13. Összes dokumentáció

61- BCSV

Konzulens:
Simon Balázs

Csapattagok

Csoma Zoltán	G0IOFT	zolyboy98@gmail.com
Sipula László Márk	A1D4QD	sipulalaci@gmail.com
Szepesi-Nagy István	K45SFS	szni98@gmail.com
Varga Zsombor	AKCJOP	zsombor.varga4@gmail.com
Ziaja Bálint	HICYBO	balint@ziaja.pro

2. Követelmény, projekt, funkcionalitás

2.1 Bevezetés

2.1.1 Cél

A dokumentum célja, hogy a projekt feladat felépítése, funkciói, megvalósítása és működése egy laikus felhasználó és a feladat CRUD résztvevői (konzulens, tárgyfelelős, fejlesztők) számára is érthetőek legyenek.

2.1.2 Szakterület

A program játékszoftver területre van fejlesztve. A játék célközönsége átlagos felhasználók, semmilyen szaktudást nem igényel. Célja a játékosok gondolkodásra sarkallása és szórakoztatása.

2.1.3 Definíciók, rövidítések

CRUD résztvevő - Create Read Update Delete; vagyis a Készítési, Olvasási, Módosítási és Törlési jogokkal rendelkező felhasználó.
Csapat - Alkalmilag összeállt, közös céllal csoport bizonyos közös tevékenységre.
Beállítások - Paraméterek meghatározása.
Dokumentum - információkat tartalmazó irat
Diagram - **Rajz, ábra** stb., ami segít megmagyarázni egy dolgot úgy, hogy megmutatja a részeit, a kialakítását és a működését.
Discord - Kommunikációs szoftver, amivel hangot, képet, videót és fájlokat lehet küldeni a többi résztvevőnek.
Draw.io - Online diagramszerkesztő alkalmazás.
Eclipse - Integrált fejlesztőkörnyezet főként JAVA alkalmazások fejlesztéséhez.
Egység - Konkrét mértékegység nélküli mérőszám.
Fejlesztő - A (jelen esetben) szoftver megvalósítását végző személy.
Github - Szoftverek fejlesztésekor használt verziókezelő rendszer.
Google Documents - Online dokumentum szerkesztő alkalmazás, amivel a változások azonnal látszódnak a dokumentumhoz hozzáférő embereknek.
Játékszoftver - Olyan szoftver, aminek célja, hogy a felhasználó játszon vele, szórakoztassa.
Java - Általános célú, objektum orientált magas szintű programozási nyelv.
Konzulens - Aki szakmai oldalról mélyen ismeri a közvetített tananyagot és elsősorban szakmai oldalról nyújt támogatást az ismeretek elsajátításában
Laikus - Nem szakképzett; hozzá nem értő
Menü - Különböző lehetőségeket összefoglaló táblázatszerű navigáló felület, aminek pontjaival egy program alrészeibe juthatunk.
Messenger - Üzenetküldő alkalmazás
Use-case Diagram - A felhasználó és a szoftver közötti kapcsolatokat leíró diagram.
Program - Elvégzendő lépések sorozata, amelyet vagy egy személy, közösség, vagy egy elektronikus berendezés végez el.

Projekt - A **projekt** egyszeri vállalkozás, melynek célja, hogy egyedi terméket, szolgáltatást vagy végeredményt hozzon létre.
Számítógép - Berendezés, amely képes bemenő adatok fogadására, ezeken végzett programok végrehajtására, eredmények visszaadására.
Szoftver - A **szoftver** alatt a legszűkebb értelemben elektronikus adatfeldolgozó berendezések (például számítógépek) memóriájában elhelyezkedő, azokat működtető programokat értünk.
Tárgyfelelős - A tárgy lebonyolításával megbízott egyetemi oktató.
Telepítés - az a folyamat, amikor egy alkalmazás vagy operációs rendszer a számítógépre kerül.
Online - Internetet használó, azon keresztül működő program.
Windows 10 - Felhasználóbarát grafikus operációs rendszer.

2.1.4 Hivatkozások

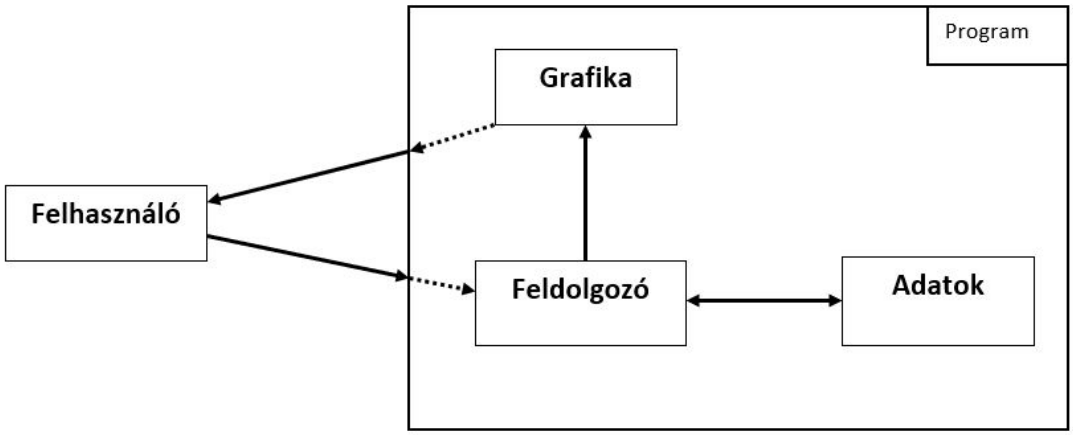
- Discord - https://discordapp.com/
- Draw.io - https://www.draw.io/
- GitHub - https://github.com/
- Google Dokumentumok - https://docs.google.com/
- Idegen szavak - https://idegen-szavak.hu/
- Tárgyhonlap - https://www.iit.bme.hu/targyak/BMEVIIIAB02
- Wikipédia - https://hu.wikipedia.org/wiki/Wiki

2.1.5 Összefoglalás

A dokumentum további részeiben funkcionális, nem-funkcionális, átadási, erőforrási követelmények, funkciók, korlátozások, kapcsolatok, use-case-ek, a projekt terve, és egy szótár van részletezve.

2.2 Áttekintés

2.2.1 Általános áttekintés



A **grafika** dolga, hogy a Felhasználó lássa a játékot, a megjelenítés a játék szempontjából az egyik legfontosabb elem, anélkül a felhasználó, nem tudja mit csinál.

A **feldolgozó** az, ami a felhasználó kéréseit (irányítás gombokkal és menüpont választás) figyeli és teljesíti. A tárolt adatokból frissíti a grafika állapotát, ami így megjelenítheti a felhasználó számára a jelenlegi állapotát.

Az **adatok** felelős az eddigi állapot elmentéséért. A fontosabb adatok például a játékosok helyzete, tárgyai és élete, a jégtáblák helyzete, teherbírása és belefagyott tárgyai, és az igluk helyzetei.

A **felhasználó** gombok segítségével a program tudtára adja, hogy ő mit szeretne cselekedni, ezt a program feldolgozza és megjeleníti számára a megfelelő pályát. A felhasználó számára a program, egy “fekete doboz”, ő csak gombokat nyomkod és nézi a pálya állapotának alakulását, ahhoz, hogy ezt a program hogyan oldja meg számára érdektelen.

2.2.2 Funkciók

A program egy egycsapatos, 3 vagy több játékos számára tervezett játékszoftver. A játék nem online, így egy számítógépen kell játszania a játékosoknak. A játék célja, hogy a csapat tagjai mind túléljék a játék rejtette meglepetéseket, csapdákat és összegyűjtsék egy rakétapisztoly alkatrészeit, majd elsüssék azt. A csapat tagjai lehetnek eszkimók és/vagy sarkkutatók.

A játék indítását követően megjelenik a menü, ahonnan gombok segítségével az Új Játék, Kilépés és egyéb menüpontok között lehet választani. A játékot bármikor meg lehet állítani menet közben, ilyenkor egy játékos sem cselekedhet.

A játék egy tengerrel körülvett jégtáblán helyezkedik el, melyet mezők alkotnak. Egy mező lehet stabil, instabil vagy lyukas. A stabil mezőn bármennyien állhatnak, az instabil mező csak egy maximális létszámú embert bír el, ha ezt meghaladja, a mező átfordul, és a rajta állók beleesnek a hideg vízbe és meghalnak. A lyukas mező egy embert sem tart meg, de fedheti hó, amitől nem tudja a játékos, hogy lyukas. Ha egy ember beleesik a lyukba meghalhat, de van esélye a túlélésre. A mezőket kezdetben különböző mennyiségű hó borítja.

A jégtáblán feltámadhat hóvihar. Ilyenkor a hóvihar által sújtott területeket újabb különböző mennyiségű friss hó fedi be, még a lyukakat is. Akit elkap a hóvihar, annak a testhője eggyel csökken. Egy eszkimónak 5, egy sarkkutatónak 4 egység testhője van alapból.
Egy cselekvő egy körben 4 egységnyi munkát végezhet. Egy munkaegység lehet ásás, lépés, eszközhasználat vagy egy sajátos cselekedet. Az eszkimók sajátos cselekedete az iglu építés. Aki az igluban tartózkodik, az védve van a hóvihartól. Egy iglu egy mezőnyi helyet foglal el, aki a mezőre lép, az egyúttal az igluba is belép, nem kell külön munkaegységet végezni a belépéshez. Egy iglu 4 kör után elolvad, hóvihar után megsemmisül. Ha egy csapattag tartózkodik már az igluban, a mező akadálynak minősül a többiek számára, áthaladni sem lehet rajta, mivel csak egy ember tartózkodhat bent. Iglu szabadkézzel két, lapáttal egy munkaegységgel lebontható. A sarkkutatók sajátos cselekedete egy szomszédos, akár hóval borított mező teherbírásának vizsgálata, vagyis hogy az adott mező hány embert bír el, mielőtt átfordulna (stabil mező soha nem fordul át). Mivel az iglukat használhatják a sarkkutatók is, a sarkkutatók által megvizsgált mezők teherbírását is látják az eszkimók.
Néhány jégmezőbe egy-egy befagyott tárgy helyezkedhet el. A játékosok használhatják ezeket a tárgyakat, ha kiássák őket. Tárgyat csak akkor lehet észrevenni és kiásni, ha nem fedi a mezőt hó. Az egységnyi hó eltakarítása egy munkaegységet igényel. Ilyen tárgy lehet élelem, búvárruha, kötél, lapát, titkos eszköz vagy a rakétapisztoly alkatrésze. Az élelemmel visszatölthető egy testhő. A lapáttal egy munkaegységért dupla annyi havat tisztíthatunk el, mint kézzel. Ha valaki búvárruhával esik lyukas mezőnél vízbe, akkor nem fullad meg. Ha a lyukba esett emberen nincs búvárruha, csak a mellette lévő játékos mentheti meg a kötéllel, különben a játékos meghal. Egy játékosnál egyszerre csak egy használható eszköz/tárgy lehet, ha újat szeretne felvenni, a már nála lévőt el kell dobnia az aktuális mezőre. Ha hó kerül egy már kiásott, de még fel nem vett vagy eldobott tárgyra, arról újra el kell takarítani a havat, hogy újra látható és felvehető legyen, de nem fagy be újra. A jelzőrakéta részeit nem lehet önmagukban alkalmazni semmire, mind a három alkatrészt egy mezőre kell vinni, majd ott a teljes csapat egy munkaráfordítással összeszerelheti és elsütheti, ezzel megnyerve a játékot.

