**Szoftvertechnológia**

**Házi feladat:**  
Potyogós

Készítette:  
Hoffmann Ákos

S8F9EN

1. Bevezető

Ebben a dokumentációban a Downfall nevű játékhoz készített modell és annak leírása található. A játék eredeti szabályait pár helyen módosítottam, de ezek egyértelműen dokumentálva vannak a követelményeknél.

1. A játék leírása (feladatkiírás)

A játékot N játékos játssza (N >= 2, de tetszőlegesen nagy egész szám lehet). A játék egy N oldalú függőleges táblából áll, melybe forgatható tárcsák vannak beépítve. A tárcsák a tábla minden oldalán látszanak. Ha egy tárcsa forog, a tárcsa minden oldala együtt forog. A tárcsák szélén kis zsebek vannak.

Egy tárcsa egyes oldalain a zsebek száma, illetve az zsebek pozíciója különböző is lehet. Minden játékos néhány számozott koronggal (érmével) kezd a tábla tetején. Egy kis zsebbe pontosan egy korong illeszkedik. Ha két érintkező tárcsa zsebei egymás mellé kerülnek, a felső zsebben lévő korong átpottyan az alsó zsebbe.

A játék célja az érmék eljuttatása a tábla tetejéről a tábla alján lévő tálcába a tárcsák forgatásával. A játékosok felváltva forgatnak egy-egy tárcsát, de az a tárcsa nem mozgatható, amelyet az előző játékos éppen mozgatott. Amíg a játékos soron van, a kiválasztott tárcsát bármilyen pozícióba forgathatja, akár több irányba is és akár többször is.

A győztes az a játékos, akinek először pottyan le az összes érméje a tálcába. A korongoknak a számozás sorrendjében kell leérkezniük. Mivel egyik játékos sem látja a többi játékos tábláját, gyakran előfordulhat, hogy segítik vagy éppen hátráltatják a többiek előrehaladását.

A játék arra késztet, hogy előre tervezzünk: belehajszolhatjuk a többi játékost, hogy a saját tárcsája forgatásával a mi korongjainkat vigye tovább, de közben vigyázzunk arra, nehogy rossz sorrendben potyogjanak le a korongjaink a tálcára.

1. Funkcionális követelmények

|  |  |  |
| --- | --- | --- |
| **ID** | **Leírás** | **Use-case** |
| 1 | több játékos lehet (>=2) | Use Case |
| 2 | a játékosok meghatározott sorrend szerint követik egymást | Use Case |
| 3 | azonos tárcsák egyszerre mozognak a játékosok oldalain | Tárcsa forgatása |
| 4 | a játékosok nem látják egymás tábláit | Use Case |
| 5 | egy játékos egyszerre csak egy tárcsát forgathat | Tárcsa forgatása |
| 6 | egy tárcsát többször, többirányba lehet forgatni | Tárcsa forgatása |
| 7 | A táblák száma megegyezik a játékosok számával | Use Case |
| 8 | A tárcsákat többször több irányba lehet forgatni | Tárcsa forgatása |
| 9 | Az előző körben forgatott tárcsát nem lehet újra forgatni | Tárcsa forgatása |
| 10 | A korongok meg vannak számozva | - |
| 11 | A játékosoknak nem kötelező lépnie | Kör befejezése |
| 12 | Egy játékos akkor nyer, ha az összes korongja sorrendben leér | Korong leesik a tálcába |
| 13 | Egy korong akkor esik le, ha az alatta lévő tárcsa zsebében nincsen korong és a két tárcsa zsebe egymás felé néz | Korong leesik az alatta lévő tárcsába |
| 14 | A korong a pálya alján leesik egy tálcába | Korong leesik a tálcába |

* 1. További saját követelmények

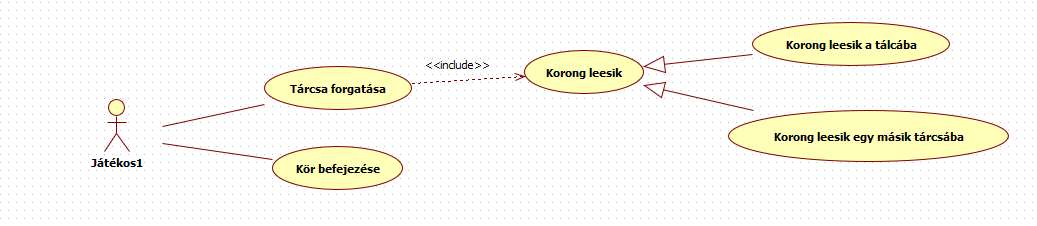
|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Leírás** | | **Use-case** |
| 1 | | 10 korongja van mindenkinek | Játékos | |
| 2 | | Minden táblán minimum 4 tárcsa de legalább 10 van | Tárcsa forgatása | |
| 3 | | Minden tárcsán minimum 1 de maximum 6 zseb vang | Tárcsa forgatása | |
| 4 | | Az utolsó tárcsáról akkor esik ki egy zsebből egy korong ha pontosan lefele néz a zseb | Korong leesik a tálcába | |
| 5 | | Az első tálcába pontosan fentről tud beesni egy korong a kezdeti tálcából. | Korong leesik egy másik tárcsába | |

