok_felszabaditasa(Gombok *tomb, int meret)

A gombbá alakított szövegeknek lefoglalt memória területet felszabadítja.

A menu típus függvényei

Ez a típus felel a menüvel kapcsolatos műveletekért.

void menu_iranyitas(SDL_Renderer *renderer,SDL_Event ev,int *jelenlegi_gomb,Nezet *jelenlegi_nezet)

Ez a függvény felel, a gombokkal való interakciókért. Ezzel tud a felhasználó közlekedni köztük, és az „Enter” gomb lenyomásával pedig át tud lépni a megfelelő nézetre.

void menu_megjelenites(SDL_Renderer *renderer,SDL_Event ev,int *kijelolt_gomb,Nezet *jelenlegi_nezet, int kepernyomagassaga, int kepernyoszelesseg)

A menu_megjelenites gyűjti össze a menu_iranyitas, Hatter_kezelese, és a Gomb_kiirasa függvényeket, és ezek segítségével hozza létre a menüt.

A szoveg_beolvasasa típus

char *input_text(size_t hossz, SDL_Rect teglalap, SDL_Color hatter, SDL_Color szoveg, TTF_Font *font, SDL_Renderer *renderer,int *kijelolt_gomb, int kepernyomagassaga, int kepernyoszelesseg, int szelesseg, int magassag)

Ez a függvény a beállítások nézethez szükséges, hiszen ott lehet ezzel beállítani a felhasználó nevét. A függvény beolvas bármely karaktert, a megadott mennyiségig, azonban vannak különleges karakterek, amelyek más reakciót váltanak ki. Ez az „enter” „space” és a „backspace” gombok. A „backspace” gombbal ki tudja törölni az utolsó karaktert, az „enter”-rel és a „space”-cel pedig be tudja fejezni a szövegbeolvasást. A végén a beírt szövegnek lefoglal dinamikusan egy memóriaterületet, és azt visszaadja a függvény végén.

A beallitasok tipus

void beallitasok_iranyitas(SDL_Renderer *renderer,SDL_Event ev,int *jelenlegi_gomb,Nezet *jelenlegi_nezet, char **nev, int kepernyomagassaga, int kepernyoszelesseg, int *szelesseg, int *magassag)

Ez a függvény szinte megegyezik a menu_iranyitas-sal, azonban az első két gombon lehet használni a jobbra és ballra nyilakat annak érdekében, hogy csökkenjen, növekedjen a sorok/oszlopok száma a játékon belüli pályán. Azonban ezeket az értékeket, csak 10 és 40 között lehet meghatározni kettesével. A harmadik gombon, pedig a „enter” gomb megnyomásával képes a felhasználó megváltoztatni a nevét az „input_text” függvény segítségével, a felhasználónév csak akkor változik meg, ha nem egy üres szöveget akarna lementeni a felhasználó.

void Beallitasok_megjelenites(SDL_Renderer *renderer,SDL_Event ev,int *kijelolt_gomb,Nezet *jelenlegi_nezet, char **nev, int kepernyomagassaga, int kepernyoszelesseg, int *szelesseg, int *magassag)

Ez a függvény gyűjti össze a Gombba_alakitas, Hatter_kezelese, és a Gomb_kiirasa, beallitasok_iranyitas, és a Gombok_felszabaditasa függvényeket, ezeknek a segítségével hozza létre a beállításokat.

void Beallitasok_megjelenites_beolvasashoz(SDL_Renderer *renderer,int *kijelolt_gomb, int kepernyomagassaga, int kepernyoszelesseg, int szelesseg, int magassag)

Ez a függvény megegyezik a Beallitasok_megjelenites függvénnyel, azonban itt nem írja ki a felhasználó nevét, hiszen annak helyén történik a szövegbeolvasás.