Ha a csapat bármely tagja meghal (megfullad vagy elfogy a testhője), a csapat elveszíti a játékot.

A játékmező mérete bizonyos játékosszám után nő egy előre meghatározott mérettel. A tárgyak véletlenszerűen vannak elhelyezve a pályán. A kezdőlépés véletlenszerűen oszlik el a csapattagok között, majd felváltva egy-egy ember léphet. A játékosok bármeddig gondolkodhatnak a lépésükön, nem időre mennek a körök. A lépés csak 4 (észak, kelet, dél, nyugat) irányba lehetséges.

2.2.3 Felhasználók

A felhasználókra nincs semmilyen kikötés megfogalmazva. A játékkal bárki játszhat kortól és nemtől függetlenül. A felhasználóknak nincsen szükségük előképzettségre, a játékleírás elolvasása után bárki számára könnyen érthetővé válik a szoftver használata.

2.2.4 Korlátozások

Nincs szükség internetre, csak egy számítógépre, amin Windows 10 és 8-as Java alatt jól fog működni a játék. Nem egy időben lépnek a játékosok, ezért nincs szükség emiatt semmilyen további korlátozásra.

2.2.5 Feltételezések, kapcsolatok

Discord - Konferencia beszélgetések lebonyolításához kellett
Draw.io - Modellek, diagramok készítésében volt szükséges
GitHub - A dokumentumok verzióinak kezelésére használtuk
Google Dokumentumok - A dokumentumok verzióinak kezelésére használtuk
Idegen szavak - Szótári szavak és definíciók pontos leírására használtuk
Tárgyhonlap - Követelmények, feladatkiírás, segítségek forrása
Wikipédia - Szótári szavak és definíciók pontos leírására használtuk

2.3 Követelmények

2.3.1 Funkcionális követelmények

Azonosító	Leírás	Ellenőrzés	Prioritás	Forrás	Use-case	Komment
R01	Legyen pálya, ami egy jégmezőt reprezentál.	Bemutatás során	Alapvető	Feladatkiírás	CreateTable	“A jégmező jégtáblákból áll.”
R02	A pályát határolja tenger.	Bemutatás során	Alapvető	Feladatkiírás	CreateTable	“egy tengerrel körülvett jégmezőn túlélniük.”
R03	A pályát különböző típusú jégtáblák alkossák.	Bemutatás során	Alapvető	Feladatkiírás	CreateTable	“Vannak stabil jégtáblák, [..] és vannak instabil jégtáblák,”
R04	Kezdetben a jégtáblákat különböző mennyiségű hó borítsa.	Bemutatás során	Alapvető	Feladatkiírás	CreateTable	“A jégtáblákat a játék kezdetén eltérő mennyiségű hó borítja.”
R05	A jégtáblák között legyen stabil, amin akárhány szereplő állhat.	Bemutatás során	Alapvető	Feladatkiírás	CreateTable	“stabil jégtáblák, amelyeken akárhány szereplő állhat,”
R06	A jégtáblák között legyen instabil, amely adott létszám felett átfordul és a szereplők meghalnak.	Bemutatás során	Alapvető	Feladatkiírás	CreateTable	“instabil jégtáblák, amik adott létszám felett átfordulnak”
R07	A jégtáblák között legyen lyukas, amire lépve	Bemutatás során	Alapvető	Feladatkiírás	CreateTable	“hóval fedett lukak is, amibe

	rögtön a tengerbe esik a szereplő.					beleesve [..]”
R08	A tengerbe esett szereplők meghalhatnak.	Bemutatás során	Fontos	Feladatkiírás	BaseAction	“csak a búvárruhát viselők élnek túl” vagy kötelesek
R09	Legyenek különböző szereplők, különböző képességekkel.	Bemutatás során	Alapvető	Feladatkiírás	CreateTable, SpecificAction	“A játékban a különböző képességű szereplőknek”
R10	A különböző szereplők egy csapatot alkotnak és a túlélésért játszanak.	Bemutatás során	Opcionális	Csapat	CreateTable, ViewTable	-
R11	Legyenek Eszkimók, akiknek eredetileg 5 egység testhőjük van.	Bemutatás során	Alapvető	Feladatkiírás	CreateTable, ViewTable	“Az eszkimóknak a játék elején 5 egység testhője van,”
R12	Legyenek Sarkkutatók, akiknek eredetileg 4 egység testhőjük van.	Bemutatás során	Alapvető	Feladatkiírás	CreateTable, ViewTable	“[..] sarkkutatónak csak 4.”
R13	Az Eszkimók legyenek képesek iglut építeni, de a Sarkkutatók ne.	Bemutatás során	Alapvető	Feladatkiírás	SpecificAction	“eszkimó tud iglut építeni,”
R14	A Sarkkutatók legyenek képesek ellenőrizni a szomszédos jégtáblák teherbírását, de az Eszkimók ne.	Bemutatás során	Alapvető	Feladatkiírás	SpecificAction	“sarkkutató meg tudja nézni, hogy az a jégtábla, amire lépne, hány embert bír el”

R15	Legyenek különböző kiásható tárgyak a játékban, amikkel tevékenységeket hajthatnak végre a játékosok.	Bemutatás során	Alapvető	Feladatkiírás	CreateTable, ViewTable, BaseAction	“különféle tárgyak lehetnek belefagyva:”
R16	Egy körben négy egységnyi munkát végezhet el egy játékos.	Bemutatás során	Fontos	Feladatkiírás	ViewTable, ActionWithCharacters	“Minden szereplő egy körben 4 egységnyi munkát végezhet.”
R17	A játék folyamán véletlenszerűen hóvihar törhet ki.	Bemutatás során	Fontos	Feladatkiírás	SnowStorm	“A jégmezőn időnként feltámad a hóvihar, [..]”
R18	A hóvihar befed egyes mezőket különböző mennyiségű friss hóval.	Bemutatás során	Alapvető	Feladatkiírás	SnowStorm, ViewTable	“néhány érintett jégtáblát újabb adag friss hóval borít be.”
R19	Amennyiben olyan mezőt érint a hóvihar, ahol játékos áll, annak a játékosnak a testhője 1 egységnyit csökken.	Bemutatás során	Fontos	Feladatkiírás	SnowStorm, ViewTable	“Akit elkap, annak a testhője egységnyivel csökken.”
R20	A tárgyak csak akkor legyenek láthatóak és kiáshatóak, ha fölöttük nincs hó.	Bemutatás során	Fontos	Feladatkiírás	ViewTable, BaseAction	“Befagyott tárgyat csak akkor lehet meglátni és kiásni, ha a jégtábla tiszta,”
R21	Legyen lapát, mint eszköz, amivel két egységnyi havat lehet ellapátolni	Bemutatás során	Alapvető	Feladatkiírás és Csapat	CreateTable, ViewTable,	“A lapáttal két egységnyi hó

	vagy egy munkaegységgel egy iglut lebontani.				BaseAction	takarítható el”
R22	Legyen kötél, mint eszköz, amivel vízbe esett társunkat lehet kihúzni.	Bemutatás során	Alapvető	Feladatkiírás	CreateTable, ViewTable, BaseAction	“köteles barátjuk a szomszéd jégtábláról azonnal kimenekít.”
R23	Legyen búvárruha, mint eszköz, aminek a segítségével nem fulladunk meg, ha beleesünk a vízbe.	Bemutatás során	Alapvető	Feladatkiírás	CreateTable, ViewTable, BaseAction	“beleesve csak a búvárruhát viselők élnek túl,”
R24	Legyen élelem, mint eszköz, aminek a segítségével növelhetjük a testhőnket.	Bemutatás során	Alapvető	Feladatkiírás	CreateTable, ViewTable, BaseAction	“Egy élelem eggyel növeli a testhőt.”
R25	Legyen pisztoly, jelzőfény és patron, amik segítségével összeállítható a jelzőrakéta.	Bemutatás során	Alapvető	Feladatkiírás	CreateTable, ViewTable, BaseAction	“jelzőrakéta alkatrészeinek (pisztoly, jelzőfény, patron)”
R26	Legyenek titkos tárgyak, ami különböző hatást fejtenek ki a játékosokra.	Bemutatás során	Opcionális	Csapat	CreateTable, ViewTable, BaseAction	“...kötél, lapát, titkos eszköz vagy a rakétapisztoly alkatrésze…”
R27	Egy karakternél egyszerre maximum egy használható tárgy lehet.	Bemutatás során	Opcionális	Csapat	BaseAction	Cserélhet másikra, ekkor a kezében lévő leesik a mezőre. A jelzőpisztoly alkatrész mellé lehet

						másik tárgyat felvenni.
R28	Egy egységnyi munkának számít: - az egy (vagy lapáttal két) egységnyi hó ásása - szomszédos mezőre lépés - tárgy kiásása - kiásott tárgy felvétele - egy iglu építése (eszkimó) - a szomszédos mező vizsgálata (sarkkutató) - iglu lerombolása (lapáttal) - a jelzőpisztoly összeépítése és elsütése.	Bemutatás során	Fontos	Feladatkiírás és Csapat	ActionWithCharacters	“hó eltakarítása, egy szomszédos jégtáblára való lépés vagy egy kiásott tárgy felvétele.” “Egy-egy képesség alkalmazása is egy-egy munkát jelent.” “egy munka felhasználásával összeszerelhetik és elsüthetik,”
R29	Az igluban átvészelhető a hóvihar testhő vesztés nélkül.	Bemutatás során	Fontos	Feladatkiírás	BaseAction, SnowStorm	“iglut építeni, amiben átvészelhetők a hóviharok.”
R30	Az iglu 4 kör után elolvad, az alatta lévő befagyott tárgyak újra kiáshatóak lesznek.	Bemutatás során	Opcionális	Csapat	ViewTable, SpecificAction	“Egy iglu 4 kör után elolvad, hóvihar után megsemmisül.”
R31	Az iglut csak akkor használhatja valaki, ha az adott iglu üres.	Bemutatás során	Opcionális	Csapat	BaseAction	Egy igluban csak egy játékos lehet egyszerre.