1. Előzőleges összefoglalás:

Felhasználandó jelölések:

* **Játékos:**
  + Egy ember vagy gép amely a játékkal lép interakcióba úgymond *játsza* a játékot
* **Játék:**
  + A megjelenítendő felület, melyen vannak a korongok és tárcsák
* **Korong:**
  + Minden játékosnak van 10 darab, ezeknek kell lejutniuk a tábla tetejéről a tábla aljára, a tárcsák forgatásával, és zsebből-zsebbe eséssel
* **Tárcsa:**
  + Ezeket tudja a játékos forgatni, a táblán és ezeken találhatóak meg a zsebek
* **Zseb:**
  + Ezekbe tud beleesni egy-egy korong

1. Use-case
   1. Részletes Use Case a játék játszásáról:



|  |  |
| --- | --- |
| **Use case:** | Use Case a játék játszásáról |
| **Résztvevők:** | Játékos |
| **Leírás:** | A Use Case leírja hogy egy játékos miképpen tud játszani a játékkal.  Egyik lehetőség hogy befejezi a körét, ezzel a következő játékos jön majd.  Másik lehetőségek, hogy forgatja a tárcsát. Ha forgat egy tárcsát annak lehet következménye, hogy a korong helyzetet változtat és leesik egy zsebből egy másikba, vagy végleg kiesik a tábláról az a korong. |

* 1. Use-case leírások

|  |  |
| --- | --- |
| **Cím** | Tárcsa forgatása |
| **Leírás** | Egy játékos forgat a kiválasztott tárcsáján tetszőleges irányokba tetszőleges mennyiségű szöget |
| **Aktorok** | Játékos |
| **Főforgatókönyv** | A tárcsa elfordul, így minden játékos azonos tárcsája is fordul |
| **Alternatív forgatókönyv** | A tárcsa elfordul, így minden játékos azonos tárcsája is fordul és valamennyi zsebből kiesik egy korong |
| **Alternatív forgatókönyv** | A tárcsa elfordul, így minden játékos azonos tárcsája is fordul és valamennyi zsebbe beesik egy korong |

|  |  |
| --- | --- |
| **Cím** | Korong leesik |
| **Leírás** | Le ellenőrzi a gép, hogy a zseb milyen helyzetben áll |
| **Aktorok** | Gép |
| **Főforgatókönyv** | A zseb a vízszinteshez képest nem néz lefelé, így nem tud kiesni belőle a korong |
| **Alternatív forgatókönyv** | A zseb a vízszinteshez képest lefelé néz, így ki tud esni belőle a korong, de nincs másik zseb és nincs is vége a táblának, ezért nem esik ki belőle |
| **Alternatív forgatókönyv** | A zseb a vízszinteshez képest lefelé néz, így ki tud esni belőle a korong mert van egy másik zseb, amibe bele tud esni |
| **Alternatív forgatókönyv** | A zseb a vízszinteshez képest lefelé néz, így ki tud esni belőle a korong és vége van a pályának |

|  |  |
| --- | --- |
| **Cím** | Korong leesik a tálcába |
| **Leírás** | Egy korong kiesik a zsebből egy másik zsebbe |
| **Aktorok** | Mivel a való életben a gravitáció csinálja ezt, jelen esetben a számítógép végzi el a szükséges műveleteket |
| **Főforgatókönyv** | Egy zseb a megfelelő pozícióban van és az alatta levő zseb is jó pozícióban van, így a felső zsebből ki tud esni a korong az alsó zsebbe |

