Csák Attila Gergő

s7b01e

Szimuláljuk az alábbi egyszerűsített Capitaly társasjátékot! Adott néhány eltérő stratégiájú játékos és egy körpálya, amelyen különféle mezők sorakoznak egymás után. A pályát körbe-körbe 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.
Adjuk meg, melyik játékos nyeri meg a játékot és mekkora vagyona (mennyi a tőkéje, milyen ingatlanokat birtokol) van ekkor!

Megoldási Terv

A feladatban rögtön kirajzolódik az, hogy szükségünk lesz absztrakt mező és játékos osztályokra, amikből származnak majd a különböző specifikusabb működésű

alosztályok. A cél az lesz, hogy az alosztályok közös működése ki legyen emelve a szülő osztályba.

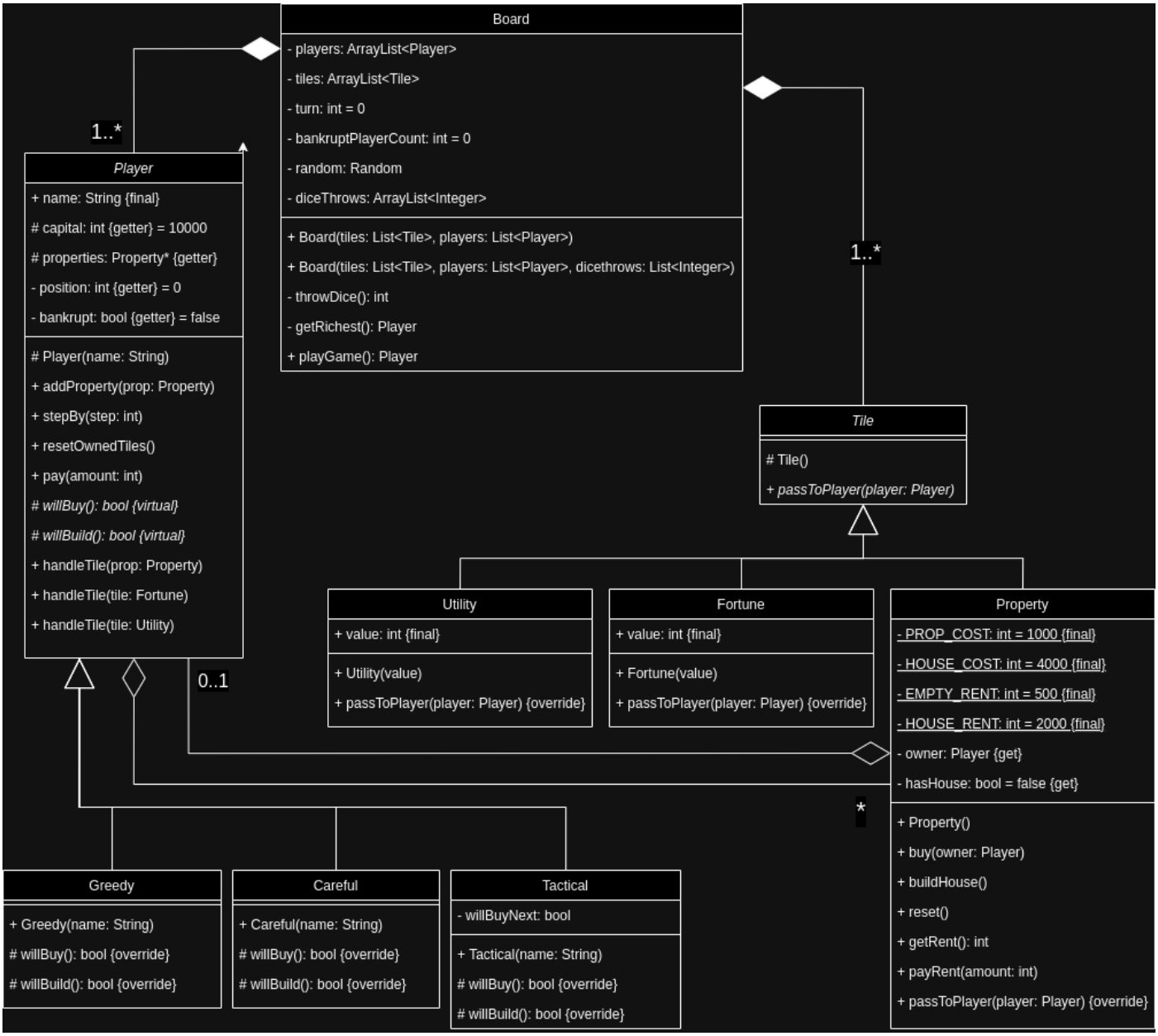
A játékos osztályba van potenciál egy sablonfüggvény mintára, mivel csak kis feltételes különbségek vannak a stratégiák működésében. A különböző mezők és a játékosok közti interakcióhoz viszont egy látogató mintára lesz szükség, hogy ne kelljen elágazásokkal ellenőrizni a mező típusát, így egyszerűbb, olvashatóbb és bővíthetőbb marad a kód.

