Házi feladat – Specifikáció 2024

Programozás alapjai 2

Készítette – Szuhi Levente (YHXVGZ)

1. Feladat

A feladat egy objektum létrehozása a Reversi (Othello) nevű játék megvalósítására. A játék grafikus felhasználói felület használata nélkül, konzolosan lesz használható. Az objektum „ismerje” a szabályokat, azaz automatikusan „forgassa” át a megfelelő korongokat, melyek szintén objektumok. A működést demonstrálására külön modulként fordított tesztprogram felel. A feladat megoldásához STL tároló nem lesz felhasználva.

2. Feladatspecifikáció

1.1 Játékszabály

A reversi játékot egy 8x8 táblán szokás játszani, két különböző színű koronggal. A játék kezdetén a tábla középső 2x2 négyzetébe kerül játékosonként 2 korong úgy, hogy az egyszínűek átlósan helyezkedjenek el. A következő játékos akkor helyezhet el korongot, ha az új korong és egy már ott lévő (saját színű) korong közrefogja az ellenfélnek legalább egy vonalát, vagyis sorát, vagy oszlopát. Ha letette a játékos a korongot akkor az összes ilyen vonalat át kell forgatni. Ha van szabályos lépés, akkor kötelező tenni, passzolni nem szabad. Ha azonban nincs lehetőség szabályosan lépni, akkor passzolni kell. A játéknak akkor van vége, ha egyik félnek sincs szabályos lépése, ilyenkor az győz, akinek több korongja van; döntetlen lehetséges.

2.2 A program használata

A program menüvezérelt, ahol a megadott lehetőségek közül választva lehet továbblépni a megfelelő helyre. A vezérlési opciók minden programpontnál „[VEZÉRLÉSI KÓD] FUNKCIÓ” formátumban vannak megjelenítve. A megfelelő vezérlési kód megadása után a program a választott opciónak megfelelően lép tovább. *Pl.: [1] Új játék választásához a bemenetnek „1”-nek kell lennie*. Hibás választásról a program értesítést küld, és a program csak akkor lép tovább, ha létező opciót kap a bemenetre. Játék közben a következő korong elhelyezéséhez meg kell adni a pálya megfelelő koordinátáját. Hibás koordináta megadásáról értesítést kap a felhasználó.

2.3 A program képességei

A játékban választani lehet, hogy gép ellen, vagy másik személy ellen szeretnénk-e játszani. A játék során a táblán levő korongok számát játékosonként jelzi a kijelzőn. A program segíti a következő játékost azzal, hogy a lehetséges szabályos lépéseket jelzi a pályán egy „X” jellel. Hibás lépésre/inputra felhívja a felhasználó figyelmét. Játék közben lehetőség van kilépni és elmenteni a játékot. Az elmentett játék egy szövegfájlba kerül mentésre. Új játék kezdésekor az előzőleg mentett játék törlődik.

2.4 Főmenü megjelenése

[1] Új játék
[2] Szabályok
[3] Teszt
[4] Kilépés

2.5 Játék mentése

Az elmentett játék egy szövegfájlba kerül. A két játékos korongjainak színét „W” és „B” karakterek jelzik, amiket vesszők választanak el egymástól. Az üres mezőket „#” jelzi. Például egy sor:

#,W,W,W,B,#,#,#

Beolvasáskor ha ezektől eltérő karaktert, vagy eltérő méretű pályát talál a program, akkor hibaüzenetet küld és törli a mentett játékot.
Egy mentés tartalma:

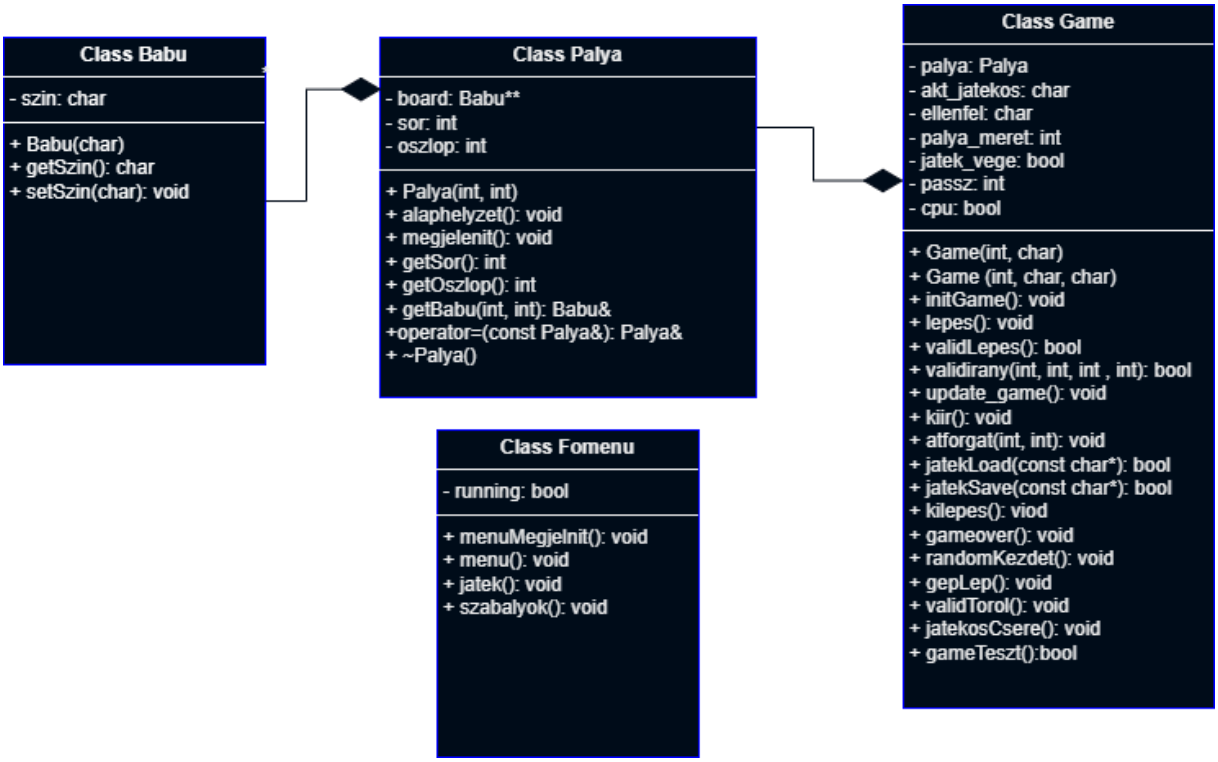
1. Aktuális játékos
2. A pálya mérete
3. A cpu ellenfél színe
4. A pálya mezői