|  |  |
| --- | --- |
| **Cím** | A korong leesik a tálcába |
| **Leírás** | Egy tárcsa zsebe úgy áll hogy ki tud esni belőle lefele egy korong és alatta nem egy zseb van, hanem a tábla vége |
| **Aktorok** | Mivel a való életben a gravitáció csinálja ezt, jelen esetben a számítógép végzi el a szükséges műveleteket |
| **Főforgatókönyv** | A korong sikeresen kiesik a tábláról a megfelelő sorrendben, így a játékos pontot kap |
| **Alternatív forgatókönyv** | A korong sikeresen kiesik a tábláról nem megfelelő sorrendben, így a játékos kiesik a játékból |

|  |  |
| --- | --- |
| **Cím** | Kör befejezése |
| **Leírás** | Egy játékos úgy dönt hogy nem csinál már mást a körében |
| **Aktorok** | Játékos |
| **Főforgatókönyv** | Egy játékos nem csinál semmit, így a gép tovább lép a következő játékosra |

1. Strukturális leírás
   1. Az osztályok leírása
      1. Játék

**Felelősségek:**

A játék menetét kezeli, tárolja és kezeli a játékosokat és tábláikat, valamint az éppen soron levő játékost és egy-egy kör lebonyolítását indítja és kezeli.

**Attribútumok:**

|  |  |
| --- | --- |
| -Jatekosok: jatekos[2..\*] | Ebben a tömbben van tárolva a játékban lévő játékosok, minimum 2 ember kell hogy játszák, de végtelen sok lehet |
| -Aktiv\_Jatekos: Jatekos | A soron lévő játékost tároljuk mindig ebben, mindig cserélődik ahogy egy játékos végez a körével |
| -Tablak: Tabla[2..\*] | Ebben a tömbben vannak tárolva a játékosok táblái, mindig ugyanannyi van mint ahány játékos. (index szerint vannak párosítva) |

**Metódusok:**

|  |  |
| --- | --- |
| +Kor(): void | Egy kör lezajlásáért felelős függvény. |
| +Jatek\_Vege(): void | Lezárja a játékot és összegzi azt, valamint kiírja hogy ki nyert. |
| +Uj\_Jatek(Jatekosok:Jatekos, Kezdo\_jatekos: Jatekos) | Játék kezdetekor beállítja a játékosokat, valamint hogy ki kezdi a játékot |
| +Tarcsa\_forgatas(T1: Tarcsa, Fok: Double): void | Egy tárcsát forgat egy megadott szöggel minden táblán. |

* + 1. Játékos

**Felelősségek:**

Saját tábla és korongok kezelése, valamint a játékos műveleteinek bevitele (tárcsa forgatása, új korong bedobása, kör vége jelzése)

**Attribútumok:**

|  |  |
| --- | --- |
| -Leesett\_Korongok: Korong[0..10] | Tárolja hogy hány melyik korongok estek már le sorrendben |
| -Sajat\_tabla: Tabla | A játékos saját táblája |
| -Aktiv\_korongok: Korong[0..10] | Éppen melyik korongok vannak bent a játékban |
| +pontok: int | Tárolja hogy az adott játékosnak mennyi pontja van |

**Metódusok:**

|  |  |
| --- | --- |
| +KorVege(): void | A játékos ezzel jelzi hogy vége a körének |
| +AddPoint(x: int): void | növelni lehet egy játékos pontját |
| +Add:Leesett(k: Korong): void | Hozzáad egy korongot a már leesett koronghoz |
| -Uj\_Korong(): void | Új korongot dob be a játékba |

* + 1. Korong

**Felelősségek:**

Tárolja az egyedi azonoítóját a korongnak.

**Attribútumok:**

|  |  |
| --- | --- |
| -ID: int | Egy azonosító a korongnak, hogy tudjuk kié és hányadik. |

**Metódusok:**

**-**

* + 1. Tábla

**Felelősségek:**

Tárolja a táblán lévő tárcsákat, melyik tárcsa volt az előző körben forgatva, illetve a tárcsák forgatását is kezeli és figyeli forgatás közben hogy esik-e egy korong lejjebb.

**Attribútumok:**

|  |  |
| --- | --- |
| +Tarolo: Tarolo[4..10] | Ebben tároljuk hogy a táblán milyen sorrendben hány darab tárcsa van |
| +Előző\_tárcsa: Tarcsa | Az előző körben forgatott tárcsa, fontos tárolni mert nem szabad egymás utáni körben ugyanazt a tárcsát forgatni |

**Metódusok:**

|  |  |
| --- | --- |
| +Forgat(Tarcsa1: Tarcsa, Fok: double): void | Egy bizonyos tárcsát forgat a táblán bizonyos szöggel, és figyeli hogy a tárcsa találkozik-e a szomszédjával a megadott szögnél ahol át tud esni és ha a szomszédja is abban a szögnél áll és az ott álló zseb üres akkor elindítja a labda cserét. |

* + 1. Tároló

**Felelősségek:**

Definiálja a tároló osztályt a tárcsáknak és zsebeknek. Itt tudjuk mindkettő leszármazottnak a helyét, méretét és tudjuk ezeket lekérdezni.