Továbbá szükségünk lesz egy pálya osztályra, ami összefogja a mezőket és a játékosokat, és a köztük történő interakciókat kezeli.

Megvalósítás

A terv szerint megvalósítottam egy abstrakt Tile és Player osztályt, amik tartalmazzák az összes olyan adattagot és metódust, ami közös az összes alosztályban. A Board osztály kezeli maga a játék szimulációját, míg a Player és Tile alosztályok tartalmazzák a sajátos működésüket.
A játék során a Board osztály adja meg minden játékosnak, hogy mennyit lépjen, és hogy milyen mezőre landolt, majd a Board osztály beleszólása nélkül, a Player és Tile osztályok között lezajlik az interakciō (kifizeti a játékos a bérleti díjat, vagyont kap, vesz egy házat...). Minden kör végén, a Board megnézi, hogy a soron lévő játékos kiesett e van nem, és ha igen, akkor kiveszi a játékból.
Ez egészen addig megy míg csak egy játékos marad, elfogytak a megadott kockadobások vagy elértük az 1001-ik kört. Ha több mint egy játékos van játékban a szimuláció végére, a legnagyobb tőkével rendelkező játékos lesz a győztes.

UML Osztálydiagram:



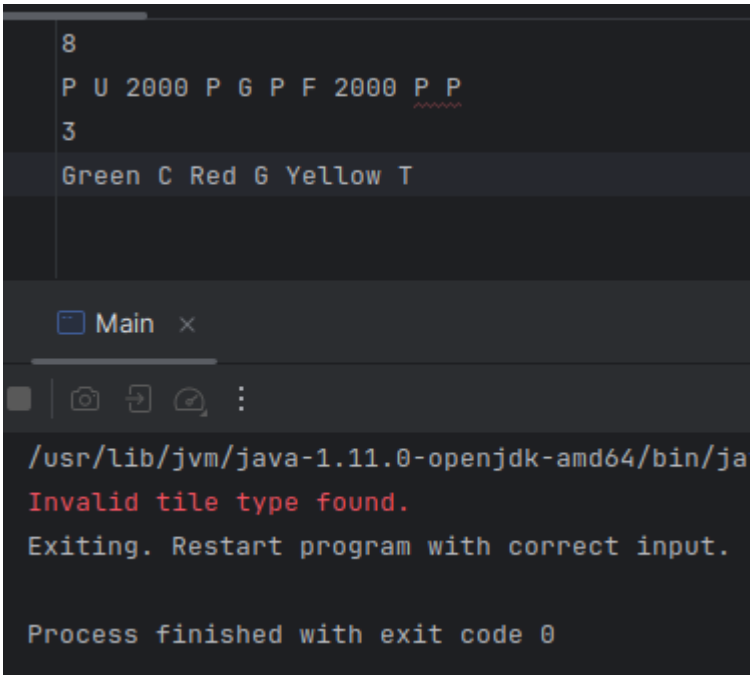
A Tile osztályból leszármazott osztályoknak a passToPlayer(player) metódusa egyszerűen meghívja az adott Player handleTile metódusát, saját magát adva paraméterként, így a Player osztály egyedi módon tudja kezelni az különböző Tile alosztályokat. Ez egy látogató mintának egy implementációja. A handleTile metódus egy túlterhelt metódus, aminek működése az átadott argumentum típusától függ.

A Player osztály handleTile(prop: Property) metódusa meghívja a willBuy és a willBuild metódusokat, amik felül vannak definiálva a Player alosztályokba. A handleTile(prop: Property) metódus egy sablonfüggvény aminek a működése a willBuy és a willBuild eltérő implementációja szab meg egy fokig. Ez egy sablonfüggvény minta.