R32	Az iglut lebonthatja két egység munkáért bárki, lapáttal egyért is.	Bemutatás során	Opcionális	Csapat	BaseAction	Az iglu lebontását követően kiáshatóak a mezőn lévő tárgyak.
R33	Az iglu hóvihar után megsemmisül.	Bemutatás során	Opcionális	Csapat	SnowStorm	Az iglut nem lehet újra használni hóvihar után, újat kell építeni
R34	A jelzőpisztoly elsütése a játék végét jelenti, a csapat nyer.	Bemutatás során	Alapvető	Feladatkiírás	BaseAction	"összeszerelhetik és elsüthetik, amivel megnyerik a játékot."
R35	Ha egy játékos meghal, megfullad a vízben vagy elfogy a testhője a játék véget ér, a társai nem játszhatnak tovább.	Bemutatás során	Alapvető	Feladatkiírás és Csapat	BaseAction	"Ha valaki menet közben meghal (vízbe esve megfullad vagy elfogy a testhője és kihűl), akkor a játék véget ér."

2.3.2 Erőforrásokkal kapcsolatos követelmények

Azonosító	**Leírás**	**Ellenőrzés**	**Prioritás**	**Forrás**	**Komment**
E01	A játékszoftver fejlesztéséhez és futtatásához számítógépre van szükség.	Bemutatás során	Alapvető	Csapat	-

E02	A számítógép legyen képes futtatni a programot.	Bemutatás során	Alapvető	Csapat	Elegendő tárhely, operációs rendszer, Java környezet stb...

2.3.3 Átadással kapcsolatos követelmények

Azonosító	**Leírás**	**Ellenőrzés**	**Prioritás**	**Forrás**	**Komment**
A01	Windows 10 operációs rendszerre lesz szükség	Bemutatás során	Fontos	Csapat	-
A02	Játék feltelepítése szükséges	Bemutatás során	Fontos	Csapat	Megtalálható legyen a gépen a program.

2.3.4 Egyéb nem funkcionális követelmények

Azonosító	Leírás	Ellenőrzés	Prioritás	Forrás	Komment
ENF01	A játékot több játékos játsza.	Bemutatás során	Opcionális	Csapat	Minden karaktert más irányíthat.
ENF02	A játék hordozható, nem egy adott gépen futtatható csak.	Bemutatás során	Opcionális	Csapat	Másik gépen futtatva is működni fog.
ENF03	A játék folyamatos, hibamentes működése garantált legyen.	Bemutatás során	Fontos	Csapat	Grafikai bugoknak, gombok hibás lekezelésének, program indokolatlan leállásának elkerülése.

2.4 Lényeges use-case-ek

2.4.1 Use-case leírások

Use-case neve	SnowStorm
Rövid leírás	Véletlenszerű időközönkénti hóvihar.
Aktorok	Controller
Forgatókönyv	**1.** A rendszer hóvihart hoz létre.
Alternatív forgatókönyv	**1.A.** A hóvihar egyes mezőket különböző mennyiségű hóval fedi be.
Alternatív forgatókönyv	**1.B.** A hóvihar által érintett mezőkön elhelyezkedő szereplő(k) testhője egy egységnyivel csökken.
Alternatív forgatókönyv	**1.C** A hóvihar által érintett igluk megsemmisülnek.

Use-case neve	CreateTable
Rövid leírás	A kontroller felelős a pálya felépítéséért.
Aktorok	Controller
Forgatókönyv	**1**. A rendszer létrehozza a különböző mezőkből álló pályát
Alternatív forgatókönyv	**2**. A rendszer elhelyez pár különböző tárgyat pár mezőn. Ezek a tárgyak a lapát, búvárruha, kötél, élelem, titkos mikkentyűk és a jelzőpisztoly részei.
Alternatív forgatókönyv	**3**. A rendszer az egyes mezőket különböző mennyiségű hóval látja el.
Alternatív forgatókönyv	**4**. A rendszer elhelyezi a játékosokat egy stabil mezőn és kinevezi egyiket kezdő játékosnak.

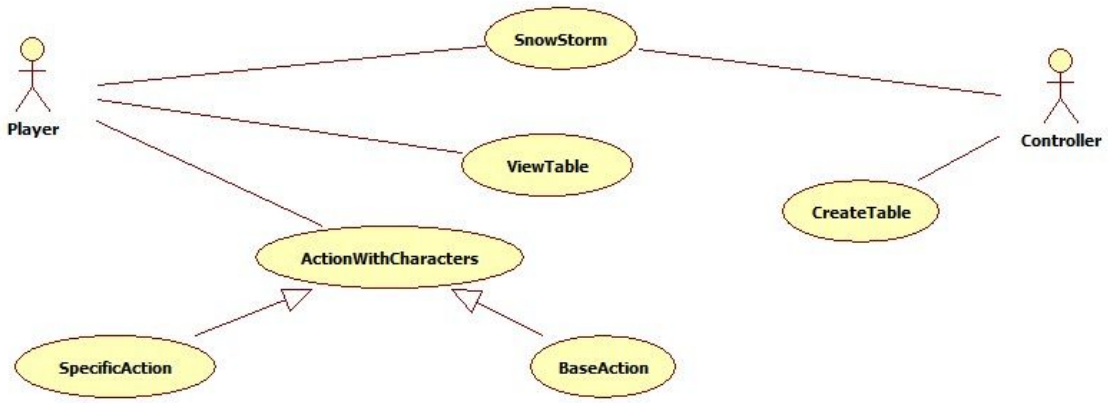
Use-case neve	ViewTable
Rövid leírás	A játékosok látják az egész pályát és azon lévő karaktereket, iglukat.
Aktorok	Player
Forgatókönyv	**1**. A játékos látja az egész játékteret.
Alternatív forgatókönyv	**2**. A játékosok látják egymást.
Alternatív forgatókönyv	**3**. A játékos látja hol áll iglu.
Alternatív forgatókönyv	**4.** A játékos látja a már felderített mezők teherbírását, amíg nem fedi be újra a hóvihar hóval.
Alternatív forgatókönyv	**5.** A játékos látja a hómezővel nem borított tárgyakat (kiásható és felvehetőeket)

Use-case neve	ActionWithCharacters
Rövid leírás	A játékosok különböző tevékenységeket végezhetnek.
Aktorok	Player
Forgatókönyv	**1.** BaseAction-t hajt végre.
Alternatív forgatókönyv	**2.** SpecificAction-t hajt végre.

Use-case neve	BaseAction
Rövid leírás	A játékosok végezhetnek alap tevékenységeket, karaktertől függetlenül, egy vagy két munkaegységért cserébe.
Aktorok	Player
Forgatókönyv	**1**. A játékos szomszédos mezőre lép.
Alternatív forgatókönyv	**1.A** A játékos stabil mezőre lép.
Alternatív forgatókönyv	**1.B.1** A játékos instabil mezőre lép, de az még elbírja, így nem fordul át a mező.
Alternatív forgatókönyv	**1.B.2** A játékos instabil mezőre lép, de az már nem bír el több embert, így átfordul és mindenki meghal, aki a mezőn tartózkodott.
Alternatív forgatókönyv	**1.C** A játékos lyukas mezőre lép, így beleesik a hideg vízbe.
Alternatív forgatókönyv	**1.C.1** Van rajta búvárruha, így nem fullad meg, bármikor kimentheti valamelyik csapattársa.
Alternatív forgatókönyv	**1.C.2** Nincs rajta búvárruha, így megfulladhat, ha nem menti ki, valamelyik csapattársa.
Alternatív forgatókönyv	**2.** A játékos havat takarít el. (ásás)
Alternatív forgatókönyv	**3.** A játékosok a jelzőpisztolyt összeépítik és elsütik.
Alternatív forgatókönyv	**4.** A játékos kiás egy tárgyat.
Alternatív forgatókönyv	**5.** A játékos felvesz egy tárgyat.
Alternatív forgatókönyv	**6.** A játékos lebont egy iglut.
Alternatív forgatókönyv	**7.** A játékos eszik, a testhője eggyel nő.
Alternatív forgatókönyv	**8.** A kötéllel kimenti a társát.

Use-case neve	SpecificAction
Rövid leírás	A játékosok végezhetnek a karakterükhöz kötődő különleges tevékenységet.
Aktorok	Player
Forgatókönyv	**1.** Az eszkimó épít egy iglut.
Alternatív forgatókönyv	**2.** A sarkkutató kiszámítja, mekkora a szomszédos mező teherbírása.