A mozgas típus

A mozgas típus felel a játékos által irányított alakzattal kapcsolatos feladatokért.

void mozgas(int *jelenlegi,Alak *alakzatra_mutato,int magassag, int szelesseg,int *sor,int *oszlop, int **tomb,SDL_Event ev,bool *dobas,int delay)

Ez a függvény gondoskodik a játékos alakzatjának irányításáért. Ha a felhasználó nem csinál semmit, akkor 0.3 másodperc elteltével tovább lép a függvény. A nyilak segítségével tud jobbra/ballra lépni, ha azzal nem lép ki a pályáról, vagy nem ütközik bele egy másik kockába. A space gombbal pedig le tudja ejteni azonnal a játékos az alakzatot. A lefele nyíllal pedig lejebb lép egyet.

void utkozes_kerulese(int *jelenlegi,Alak *alakzatra_mutato,int magassag,int szelesseg, int sor,int *oszlop,bool *kerulte, SDL_Event ev, int **tomb)

Ez megegyezik a mozgas függvénnyel azzal a kivétellel, hogy megkapja mutatóként a kerulte változót is. Ezt az igaz-hamis értéket aszerint változtatja, hogy sikerült-e kikerülnie az alatta lévő kockát, vagy sikerült-e arrébb mozdulnia a pálya alján.

void kotelezo_lefele(Alak *alakzatra_mutato,int jelenlegi,int szelesseg,int magassag,int *sor,int oszlop,int **tomb)

Az a feladata, hogy megnézze, képes-e még eggyel lejjebb menni. Ha képes rá, akkor meg is teszi azt.

void eses_gyorsitasa(int jelenlegi_ido,int *szint,int kezdodes,int *delay)

Megnézi, hogy eltelt-e már 30 másodperc a legutóbbi frissítés óta, ha letelelt, akkor csökkenti a delay értéket, ami meghatározza, hogy hány másodpercig tud gondolkozni a felhasználó mielőtt újra lejjebb esne, és növeli a játékos szintjét eggyel.

Az Elorejelzes típus

static void (alakzatneve)_megrajzolasa(SDL_Renderer *renderer)

Ezek a függvények megrajzolják a képernyő bal oldalára a következő alakzatot.

static void Kovetkezo_Alakzat_Megrajzolasa(SDL_Renderer *renderer,int kovetkezo)

Eldönti, hogy melyik alakzatot kell megrajzolnia.

static void Felirat_Kiirasa(SDL_Renderer *renderer,char *tombelem,int szelesseg)

A képernyő bal oldalára kiír egy szöveget.

void Elorejelzes_megjelenitese(SDL_Renderer *renderer,int kovetkezo,char *tombelem,int szelesseg)

Megrajzolja a következő elemet és kiír egy feliratot a Felirat_Kiirasa és a Kovetkezo_Alakzat_Megrajzolasa függvények segítségével.

A megjelenitesek típus

A megjelenitesek típus felel a játék nézettel kapcsolatos megjelenítésekkel.

void sdl_init(char const *felirat, int szeles, int magas, SDL_Window **pwindow, SDL_Renderer **prenderer)

Ez felel a program által használt ablak létrehozásáért.

static void tablazat_megrajzolasa(SDL_Renderer *renderer,int **tomb, int kepernyomagassaga, int kepernyoszelesseg,int szelesseg, int magassag)

Ez a függvény felel a játék pályájának a megrajzolásáért. A tömbnek minden eleme egyenlő egy 0 és 7 közötti értékkel, amely egy-egy színt reprezentál.

static void jatekos_megrajzolasa(SDL_Renderer *renderer,Alak *alakzatra_mutato,int sor,int oszlop,int jelenlegi, int kepernyomagassaga, int kepernyoszelesseg, int szelesseg, int magassag)

A játékos által irányított alakzatot jeleníti meg a pálya fölé.

static void ghost_jatekos(SDL_Renderer *renderer,int jelenlegi,Alak *alakzatra_mutato,int magassag, int szelesseg, int sor, int oszlop, int **tomb, int kepernyomagassaga, int kepernyoszelesseg)

Megnézi a also_utkozes függvény segítségével, hogy a játékos alakzata hova fog érkezni, és oda megjeleníti az alakzat halványabb verzióját.

static void grid_megrajzolasa(SDL_Renderer *renderer, int kepernyomagassaga, int kepernyoszelesseg, int szelesseg, int magassag)

Megrajzolja a pályára a kockákat elválasztó vonalakat.

void jatek_megrajzolasa(SDL_Renderer *renderer,int **tomb,Alak *alakzatra_mutato,int sor,int oszlop,int jelenlegi,int magassag,int szelesseg, int szint,int pontozas, int kovetkezo, int kepernyomagassaga, int kepernyoszelesseg)

Ez a függvény gyűjti össze a Gombba_alakitas, Hatter_kezelese, tablazat_megrajzolasa, jatekos_megrajzolasa, ghost_jatekos, grid_megrajzolasa, Gombok_kiirasa, Elorejelzes_megjelenitese és a Gombok_felszabaditasa függvényeket, ezek segítségével megrajzolja a játék jelenlegi állását.