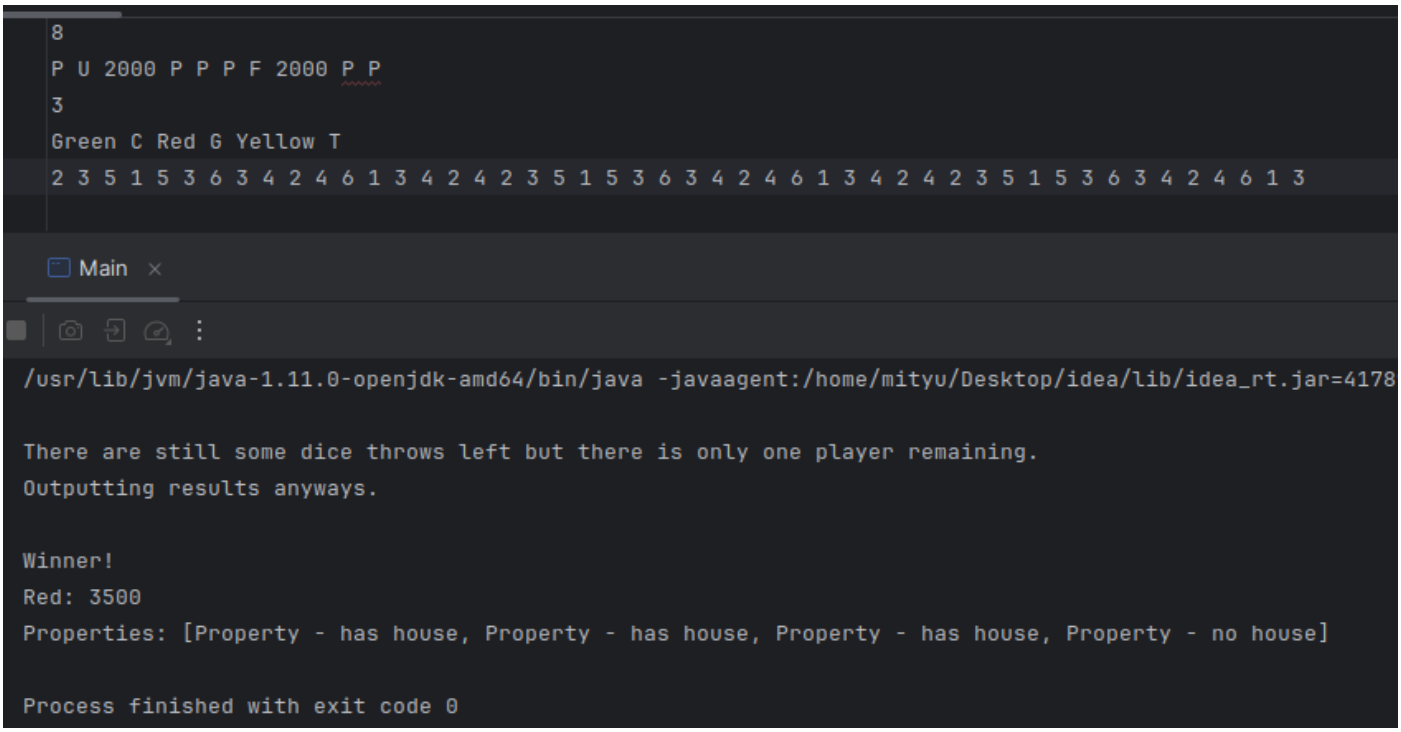
Képek a futtatásról:

Mikor nincsenek megadva kockadobás értékek, a játék addig fut míg csak egy játékos marad (vagy ha több mint 1000 kör lefutott már). A kockadobások meg véletlen vannak kiválasztva a Random osztály segítségével.

Egy példa bemeneti hiba. Az egyik mező érték rosszul lett megadva.