2.4.2 Use-case diagram



2.5 Szótár

Fogalom:	**Jelentés:**
Akadály	Az eszkimó, ha iglut épít és benne tartózkodik valaki az a többi játékos számára akadályt képez.
Mező/Jégmező	Egy jégmező az, amelyen a karakterek/ iglu/eszközök lehetnek. A jégmezőknek eltérő teherbírásuk lehet.
Áthalad	Amikor a karakter a saját mezejéről egy szomszédos mezőre lép.
Befed	A hóvihar után a tiszta jégmezőt friss hó lep el.
Búvárruha	A búvárruha viselése megvédi a játékost a lyukba esés utáni haláltól.
Cél	A rakétapisztoly összeépítése és elsütése utáni állapot.
Csapat	3 vagy több eszkimókból és/vagy sarkkutatókból álló játékosok halmaza.
Cselekvő	A soron következő játékos.
Csapda	A titkos felvehető eszközök közül a játékosok szempontjából a rosszabbik, egyfajta büntetés a játék nehezítése érdekében.
Elkap	Ha a hóvihar egy játékost érint, akkor a testhője egy egységgel csökken.
Eltakarít	A játékos a havas jégtábláról letakarítja a havat.
Eszkimó	5 testhővel kezdő, iglu építésére képes, választható karakter.
Élelem	Egy egységgel növeli a testhőt a használata.
Épít	Az eszkimó iglut tud építeni a jégmezőn.
Felvétel	Az eszközöket a játékos felveszi és használni tudja.
Havas jégtábla	Ilyen mezőkön nem látszódik a jégbe fagyott tárgy (ha van ilyen). A játékosoknak a mezőt le kell tisztítaniuk.
Használ	Az eszközök különböző cselekvésekre vagy azok gyorsítására alkalmazhatóak..
Helyzet	Az a mező terület, ahol az adott időpillanatban az adott karakter elhelyezkedik.

Hóvihar	Véletlenszerű esemény. A pálya bizonyos részén a tiszta jégtáblákat befedheti hóval és a játékos testhőjét eggyel csökkenti.
Iglu	Az eszkimók által felépíthető hely. Megvéd egy játékost az esetleges hóvihartól. Egyalkalmas, vagyis hóvihar után megsemmisül, vagy négy kör után elolvad.
Instabil mező	A mezők közül az a fajta, aminek van teherbírása. Bizonyos emberszám után átfordul (arra a pillanatra kb. lyukas mezőként viselkedik), a rajta lévő játékosok a vízbe esnek. A mezők teherbírása alapjáraton nem látszódik, de sarkkutatók felfedhetik a csapat számára.
Indít	A játék kezdetét jelenti.
Irányít	A játékos egy periféria hatására, különböző lépéseket hajt végre.
Játék vége	A játékosok nem képesek további műveletekre.
Játékos	Felhasználó, aki egy karaktert irányít.
Jégtábla	Jégmezők összessége, amin az egész játék történik.
Jelzőpisztoly	A játék megnyerésének feltétele a jelzőpisztoly megépítése. Darabjai: pisztoly, jelzőfény, patron.
Karakter	Játékos által irányított szereplő.
Kiásás	A tiszta jégtáblán látott tárgyak felszabadítása, hogy fel lehessen venni.
Kihúz	A kötél segítségével megmenthetjük a szomszédos mezőn elsüllyedt játékos társunkat.
Kilép	A játékot a cél előtt hagyják abba a felhasználók.
Kötél	A kötél segítségével a szomszédos mezőn elsüllyedt csapat társunkat húzhatjuk ki.
Lapát	Ennek segítségével a hó ellapátolás és az iglu lebontása felgyorsíthatóak.
Luk	Speciális mező, ha a játékos erre a mezőre lép, akkor vízbe esik.
Megfullad	A játékos/ok, ha lyukra lépnek vagy a jégmező nem bírja el az adott mennyiségű embert, akkor a játékos(ok) vízbe esik/esnek és meghalhat(nak).
Meghal	A játékos vízbe esve megfullad vagy elfogy a testhője.

Meglepetés	A titkos felvehető eszközök közül a játékosok szempontjából a jobbik, egyfajta segítség a játék könnyítése érdekében.
Mozog	A karakter a játékosok inpoutjára történő helyzetváltoztatása.
Munkaráfordítás	Amíg ez az érték nem csökken nullára, addig a játékos munkafolyamatokat végezhet. Minden kör elején ennek értéke: 4.
Összeszerel és elsüt	A játék megnyeréséhez szükséges cselekvések a jelzőpisztollyal.
Sajátos cselekedet	Nem általános módja a munkaegység felhasználásának. Mindkét fajta karakternek más. Eszkimónak az igluépítés, sarkkutatónak a mezőfelderítés.
Sarkkutató	4 testhővel kezdő, szomszédos mezők teherbírásának felfedezésére képes, választható karakter.
Stabil mező	A mezők közül az a fajta, ami soha nem fordul át, akárhányan állhatnak rajta.
Tenger	A pályát(jégmezőt) körülvevő közeg.
Testhő	A játékos életeinek száma: az eszkimónak az eredeti élete 5, a kutatóknak 4.
Tiszta jégtábla	A mezőt nem borítja hó, a rajta lévő eszközöket a játékos ki tudja ásni.
Védve van	A védett játékosok testhője nem csökken ha hóvihar sújtotta területen vannak.
Vizsgál	A sarkkutató ellenőrizni tudja az adott jégmező teherbírását.

2.6 Projekt terv

Határidő	**Feladat**
február 24.	Követelmények, projekt, funkcionalitás - beadás
március 2.	Analízis modell kidolgozása 1. - beadás
március 9.	Analízis modell kidolgozása 2. - beadás
március 16.	Szkeleton tervezése - beadás
március 23.	Szkeleton beadása
március 30.	Prototípus koncepciója - beadás
április 6.	Részletes tervek - beadás
április 20.	Prototípus készítése és tesztelése
április 27.	Prototípus beadása
május 4.	Grafikus felület specifikációja - beadás
május 11.	Grafikus változat készítése
május 18.	Grafikus változat és az összefoglalás elkészítése - beadás

A dokumentumok együttes elkészítéséhez a Google Dokumentumokat, a diagramok elkészítéséhez a draw.io-t használjuk, így az egymás új fejlesztéseit egyből láthatjuk és reagálni tudunk azokra.
Az általános, egyszerűbben eldönthető kérdéseket Messengeren vitatjuk meg, míg a több időt igénylőket személyes összejövetel, vagy Discord konferenciabeszélgetések segítségével tárgyaljuk meg.
A későbbiekben elkészülő kódokat Github segítségével osztjuk meg egymás között.

2.7 Napló

Kezdet	Időtartam	Résztvevők	Leírás
2020.02.13 - 11:00	0.5 óra	Ziaja	Ziaja összefoglalja a fontosabb követelményeket és információkat a feladatszöveg alapján
2020.02.15 - 18:00	3 óra	Sipula Varga	Feladat értelmezése Sipula megírja: a Cél, és Követelmények pontokat Varga megírja: Funkciók és Szakterület pontokat
2020.02.16 - 12:00	5 óra	Ziaja Sipula Varga Csoma Szepesi-Nagy	Mindenki: Eddigi kitöltött template átnézése, javításai; csapat döntéseit átbeszélni; Definíciók összeszedése Varga és Sipula: Funkciók Ziaja: Kezdetleges use-case diagram Szepesi: use-case diagram Csoma: Funkciók és Követelmények javításai
2020.02.19 - 12:15	3 óra	Sipula Varga Szepesi-Nagy Csoma	Sipula, Varga, Szepesi-Nagy: Részt vett a konzultáción Mindenki: a konzin szerzett információk segítségével use-case javítások történtek
2020.02.20 - 9:00	2.5 óra	Sipula Ziaja	Sipula: Ellenőrzi az egész doksit

			Ziaja: Use-case leírások javításai, funkciók javításai, általános áttekintés szerkesztése és magyarázata
2020.02.21 - 14:00	1 óra	Csoma Sipula Szepesi-Nagy Varga Ziaja	Mindenki: Csoportos megbeszélés, ahol néhány főbb módosítást vezetett be a csapat.
2020.02.21 - 22:00	2 óra	Sipula Varga	Mindenki: Módosítások utáni anomáliák keresése, Funkciók és Követelmények pontosítása ezek fényében.
2020.02.22 - 09:00	2 óra	Ziaja	Ziaja: egész doksi átolvasása, javítások és ki nem dolgozott részek keresése.
2020.02.22 - 21:00	4 óra	Sipula Varga	Sipula: Szótárhoz szavak keresése, 2.1.4 kiegészítése, 2.2.2 átnézése + javítása, Dokumentum utolsó formázása Varga: 2.1.3 kijavítása, kiegészítése, szótár bővítése Mindenki: 2.5 Kiegészítése, meglévő hibás elemek javítása, Dokumentum nyomtatása

3. Analízis modell kidolgozása

3.1 Objektum katalógus

3.1.1 Szereplő

Különböző képességű szereplőknek kell haladni a jégtáblákon, használni a tárgyakat és elsütni a jelzőrakétát, ezzel megnyerve a játékot.

3.1.1.1 Eszkimó

Egyfajta szereplő, akinek felelőssége alá tartoznak a tevékenységek végrehajtása, eszközök kezelése, a testhő és munkaráfordítás nyilvántartása és igluk építése.

3.1.1.2 Sarkkutató

Egyfajta szepelő, akinek felelőssége alá tartoznak a tevékenységek végrehajtása, eszközök kezelése, a testhő és munkaráfordítás nyilvántartása és a jégmezők teherbírásának vizsgálata.

3.1.2 Jégtábla

A jégtábla felelőssége alá tartozik, a rajtuk keletkezett hó mennyiségének nyilvántartása, az épült igluk megsemmisítése és hátralévő idejük számolása, illetve a rajtuk álló karakterek tárolása. Egy jégtábla lehet stabil vagy instabil, ha túl sokan állnak rajta (lyukason 1 is), akkor át kell fordulnia.

3.1.3 Jelzőrakéta

Az összeszerelt jelzőrakétával lehet megnyerni a játékot, az alkatrészek összeszereléséért felelős.

3.1.4 Pisztoly

A jelzőrakéta 3/1 (pisztoly) alkatrésze, ami szükséges a jelzőpisztoly elsütéséhez.

3.1.5 Patron

A jelzőrakéta 3/2 (patron) alkatrésze, ami szükséges a jelzőpisztoly elsütéséhez.

3.1.6 Jelzőfény

A jelzőrakéta 3/3 (jelzőfény) alkatrésze, ami szükséges a jelzőpisztoly elsütéséhez.