Az Alakzatok_inicializalasa típus

void (alakzat neve)_ alakzat(Alak *kimenetel,int szelesseg)

Ezek a függvények lementik a kimenetel elemeibe az alakzat kockáinak pozícióit.

A tetris_palya típus

Alak *alak_sorsolas(int *melyik_lesz,Alak *kimenetel, int szelesseg)

Megkapja a melyik értékét, és az alapján a kimenetelbe betáplálja a megfelelő alakzat pozícióit, majd a végén kisorsol egy számot, amelyet maradékosan eloszt 7-tel, és lementi a melyikbe.

int **tablazat_inicializalasa(int magassag,int szelesseg)

Dinamikusan lefoglalja a táblázat sorainak és oszlopainak a memóriaterületet, majd pedig azokat feltölti 0-kal.

void tablazat_felszabaditasa(int **tomb, int magassag)

Felszabadítja a táblázat sorainak, oszlopainak lefoglalt memóriaterületet.

bool also_utkozes(int jelenlegi,Alak *alakzatra_mutato,int magassag, int szelesseg,int sor,int oszlop,int **tomb)

Megnézi, hogy a következő esés után eléri-e a pálya alját, vagy beleütközik-e egy másik kockába.

void elem_lementese(int **tomb,int szelesseg,int sor,int oszlop,int jelenlegi, Alak *alakzatra_mutato)

Lementi a tömb megfelelő elemébe a játékos által irányított alakzatot.

static void lejebb_mozgatas(int **tomb,int kezdoertek, int szelesseg)

A felette lévő sorokat lejjebb mozgatja, és a legfelső sort telerakja 0-kal.

void telesorok(int **tomb, int magassag, int szelesseg, int *pontozas, int szint)

Végigmegy a tömbön és ha azt észleli, hogy egy sor minden elemében van valamilyen nem nullával egyenlő érték, meghívja a lejebb_mozgatas függvényt. Azt is megfigyeli, hogy ez egy meghívás alatt hányszor történik meg, mert annak, a pálya szélességének és a szint szorzatának a 10 szeresét hozzáadja a pontozáshoz.

A collision_detection típus

void collision_eszleles(int most, int *ezelotti,bool *game,bool dobas,Alak *alakzatra_mutato,int **tablazat,SDL_Renderer *renderer,SDL_Event ev,Alak *kimenetel, int szint,int *pontozas, int *kovetkezo,int *jelenlegi,int *delay, int *sor, int *oszlop ,int kepernyomagassaga, int kepernyoszelesseg, int szelesseg, int magassag)

Ellenőrzi, hogy letelt-e már a megfelelő idő, ha igen akkor a kotelezo_lefele függvény lejjebb löki a játékost egy kockányival. Ilyenkor a játékot újra megjeleníti.

Ezt követően az also_utkozes függvény ellenőrzi, tud-e még lejjebb menni. Ha nem, akkor van még egy esélye, hogy oldalra tudjon lépni.

Amennyiben az nem sikerült akkor az elem_lementesev függvénnyel lementi a tömbbe a játékos alakzatját, és egy új alakzatot sorsol a játékosnak.

Abban az esetben, ha már nem fér ki a pályán, akkor a játékmenetnek vége van.

A kesz_jatek típus

A kesz_jatek egy függvényből áll, amely a tetris_palya, a mozgas és a megjelenitesek típusok segítségével megalkotja a játékot.

void jatek(SDL_Renderer *renderer, SDL_Event ev,Nezet *Jelenlegi_nezet, int kepernyomagassaga, int kepernyoszelesseg,int *pontozas,int szelesseg, int magassag)

A beállításokban megadott sorok és oszlopok száma szerint lefoglalja a pálya területét, amit a tablazat_inicializalasa segítségével feltölt 0-kal. A játékosnak kisorsolja a kezdő alakzatot az alak_sorsolas függvénnyel.

A játék maga egy ciklusban játszódik le, ami addig megy, amíg a játékos nem veszít, vagyis a game hamis nem lesz.