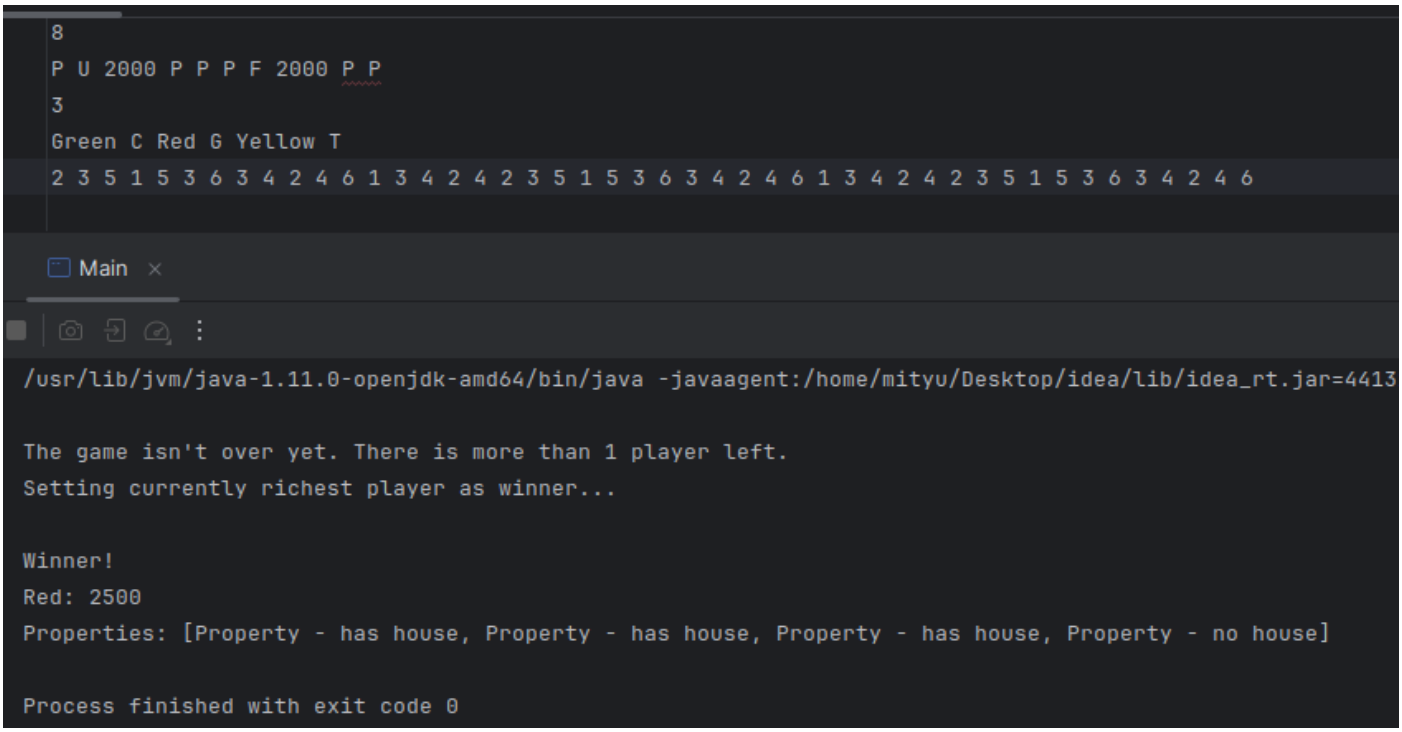


Itt túl sok kockadobás lett megadva, ezért a program hamarabb leáll, és az utolsó, nem kiesett játékos adatait választja meg győztesnek.



Itt pont vége lett a játéknak az utolsó kockadobás értéknél. A nyertes az utolsó, még játékban maradt játékos.

Itt a kockadobás értékek hamarabb elfogynak, minthogy lett volna egy nyertes. Ilyenkor a program a legtöbb tőkével rendelkező játékost választja nyertesként.



Tesztelési terv

Mivel a program működését a felhasználó csak a bemeneti fájl megadásával tudja befolyásolni, ezért néhány edge-case-t már a bemenet feldolgozása során ki tudunk szűrni (például, hogy mi van, ha nincs megadva egy játékos vagy mező sem) úgy, hogy ellenőrizzük ezeket az eseteket és elutasítjuk a kapott bemenetet.
Továbbá nem használunk egység teszteket, így a különböző osztályok metódusaiknak a tesztelése sem lehetséges.
Így majdnem minden edge-case a hibásan megadott bemenet miatt lép elő.

User Stories

AS A	Felhasználó	
I WANT TO	A nyertes játékos adatait megkapni	
1	GIVEN	Nem adok meg bemeneti fájlt
	WHEN	Elindítom a programot
	THEN	A program hibaként kiírja, hogy adjunk meg egy bemeneti fájlt.
2	GIVEN	Üres bemeneti fájlt adok meg
	WHEN	Elindítom a programot
	THEN	A program hibaként kiírja, hogy adjuk meg a pálya méretét.
3	GIVEN	Csak a pálya méretét adjuk meg
	WHEN	Elindítom a programot