2.6 Játék vége

A játék végén a program győztest hirdet és átirányít a főmenübe.

3. Terv

3.1. UML diagram és objektumok



A játékot 4 osztály segítségével tervezem megvalósítani.

A Fomenu osztály feladata a főmenü megjelenítése, és a menü továbbléptetéshez szükséges input bekérése. Továbbá megjeleníti a szabályokat és a teszt eseteket is.

A Babu osztály tartalmazza egy bábu színét és az beállításához/lekérdezéséhez szükséges függvényeket.

A Palya osztály feladata a pálya felépítése a bábukból. Egy két dimenziós tömbben vannak tárolva a bábuk, amik a megfelelő koordinátán levő bábu információját tárolja. Ez az osztály felel a pálya megjelenítéséért is.

A Game osztály felel a játék vezérléséért. A következő lépés előtt ellenőrzi a szabályos lépéseket, amit a táblán egy X-el jelöl majd. Ellenőrzi az következő lépés inputjának helyességét is. Szabályos lépést követően átforgatja a megfelelő bábukat. Ha nem lehetséges szabályos lépést végrehajtani, akkor a szabályoknak megfelelően passzolásra „kényszeríti" az aktuális játékost. Amennyiben egyik játékos sem tud már szabályos lépést végrehajtani, úgy összegzi a bábuk számát és kijelzi az eredményt. A játék elején, amennyiben mentett játék folytatása lett választva, úgy megpróbálja betölteni a mentett játékot. Ha sikertelen a betöltés, úgy értesíti erről a felhasználót. Szintén a játék elején lehet kiválasztani, hogy kétszemélyes játékot vagy gép ellenit szeretne-e a felhasználó játszani.

3.2 Algoritmusok

a) Szabályos lépés keresése

1. A pálya minden celláját bejárja a program. Ha olyan cellát talál ami üres, akkor az őt körbevevő 8 irányt egyesével vizsgálja meg. Megnézi hogy az adott irányba indulva:

- nem lép-e ki a pályáról
- ellentétes színű bábut talál-e

2. Ha a fenti feltételek teljesülnek akkor ebbe az irányba tovább kell vizsgálni a cellákat. A következő lehetőségek fordulhatnak elő:

- A következő cella ellentétes színű:
  - Ilyenkor az adott irányba továbblépve ellenőrizzük a következő cellát.
- Kiléptünk a pályáról:
  - Visszalépünk az 1. ponthoz és egy olyan irányt vizsgálunk, ami még nem volt.
- A következő cella azonos színű:
  - A kiindulási cella egy szabályos lépés pontja. Ilyenkor a bábu színét 'X'-re állítja a program és nem vizsgálja a további irányokba. A következő cella vizsgálata következik.

Amennyiben mind a 8 irány vizsgálata megtörtént és nem talált szabályos lépést a program, úgy továbblép a következő cellára.

b) CPU lépés

A CPU játékos lépésekor megkeresi a szabályos lépéseket és azokat egy „X"-vel jelöli. A lépés algoritmus megszámolja a valid lépéseket majd egy 1 – max. lépésszám intervallumon belül generál egy véletlen számot. Ennek a számnak megfelelő szabályos lépést választja a gép.

c) Játék vége

A validlepes függvény megkeresi az összes létező lépést az aktuális játékosnak megfelelően. Ha nem maradt a táblán szabad (azaz „.”-val jelzett) mező, akkor az a játék végét jelenti. Ilyenkor a passz értéket 2-re állítja a függvény (ami önmagában a játék végét jelenti), továbbá a jatek_vege logikai értéket is TRUE-ra állítja. Így ennek a két értéknek az együttes jelzése jelenti, hogy a játék azért ért véget, mert nincs több lépés. A két érték együttes használatára azért van szükség, mert a játékból ki lehet lépni játék közben. Ilyenkor a jatek_vege TRUE értéket kap, de a passz számláló nem lesz legalább kettő.

4. Tesztelés

A program tesztelése a fomenu.cpp-ben van megvalósítva. Egy „TESZT” nevű makró értékét 0-ról 1-re állítva és a programot futtatva a teszt rész fog lefutni. 3 teszt kategória van létrehozva a programban a 3 játékkal kapcsolatos osztályhoz.

1. Teszt1 - Babu osztály tesztjei
2. Teszt2 - Palya osztály tesztjei
3. Teszt3 - Game osztály tesztjei

A Game osztályra viszonylag nehezen sikerült külön teszteseteket generálni, így ott egy olyan teszt is lefut a végén, ami két cpu játékos segítségével lejátszik egy játékot. A játék végén kijelzi az eredményt és a pálya utolsó állását is.