A ciklus elején megnézi a játék elkezdése óta eltelt ms-t ami a kotelezo_lefele időzítése miatt szükséges. A játékot a jatek_megrajzolasa megjeleníti, majd pedig várakozik a játékos válaszára.
Végezetül pedig meghívja a collision_eszleles függvényt, hogy megnézze, lejebb essen eggyel a játékos alakzatja és hogy ellenőrizze, hogy mehet-e tovább a játékmenet.
A függvény végén felszabadítja a táblázatnak, és a játékos által irányított alakzatnak lefoglalt memória területet és tovább lép a „Jatek_utani_nezet”-re

Az Eredmeny_mentese típus

void mentes(char *nev, int pontozas)

Ellenőrzi, hogy létezik-e az „eredmenyek.txt”, ha nem akkor létrehoz egy fájl ezzel a névvel, és beleírja a játékos nevét, és az eredményét, ha pedig létezik, akkor ezt a fájl végéhez írja hozzá.

A Jatek_utani_nezet típus

void jatek_utani_nezet_iranyitas(SDL_Renderer *renderer,SDL_Event ev,int *jelenlegi_gomb,Nezet *jelenlegi_nezet,char **nev,int pontozas)

Ennek a függvénynek a segítségével lehet navigálni két gomb között, a megfelelő gombon az „enter” gombot lenyomva lementi a játékos nevét és pontszámát. Bármely gomb lenyomásával tovább fog lépni a menübe.

void Nem_Igen_kiirasa(SDL_Renderer *renderer,char *tombelem,bool kivalasztott,int kepernyomagassaga, int kepernyoszelesseg, int hanyadik_elem)

A képernyő alsó felére kiírja a számára megadott szöveget.

void jatek_utani_nezet_megjelenitese(SDL_Renderer *renderer,SDL_Event ev,Nezet *Jelenlegi_nezet, int *Jelenlegi_gomb,char **nev,int pontozas, int kepernyomagassaga, int kepernyoszelesseg)

A Gombba_alakitas, Hatter_kezelese, Gombok_kiirasa, Nem_Igen_kiirasa és a jatek_utani_nezet_iranyitas, Gombok_felszabaditasa függvények segítségével létrehozza a Játék utáni nézetet.

A dicsoseglista típus

bool letezik_e_a_fajl()

Ellenőrzi, hogy létezik-e az „eredmenyek.txt” fájl, ha létezik, akkor igaz értéket ad vissza, ha pedig nem akkor hamisat.

int hany_sora_van()

Megszámolja, hogy hány sor van benne az „eredmenyek.txt” fájlban.

void eredmenyek_beolvasasa(Eredmenyek *tomb)

Beolvassa a fájlból a játékosok neveit, és az általuk elért pontszámokat egy Eredmenyek típusú tömbbe.

void eredmenyek_csereje(int *elso, int *masodik)

Megcseréli két játékosnak az eredményét.

void felhasznalonevek_csereje(char **elso, char **masodik)

Megcseréli két játékos nevét.

void eredmenyek_rendezese(Eredmenyek *tomb,int mennyi)

A játékosok által elért pontszámok szerint csökkenő sorrendbe rendezi a paraméterként kapott tömböt a függvény.

void eredmenyek_felszabaditasa(Eredmenyek *tomb,int mennyi)

Felszabadítja a játékosok nevének dinamikusan lefoglalt memória területét.

void dicsosegek_kiirasa(SDL_Renderer *renderer,char *tombelem,bool eredmeny_e,int kepernyomagassaga, int kepernyoszelesseg, int hanyadik_elem)

Kiírja a paraméterként megkapott szöveget a bal vagy jobb oldalra az eredmeny_e értékétől függően.

void dicsosegek_megjelenitese(SDL_Renderer *renderer, int kepernyomagassaga, int kepernyoszelesseg)

A hany_sora_van, eredmenyek_beolvasasa, eredmenyek_rendezese, dicsosegek_kiirasa függvények segítségével beolvassa, eltárolja a lementett eredményeket, azokat csökkenő sorrendbe rendezi, kiírja őket. Ha kevesebb eredmény van lementve, mint 5 akkor azt mind kiírja, ha pedig legalább 5 eredmény van akkor az 5 lejobbat írja ki. A függvény végén pedig az eredmenyek_felszabaditasa segítségével felszabadítja a neveknek lefoglalt memória területet.

void Dicsoseglista_megjelenitese(SDL_Renderer *renderer,SDL_Event ev,Nezet *jelenlegi_nezet, int kepernyomagassaga, int kepernyoszelesseg)

A Hatter_kezelese, Gombok_kiirasa, dicsosegek_megjelenitese függvények segítségével megjeleníti a dicsőséglistát. Bármely gomb lenyomásával visszatér a menübe.