	THEN	A program hibaként kiírja, hogy adjunk meg a pálya méretnek megfelelő számú mezőt.
4	GIVEN	Csak a mezőkkel kapcsolatos adatokat adom meg a bemeneti fájlba
	WHEN	Elindítom a programot
	THEN	A program hibaként kiírja, hogy adjuk meg a játékosok számát.
5	GIVEN	A mezőkkel kapcsolatos adatokat és a játékosok számát megadom, de nem a játékos adatokat, a bemeneti fájlba.
	WHEN	Elindítom a programot
	THEN	A program hibaként kiírja, hogy nem adtunk meg elég játékost.
6	GIVEN	Több mezőnek az adatait adom meg, mint amekkora a pálya.
	WHEN	Elindítom a programot
	THEN	A program hibaként kiírja, hogy helytelenül adtuk meg a játékosok számát.
7	GIVEN	Több játékosnak az adatait adom meg, mint amekkora a megadott játékosszám értéke.
	WHEN	Elindítom a programot
	THEN	A program hibaként kiírja, hogy helytelen kockadobás értéket a
8	GIVEN	Kevesebb mezőnek az adatait adom meg, mint amekkora a pályaméret értéke.
	WHEN	Elindítom a programot
	THEN	A program hibaként kiírja, hogy helytelen mezőtípus lett megadva.
9	GIVEN	Kevesebb játékosnak az adatait adom meg, mint a játékos szám, amit megadtam.
	WHEN	Elindítom a programot
	THEN	A program hibaként kiírja, hogy nincs elég játékos megadva.
10	GIVEN	Kockadobásként egy 6-nál nagyobb érték van megadva.
	WHEN	Elindítom a programot
	THEN	A program hibaként kiírja, hogy a kockadobás értékei csak 1 és 6 között lehetnek.
11	GIVEN	Kockadobásként egy 1-nél kisebb érték van megadva.

	WHEN	Elindítom a programot
	THEN	A program hibaként kiírja, hogy a kockadobás értékei csak 1 és 6 között lehetnek.
12	GIVEN	Kockadobásként egy nem egész szám érték van megadva.
	WHEN	Elindítom a programot
	THEN	A program hibaként kiírja, hogy érvénytelen kockadobás érték lett megadva.
13	GIVEN	Negatív pálya méretet adok meg.
	WHEN	Elindítom a programot
	THEN	A program hibaként kiírja, hogy legalább 1 legyen a pálya mérete.
14	GIVEN	Pályaméretként nem egy egész szám értéket adok meg.
	WHEN	Elindítom a programot
	THEN	A program hibaként kiírja, hogy pozitív egészként adjam meg a pálya méretet
15	GIVEN	Hibás értéket adok meg mező típusként.
	WHEN	Elindítom a programot
	THEN	A program hibaként kiírja, hogy nem található az adott mező típus.
16	GIVEN	Kihagyom egy szolgáltatás vagy szerencse mező értékét
	WHEN	Elindítom a programot
	THEN	A program hibaként kiírja, hogy adjunk meg egy pozitív egészet a mező értékeként.
17	GIVEN	Egy mező értékeként nem egy egész szám értéket adok meg.
	WHEN	Elindítom a programot
	THEN	A program hibaként kiírja, hogy adjunk meg egy pozitív egészet a mező értékeként.
18	GIVEN	Negatív játékosszámot adok meg
	WHEN	Elindítom a programot
	THEN	A program hibaként kiírja, hogy legalább 1 játékos legyen.
19	GIVEN	Játékosszámként nem egy egész szám értéket adok meg.
	WHEN	Elindítom a programot
	THEN	A program hibaként kiírja, hogy pozitív egészként adjam meg a játékosszámot.

20	GIVEN	Hibás értéket adok meg játékos típusként.
	WHEN	Elindítom a programot
	THEN	A program hibaként kiírja, hogy nem található az adott játékos típus.
21	GIVEN	Nem adom meg egy játékos nevét.
	WHEN	Elindítom a programot
	THEN	A program hibaként kiírja, hogy nem található az adott játékos típus.
22	GIVEN	Túl kevés kockadobás értéket adok meg, és nem csak egy játékos marad minden lépés után
	WHEN	Elindítom a programot
	THEN	A program kiválasztja a megmaradt játékosok közül a legtöbb tőkével rendelkezőt mint győztest, de azt is kiírja, hogy a játéknak nem lett teljesen vége, és több játékos maradt még a játékba.
23	GIVEN	Túl sok kockadobás értéket adok meg, és a játéknak vége mielőtt minden adott lépés le lett volna lépve.
	WHEN	Elindítom a programot
	THEN	A program jelzi, hogy van még lépés, de csak egy játékos maradt, majd megadja a nyertes játékost.
24	GIVEN	Olyan bemeneti konfigurációt adok meg, hogy sosincs vége a játéknak. (Pl: csak szerencse mezők vannak)
	WHEN	Elindítom a programot
	THEN	A program megszakítja a szimulációt 1000 kör után, ezt jelzi, majd megadja a legnagyobb tőkével rendelkező játékost, mint győztest.