**Programozás 3. beadandó**

Készítette: Fodor Dániel Gábor  
NK: A5KJJ5

**Feladat**

3. Készítsünk C++ programot a következő feladat megoldására!

Szimuláljuk az alábbi egyszerűsített Monopoly társasjátékot! Adott néhány eltérő stratégiájú játékos és egy körpálya, amelyen különféle mezők sorakoznak egymás után. A pályát körbe-körba újra és újra bejárják a játékosok úgy, hogy egy kockával dobva mindig annyit lépnek, amennyit a kocka mutat. A mezők három félék lehetnek: ingatlanok, szolgáltatások és szerencse mezők. Az ingatlant meg lehet vásárolni 1000 Petákért, majd újra rálépve házat is lehet rá építeni 4000 Petákért. Ha ezután más játékos erre a mezőre lép, akkor a mező tulajdonosának fizet: ha még nincs rajta ház, akkor 500 Petákot, ha van rajta ház, akkor 2000 Petákot. A szolgáltatás mezőre lépve a banknak kell befizetni a mező paramétereként megadott összeget. A szerencse mezőre lépve a mező paramétereként megadott összegű pénzt kap a játékos. Háromféle stratégiájú játékos vesz részt a játékban. Kezdetben mindenki kap egy induló tőkét (10000 Peták), majd A „mohó” játékos ha egy még gazdátlan ingatlan mezőjére lépett, vagy övé az ingatlan, de még nincs rajta ház, továbbá van elég tőkéje, akkor vásárol. Az „óvatos” játékos egy körben csak a tőkéjének a felét vásárolja el, a „taktikus” játékos minden második vásárlási lehetőséget kihagyja. Ha egy játékosnak fizetnie kell, de nincs elégendő pénze, akkor kiesik a játékból, házai elvesznek, ingatlanjai megvásárolhatókká válnak.

A játék paramétereit egy szövegfájlból olvassuk be. Ez megadja a pálya hosszát, majd a pálya egyes mezőit. Minden mezőről megadjuk annak típusát, illetve ha szolgáltatás vagy szerencse mező, akkor annak pénzdíját. Ezt követően a fájl megmutatja a játékosok számát, majd sorban minden játékos nevét és stratégiáját. A tesztelhetőséghez fel kell készíteni a megoldó programot olyan szövegfájl feldolgozására is, amely előre rögzített módon tartalmazza a kockadobások eredményét.

**Írjuk ki, hogy adott számú kör után hogyan állnak (mennyi a tőkéjük, milyen ingatlanokat birtokolnak**

**Specifikáció**

A különböző játékosok és mezők megadásához 4-4 osztályt hozunk létre. 1-1 ősosztályt(Player,field) és 3-3 származtatott osztályt. Ezeket a „Track” nevű osztály kapcsolja össze, tárolja és ebben találhatóak a játékot működtető függvények is. Minden játékosnak van neve (string name), pénze (int money), pozíciója (int position) és egy körszámlálója (int roundcnt). A különbségek a függvényekben jelentkeznek majd. Egy játékosnak még vannak függvényei is, movetoposition(), pay(), illetve az adatai lekérdező függvények (isalive(), getposition(), getroundcnt()).

A mezőknek 2 virtuális függvényük van, a price() (mennyit kell fizetni az adott mezőre lépve) illetve a buy() (megvásárolják a mezőt). Ezek minden mezőnél másképp működnek.

A Track osztályban 1-1 vectorban tároljuk a játékosokat (players) és a mezőket (fields). Ebben az osztályban található a futtató függvény (play()), játékosok és mezők beolvasása (addpalyer(), addfield()) és az eredmények kiíratása is (results()).

**Terv**

A játékosok fajtái és metódusai:

|  |
| --- |
| **Track** |
| #fields : vector<fields\*>  #players : vector<players\*> |
| +results() : void  +addfield(const string&, int) : void  +adddplayer(const string&, const string&)  +play(int) : void  +newposition(player\*, int) : int |

|  |
| --- |
| **Field** |
|  |
| +buy(player\*) : virtual void  +price(player\*) : virtual int |

|  |
| --- |
| **Lucky** |
| #paramprice : int |
| +lucky (int)  +price (player\*) : int |

|  |
| --- |
| **Player** |
| #name : string  #money: int  #position : int  #roundcnt : int |
| +player (const string&)  + getname : const std::string&  +getmoney : int  +getposition() : int  +getroundcnt() : int  +movetoposotion (int) : void  +pay (int) : void  +shouldpay(int) : virtual bool |

|  |
| --- |
| **Service** |
| #paramprice : int |
| +service(int)  +buy(player\*) : void  +price(player\*) : int |

|  |
| --- |
| **Greedy** |
|  |
| +greedy(const string&)  +shouldpay(int) : bool |

|  |
| --- |
| **Tactical** |
|  |
| +tactical(const string&)  +shouldpay(int) : bool |

|  |
| --- |
| **Ingatlan** |
| #owner : player\*  #istherehouse : bool  #hasOwner : bool |
| +ingatlan()  +clear() : void  +buy(player\*) : void  +price(player\*) : int |

|  |
| --- |
| **Careful** |
|  |
| +careful(const string&)  +souldpay(int) : bool |

A beolvasás során adjuk meg a játszandó körök számát, játékosok számát, játékosok neveit és tulajdonságait, mezők számát, fajtáját és a kezdeti árukat (ha van).

Ezután a play függvény meghívásával kezdődhet is a játék, a results() függvénnyel kiírhatjuk a játékosok adatai (neve,fajtája,pénze, ingatlanjai).

Állapottér: palya : track, fields : field, players : player, results : string

Előfeltétel: palya = palya0 ∧ fields = fields0

Utófeltétel: palya = palya0 ∧ ∀i∈[1..n]: roundc, play(roundc) ∧ results = results()

A play() függvény:

|  |  |  |  |
| --- | --- | --- | --- |
| Int roundc | | | |
| i = 1 … roundc | | | |
|  | int dice | | |
| Players[i].isalive() | | |
| continue | newposition(players[i],dice)  movetoposition(newposition)  int price(players[i]) | |
| players[i].shouldpay(price) | |
|  | field.buy(players[i])  players[i].isalive() |

Az algoritmus alapján minden körben minden játékos 1-szer dob (random szám 1-6 között), majd megnézzük, játékban van e egyáltalán. Ha igen, elléptetjük a megfelelő helyre, megvizsgáljuk az adott telek tulajdonságait és eldöntjük lesz a vásárlás, ha igen, megtörténik. Majd végezetül megnézzük újból játékban marad e a játékos.

Elérhetőség: wignergpulab@gmail.com