**Attribútumok:**

|  |  |
| --- | --- |
| +Pozicio: double | Relatív pozició, hogy éppen hogy áll. (Felfele 0 fok és 360 fokban lehet körbe menni) |
| +Meret: int | Megadható hogy hány darabot tud tárolni. Zseb esetén korongra értendő, tárcsa esetén zsebre. Ezzel van megoldva a kezdeti tároló is, ami egy fix tárcsa, feltöltve mind a 10 koronggal és az alsó tálca is ami szintén egy fix tárcsa, 10 darab korongnak. |

**Metódusok:**

|  |  |
| --- | --- |
| +Set\_Pos(Fok: double): void | Forgatja a tárolót bizonyos szöggel |
| +getPos(): double | Lekérhető a tároló mindenkori állása |

* + 1. Tárcsa

**Felelősségek:**

Tárolja a saját pozícióját és el tudjuk forgatni bizonyos szöggel, valamint a tárcsán található zsebeket is tároljuk és a tárcsával szomszédos tárcsákat a hozzátartozú szögükkel hogy mikor tud átesni egy korong.

**Attribútumok:**

|  |  |
| --- | --- |
| +Zsebek: zseb[1..Meret] | A tárcsán lévő zsebeket tárolja, minimum 1 darab van, de maximum 6 lehet egy tárcsán |
| +Szomszed\_tarcsak: Tarcsa[1..\*] | Tárolja hogy mely tárcsák a szomszédjai |
| +Szomszed\_szogek: double[1..\*] | A tárolt szomszédos tárcsákhoz adott szög hogy melyik állásban tud átesni a korong |

**Metódusok:**

|  |  |
| --- | --- |
| +Add\_Szomszed(Tarcsa1: Tarcsa, x: int): void | Hozzá lehet adni egy szomszédos tárcsát melybe át tud esni a korong, ha a megadott szögben áll. |

* + 1. Zseb

**Felelősségek:**

Tárolja a saját pozícióját a tárcsán és hogy van-e benne korong vagy sem, ha igen akkor melyik.

**Attribútumok:**

|  |  |
| --- | --- |
| +Pozicio: double | Tárolja a relatív pozícióját a tárcsán |
| -Korong\_ID: Korong[0..Meret] | Ez alapján tudjuk hogy van-e benne korong vagy sem és ha van akkor melyik(ek) |

**Metódusok:**

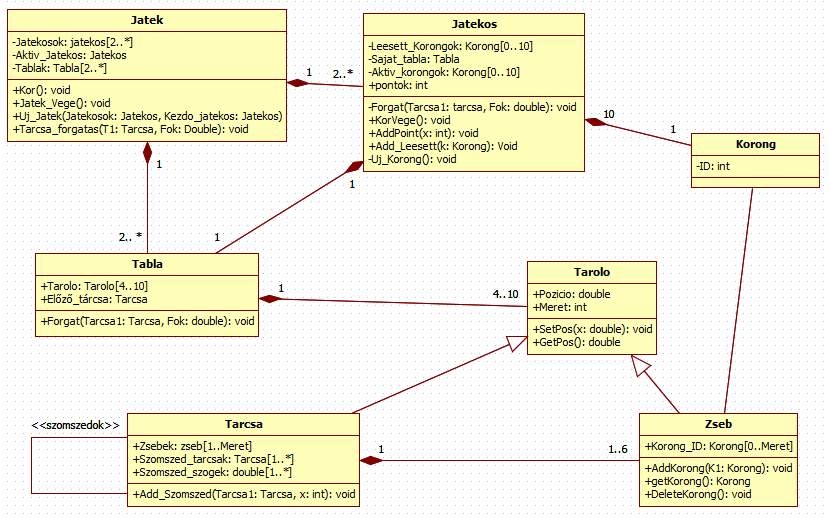
|  |  |
| --- | --- |
| +AddKorong(K1: Korong): void | Bele tudunk tenni egy korongot a zsebbe |
| +GetKorong(): Koroing | Lekérhető hogy melyik korong van benne |
| -DeleteKorong(): void | El tudunk távolítani egy korongot a zsebből |

* 1. Osztálydiagram

Van egy minimálisan elvárt magja a feladatnak: a legfontosabb osztályok és a közöttük lévő megfelelő kapcsolatok (öröklődés, asszociációk, függőségek).