3.1.7 Élelem

Létezik élelem, aminek elfogyasztásával a szereplők az élethőjüket tudják növelni.

3.1.8 Kötél

A kötél segítségével tud egy karakter egy másik karaktert megmenteni a vízbefulladástól. Ennek felügyelete a Rope feladata.

3.1.9 Lapát

A lapáttal egy karakter két egység havat tud eltakarítani egy munkaráfordítással. Ennek felügyelete a lapát feladata.

3.1.10 Búvárruha

Buvárruhát viselő karakter vízbeesés esetén nem fullad meg. Ennek felügyelete a Swimsuit feladata.

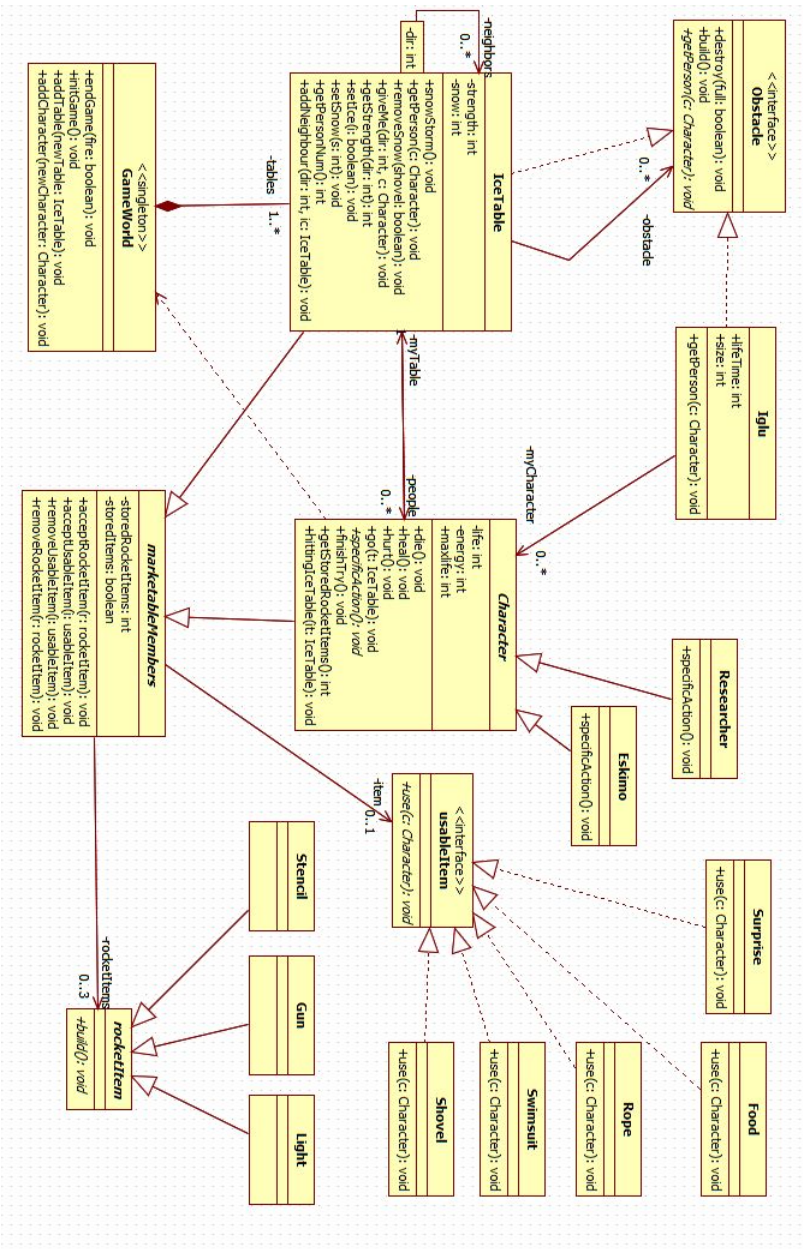
3.1.11 Iglu

Iglu állhat egy jégtáblán, a benne lévő szereplőket megvédi hóvihar esetén.

3.1.12 Meglepetés

Ez az entitás felelős a játék menetét befolyásoló jó vagy rossz kimenetelekért. Ennek használata felelős a játék körinek hóviharon kívüli befolyásolásáért (plusz cselekedet, körből kimaradás, testhő feltöltés stb.).

3.2 Statikus struktúra diagramok



3.3 Osztályok leírása

3.3.1 Character

- **Felelősség**

  Szomszédos mezőre lépés kezdeményezése a jégtábla felé, saját testhő és energia nyilvántartása. Végül a fegyver megépítésére való próbálkozás is a feladata. Megtisztít jégtáblát hótól és jégtől egyaránt.

- **Ősosztályok**

  marketableMembers

- **Attribútumok**

  life: int: A testhő tárolására szolgál
  energy: int: A munkaegységek tárolására szolgál
  maxlife: int A karakter életének maximális száma.
  myTable: IceTable: A saját jégtábla tárolására szolgál

- **Metódusok**

  void die(): A szereplő vízbe esett, meg fog fulladni, ha nincs nála búvárruha. Ha megfullad, akkor játéknak vége, vesztett a csapat.
  void hurt(): A játékos testhője eggyel csökken.
  void heal(): A játékos kap egy testhőt.
  void go(IceTable it): A játékos véglegesen átlép az adott jégtáblára, elmenti myTable-ként.
  void specificAction(): Absztrakt függvény, tőle öröklődés során kötelező megírni.
  void finishTry(): A jégmezőjét használva, kideríti, hogy véget vethet-e a játéknak és győz-e a csapat. Megnézi mindenki egyazon mezőn van és megvan-e minden jelzőrakéta alkatrész.
  int getStoredRocketItems(): A tárolt jelzőrakéta alkatrészek száma
  void hittingIceTable(IceTable it): Az it jégtáblának a tisztítása, akár jégtörés, akár hólapátolás.

3.3.2 Eskimo

- **Felelősség**

  Eszkimónak kell tudnia iglut építeni és ezen kívül megörökli a Character összes felelősségét.

- **Ősosztályok**

  Character marketableMembers

- **Metódusok**

  void specificAction(): Iglu építése, a saját táblán keresztül.

3.3.3 Food

- **Felelősség**
  A használójának egy testhőt kell biztosítani.
- **Interfészek**
  usableItem
- **Metódusok**
  void use(Character c): Az adott karakternek eggyel kell növelni a testhőjét.

3.3.4 GameWorld

- **Felelősség**
  A jégtáblák tárolása, hóviharok keltése, egy kör leteltének következményeinek kezelése, és a játéknak véget vetni.
- **Attribútumok**
  tables: IceTable: A táblák tárolása, hogy azokon hóvihar keletkezhessen, hogy bomoljanak a táblák iglujai.
- **Metódusok**

  void addCharacter(newCharacter: Character): Új karakter létrehozásáért/játékba helyezéséért felel.
  void addTable(newTable: IceTable): Új mező hozzáadásáért felelős.
  void endGame(boolean fire): Ha a fire true, akkor nyertek a játékosok, vége a játéknak. Ha false, akkor vesztettek, de ugyanúgy vége a játéknak.
  void initGame(): A játék létrehozásáért felelős függvény.

3.3.5 Gun

- **Felelősség**
  A pisztoly entitása.
- **Ősosztályok**
  rocketItem

3.3.6 IceTable

- **Felelősség**
  Egy jégtábla kezelése, rajta lévő igluk, karakterek kezelése.

- **Ősosztályok**
  marketableMembers

- **Interfészek**
  Obstacle

- **Attribútumok**
  strength: int: A jégtábla teherbírásának tárolására használandó
  snow: int: A jégtáblán lévő hó tárolása
  people: Character: A jégtáblán álló szereplők nyilvántartása.
  obstacle: Obstacle: A jégtáblán álló igluk nyilvántartása.
  neighbors: IceTable: Szomszédos táblák tárolása

- **Metódusok**
  void addNeighbour(dir: int, ic: IceTable): Egy mező szomszédjának a regisztrálásáért felelős függvény.
  void snowStrom(): Hóvihar van, azaz rajta álló emberek testhőjének csökkentése, és életlenszerű mennyiségű hó hozzáadása.
  void getPerson(Character c): c karakter elfogadása, azaz elmenteni people közé és szólni neki, hogy már rajtunk áll sikeresen. Ha nem sikeres, akkor arról tájékoztatni, hogy vízbe esett.
  void removeSnow(shovel: boolean): Hó (false -1; true -2) mennyiségű hő eltűntetése.
  void giveMe(int dir, Character c): int irányban lévő szomszédnak átadni c-karaktert (getPerson()-függvénnyel)
  int getStrength(int dir): Kideríti az adott szomszéd hó vastagságát.
  void setIce(i: boolean): Tör vagy létrehoz jeget.
  void setSnow(s: int): Beállítjuk a hó mennyiségét.
  int getPersonNum(): Visszaadja, hogy hányan állnak jelenleg a jégtáblán.

3.3.7 Iglu

- **Felelősség**
  Hátralévő idő és karakterek számontartása, benne álló karakter(ek) megvédése hóvihar esetén.

- **Interfészek**
  Obstacle

- **Attribútumok**
  lifeTime: int: Hátralévő idő nyilvántartása
  size: int: Méretének nyilvántartása
  myCharacter: Character: Bent álló(k) nyilvántartása

- **Metódusok**
  void getPerson(Character c): Adott játékos eltárolása.

3.3.8 Light

- **Felelősség**
  A jelzőfény nyilvántartása.

- **Ősosztályok**
  rocketItem

3.3.9 marketableMembers

- **Felelősség**

Rakétaalkatrészek és tárgyak átruházásához szükséges ős.

- **Attribútumok**
  storedRocketItems: int: Alkatrészek mennyiségének tárolása.
  storedItem: boolean: Van-e eltárolva valami
  rocketItems: rocketItem: Alkatrészek tárolása
  item: usableItem: Tárgyak tárolása.