Súlyos hibák:

* Nincsenek meg a minimális magot alkotó osztályok és a kapcsolataik
* Szintaktikailag hibás diagram
  + öröklés iránya rossz
  + kompozíció iránya rossz
  + asszociációk hiányoznak
  + függőségek hiányoznak (pl. függvényparaméterek típusaitól)
* Nem alkot összefüggő gráfot a diagram
* Nem objektumorientált megoldás: felelősségek rosszul szétosztva
* Típuslekérdezés: instanceof, int, enum, is...(), can...()
* Egy nagyon okos osztály irányít mindenkit, a többiek csak adatokat tárolnak
* Üres leszármazott osztály, amely őse nem absztrakt osztály vagy interfész
* Koordináták, grafika, stb.



1. Viselkedés leírása
   1. Szekvencia diagramok

Minimum annyi legyen, hogy lefedjenek minden osztályt és függvényt az osztálydiagramból. Van egy minimálisan elvárt készlet a szekvenciákból, amely a feladat alapvető működését demonstrálja.

Súlyos hibák:

* Szintaktikailag hibás diagram
* A szekvencia diagramba több pontból lépünk be párhuzamosan
* Nem aktív objektum kezdeményez hívást
* Interfész vagy absztrakt függvény kezdeményez hívást
* Osztály vagy függvény nem létezik az osztálydiagramon
* A hívó nem ismeri a hívott objektumot (dependencia kevés, meg is kell kapnia)
* A hívó olyan függvényt hív, amelyet a hívott objektum ismert típusa alapján nem láthatna (őstípusát kapja, leszármazottban deklarált metódust hív)
* Ugyanaz a függvény különböző diagramokon mást csinál
* Elágazás/feltételvizsgálat típusra: int, enum, is...(), instanceof
* Egy nagyon okos osztály irányít mindenkit
* Csak kérdezgetések vannak (get/set), nincs interakció
* Túl sok minden van ábrázolva egy diagramon
* Értelmetlen diagram
* Egymással inkonzisztens diagramok
* Osztálydiagrammal inkonzisztens diagramok
  1. Állapotgép diagramok

Csak ha a feladatban van értelmük.

1. Napló

|  |  |  |  |
| --- | --- | --- | --- |
| Időpont | Időtartam | Elvégzett munka | Hivatkozás |
| 2019.11.01 | 2 óra | A feladatot a leírással és az útmutatók elolvasásával kezdtem, majd online kerestem a downfall game-ről egy játék útmutatót, hogy a maradék szabályt is megértsem és jól tudjam modellezni. Ezek után elkészítettem a dokumentáció sablonját és bemásolta a feladat leírását. A követelményeket megírtam amikből majd dolgozni fogok és összegyűjtöttem az addigi forrásaimat. | 1., 2., 3., 9. |
| 2019.11.02 | 3 óra | A use case-eket elkészítettem és megírtam hozzá a dokumentációt és a táblázatokat. | 4., 5. |
| 2019.11.04 | 3 óra | Megcsináltam az osztálydiagramokat is és a hozzá tartozó dokumentációt. (táblázatok, minden osztály leírása stb.) | 6. |
| 2019.11.06 | 5 óra | Szekvenciadiagramok elkészítése | 7. |

**Összes elvégzett óra:**

**Modellező eszköz:** WhiteStar UML

**Egyéb eszközök:** Word

**Megjegyzés:**

1. Forrás:

<https://www.iit.bme.hu/system/files/uploads/module_files/StHf2019.pdf>

<https://www.iit.bme.hu/system/files/uploads/module_files/SeLesson01_UmlHu.pdf>

<https://www.iit.bme.hu/system/files/uploads/module_files/SeLesson02_UmlHu.pdf>

<https://www.iit.bme.hu/system/files/uploads/module_files/SeLesson03_UmlHu.pdf>

<https://www.iit.bme.hu/targyak/BMEVIIIAB01>

<https://www.iit.bme.hu/system/files/uploads/module_files/SePractice01_LogisticsHu.pdf>

<https://www.iit.bme.hu/system/files/uploads/module_files/SzofttechJavitas.pdf>

https://www.iit.bme.hu/system/files/uploads/module\_files/StHf\_Pacman.pdf

http://www.geekyhobbies.com/downfall-board-game-review-and-rules/