- **Metódusok**
  void acceptRocketItem(rocketItem r): Átvesszük az alkatrészt
  void removeRocketItem(rocketItem r): Átadjuk az alkatrészt
  void acceptUsableItem(usableItem i): Átvesszük a tárgyat
  void removeUsableItem(usableItem i): Átadjuk a tárgyat

3.3.10 Obstacle

- **Felelősség**
  Akadály vagy akadályt tartó interface

- **Metódusok**
  void destroy(full: boolean): Iglu esetében bomlik (egyet/összeset), tábla esetén továbbítja az iglunak a kérést.
  void build(): Iglu esetében felépül/fejlődik, tábla esetén továbbítja az iglunak.
  void getPerson(c: Character): Iglu esetében eltesszük a karaktert, tábla esetében a karakternek is szólunk, hogy elfogadtuk az áthelyezési kérelmét.

3.3.11 Researcher

- **Felelősség**
  Sarkkutatók derítik fel a mezők teherbírását és ezen kívül megörökli a Character összes felelősségét.

- **Ősosztályok**
  Character marketableMembers

- **Metódusok**
  void specificAction(): Egy szomszédos mező felderítése, a saját táblán keresztül.

3.3.12 rocketItem

- **Felelősség**
  Jelzőrakéta alkatrészek őse, összeépíthetőek kell legyenek.

- **Metódusok**
  void build(): Absztrakt függvény, öröklés esetén megvalósítandó.

3.3.13 Rope

- **Felelősség**
  Egy karakter kimentésének levezénylése.

- **Interfészek**
  usableItem

- **Metódusok**
  void use(Character c): Kimenti a karaktert a vízből.

3.3.14 Shovel

- **Felelősség**
  Használatakor elvárt, hogy két mennyiségű hó eltűnjön a tábláról.
- **Interfészek**
  usableItem
- **Metódusok**
  void use(Character c): A kapott karakternek mezőjéről eltűntet két mennyiségű havat.

3.3.15 Stencil

- **Felelősség**
  Patron a jelzőrakéta egy alkatrésze, összeszerelhető.
- **Ősosztályok**
  rocketItem

3.3.16 Surprise

- **Felelősség**
  A játék befolyásolása meglepetésszerűen
- **Interfészek**
  usableItem
- **Metódusok**
  void use(Character c): Meglepetés végrehajtása.

3.3.17 Swimsuit

- **Felelősség**
  Játékos megmentése, ha vízbeesik.
- **Interfészek**
  usableItem

- **Metódusok**

  void use(Character c): A karakter nem hal meg, de a vízben marad.

3.3.18 usableItem

- **Felelősség**

  Használható tárgy interface-e, minta a tárgyaknak.

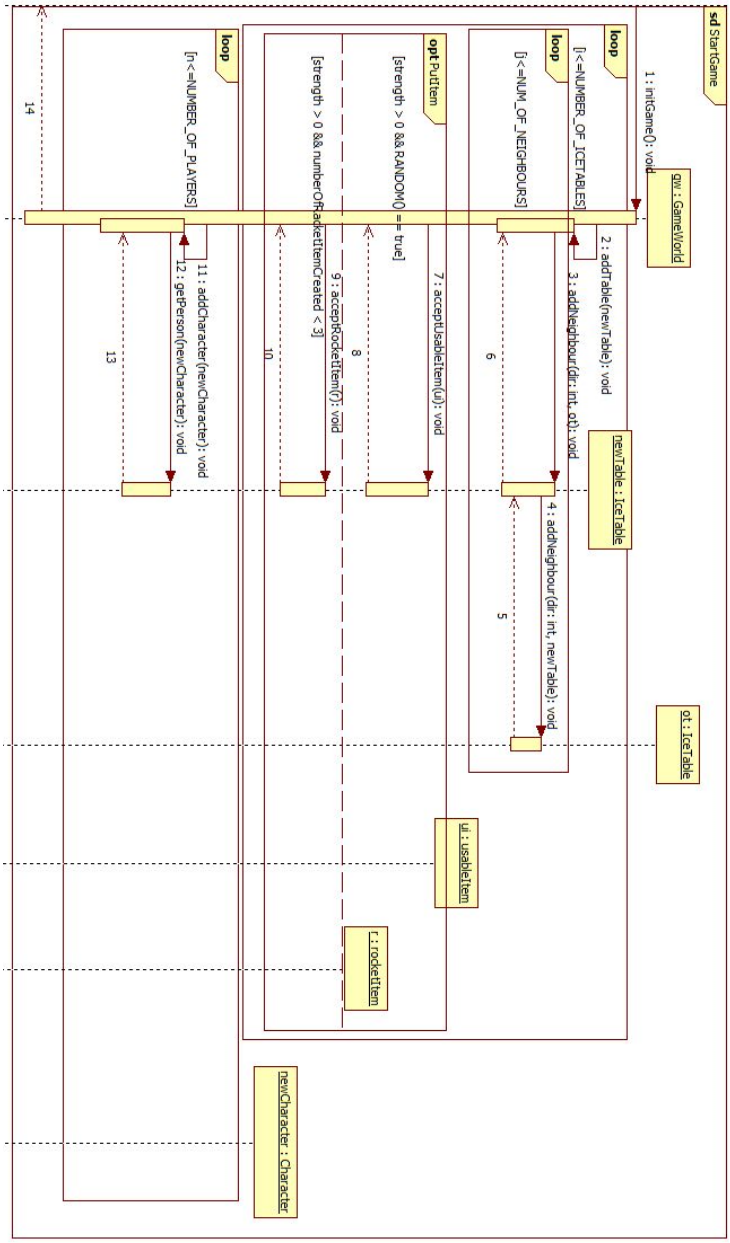
- **Metódusok**

  void use(Character c): Absztrakt metódus, minta, hogy hogyan kell kinéznie egy tárgynak

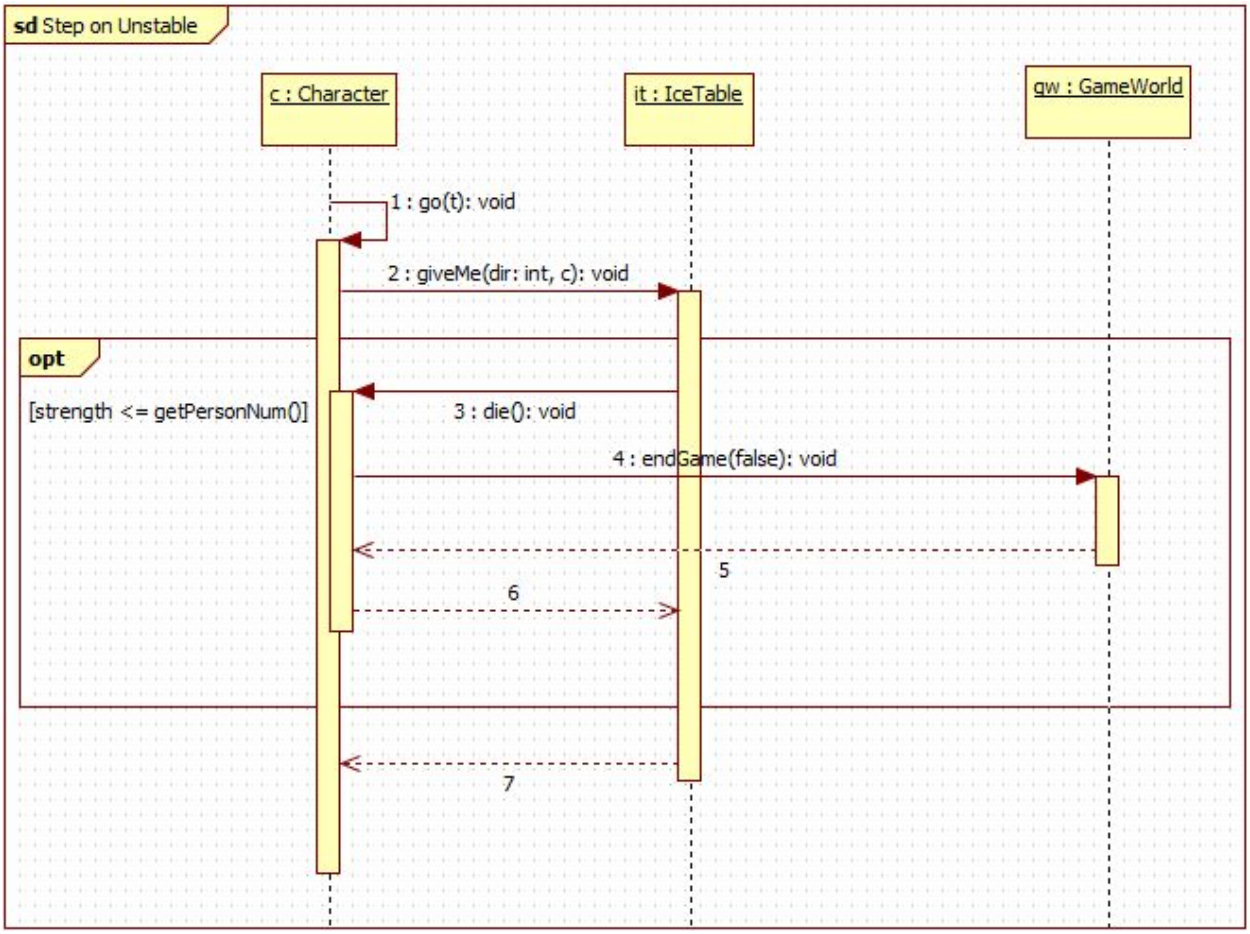
  .

3.4 Szekvencia diagramok

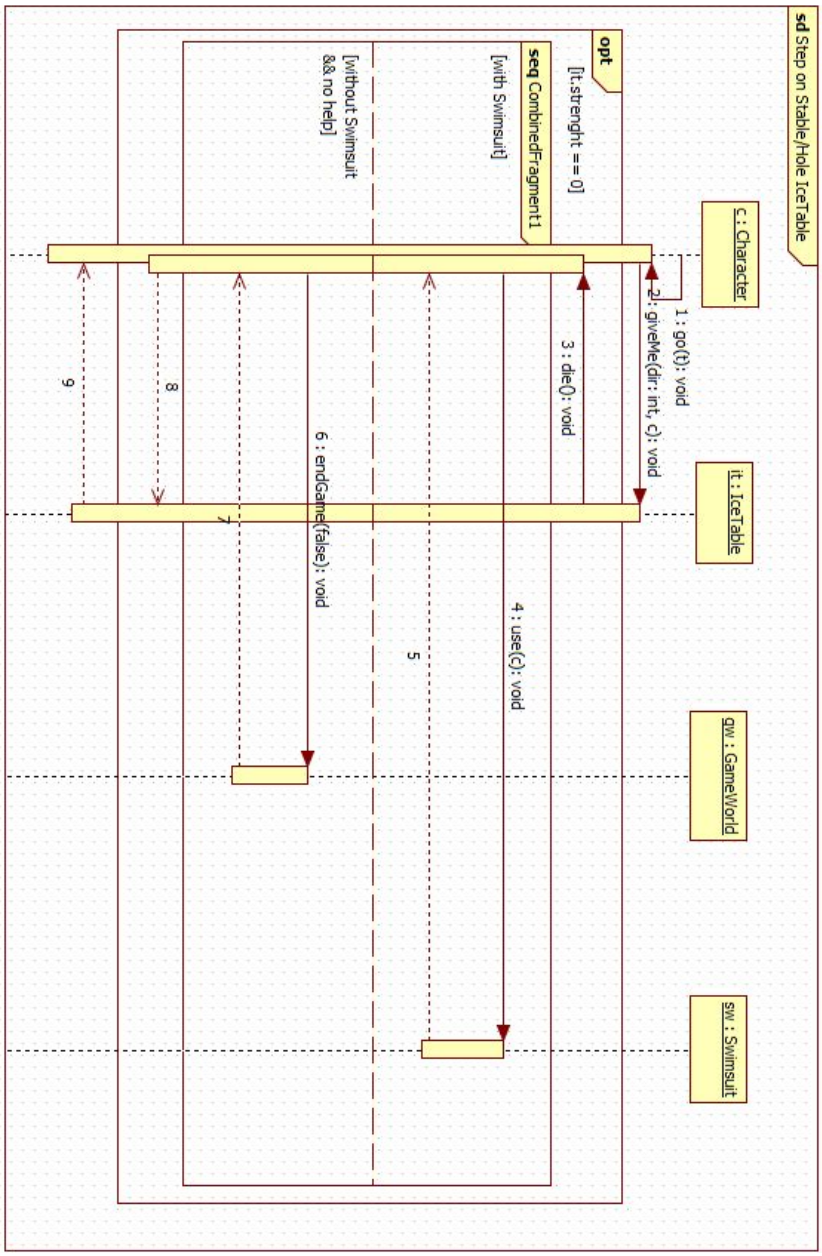
3.4.1 Game initialization



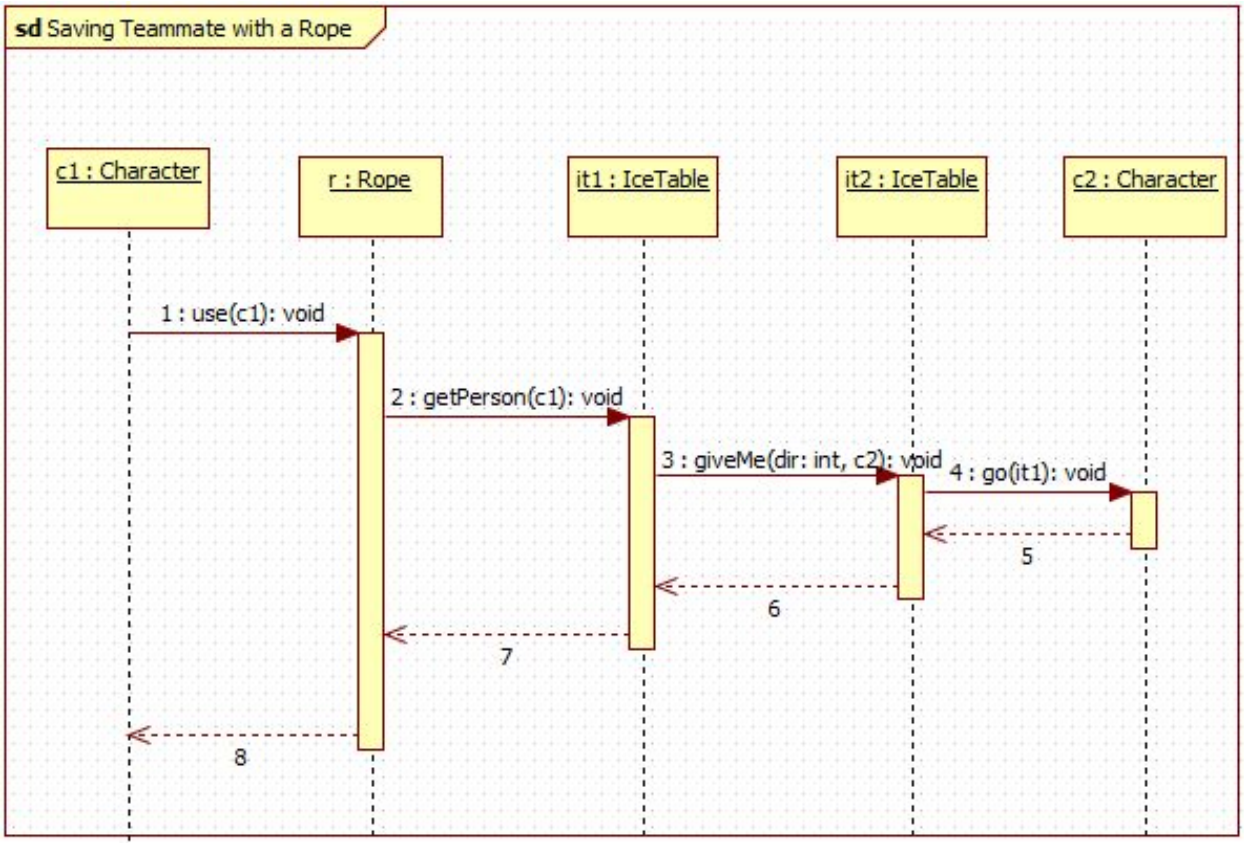
3.4.2 Stepping onto unstable



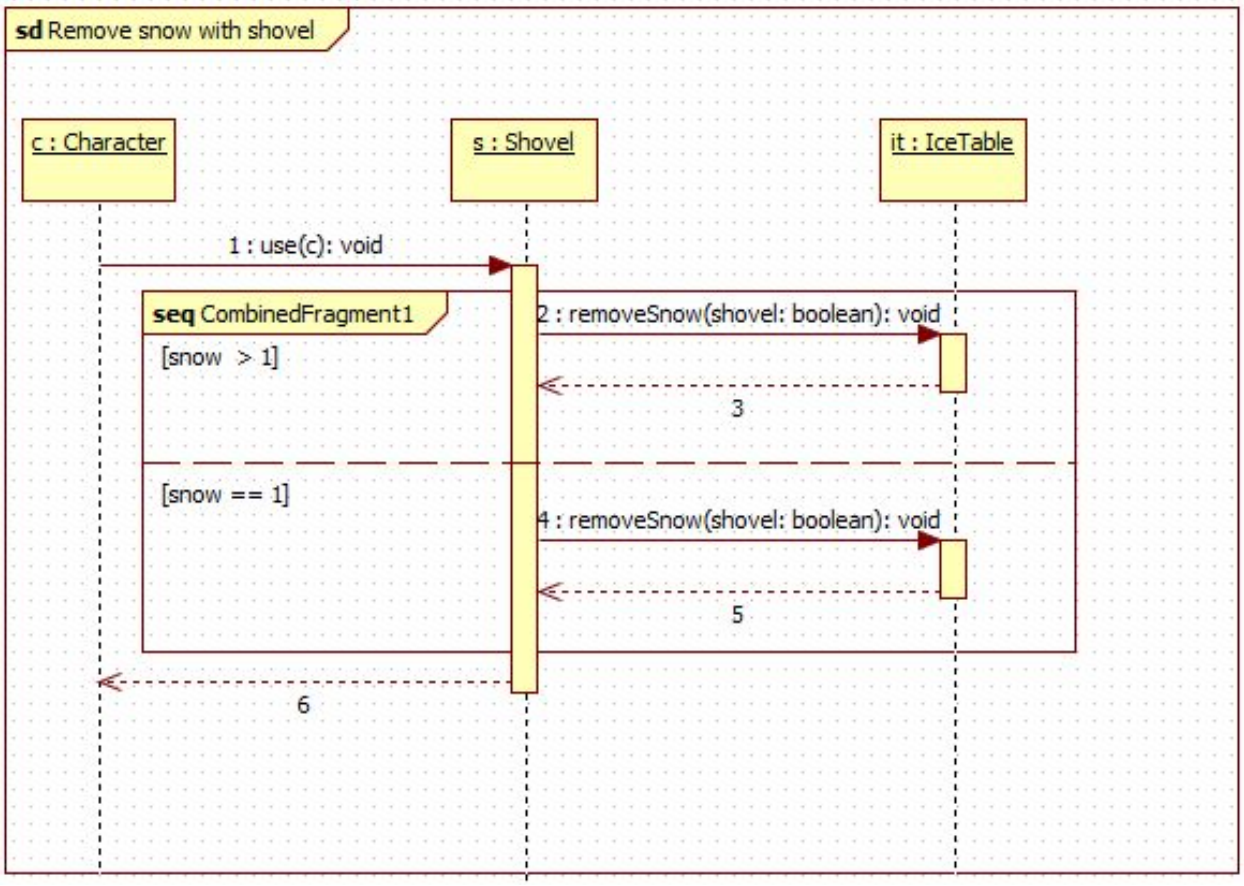
3.4.3 Stepping onto Stable or Hole IceTable.



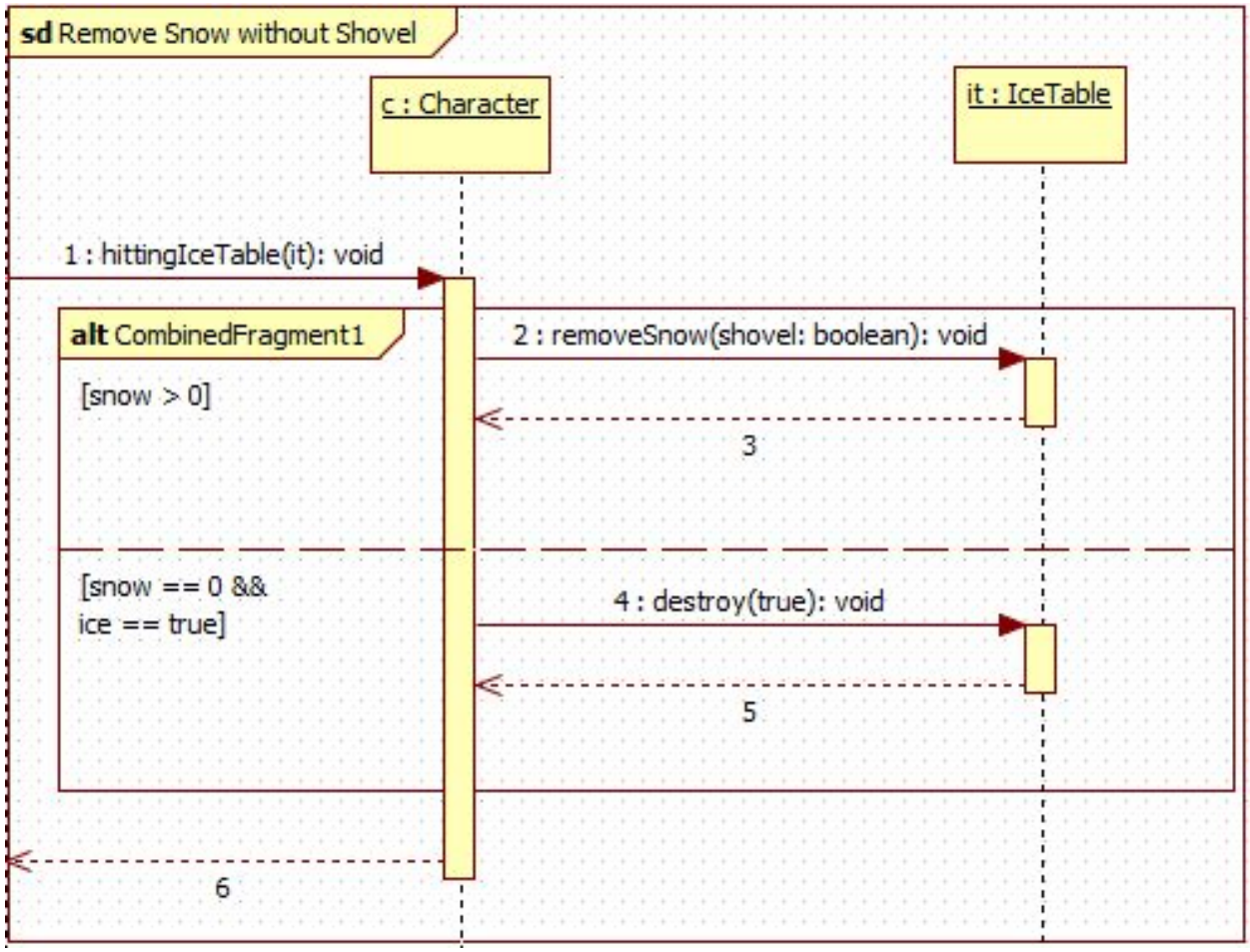
3.4.4 Saving Teammate with a Rope



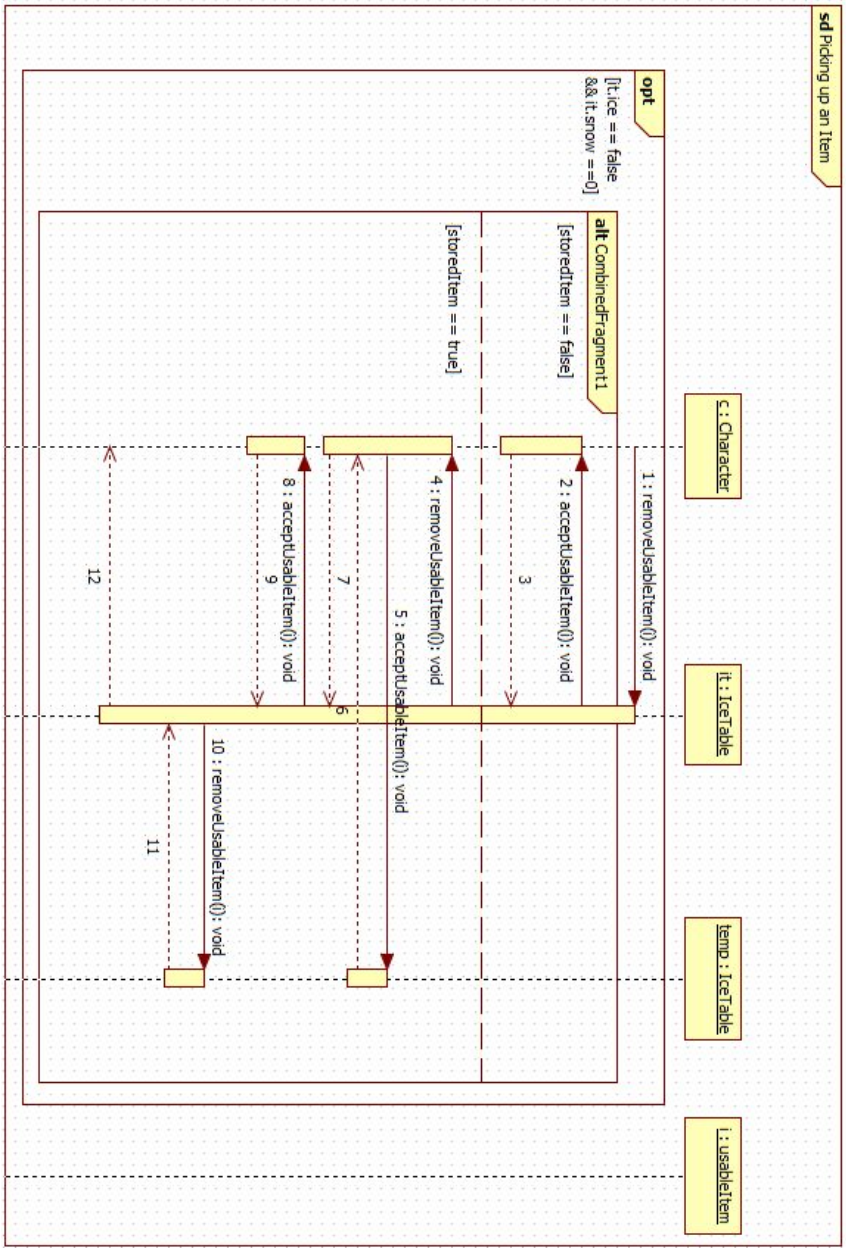
3.4.5 Removing snow with shovel



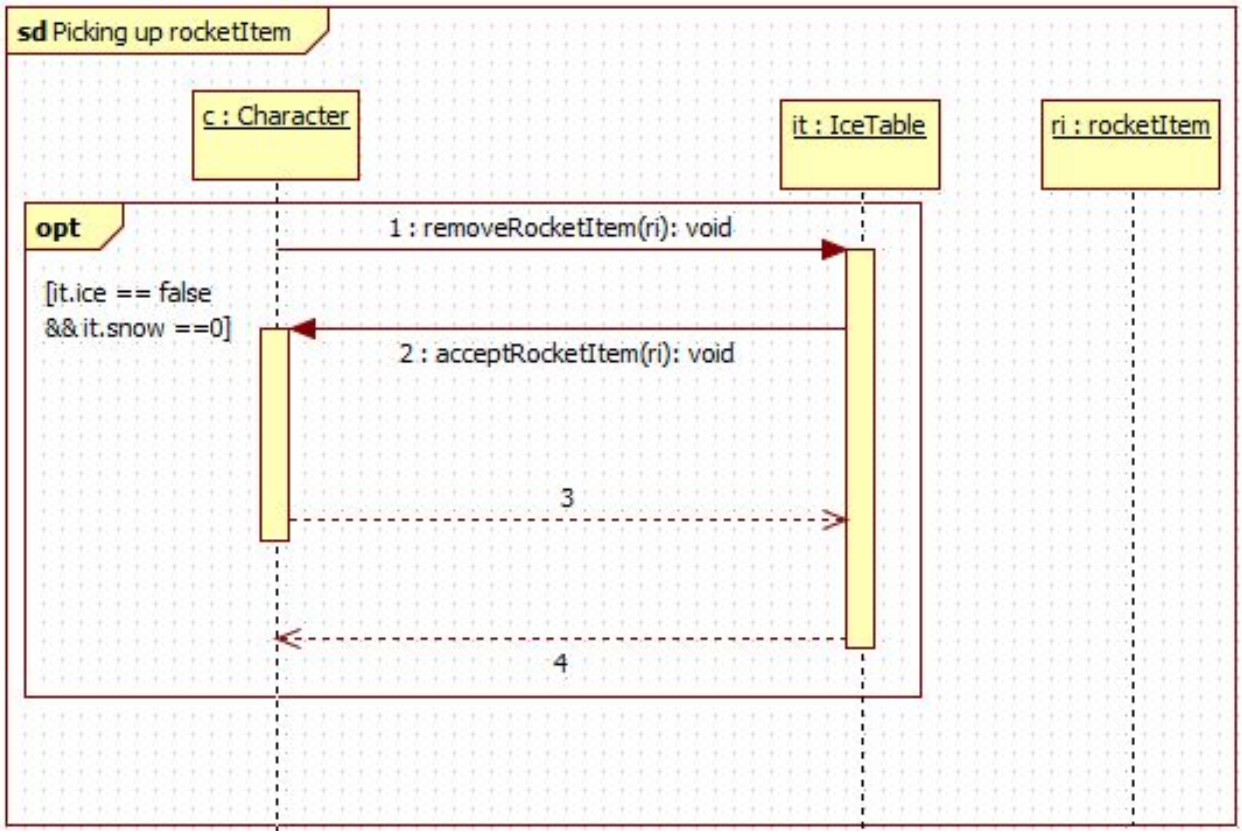
3.4.6 Removing snow without shovel



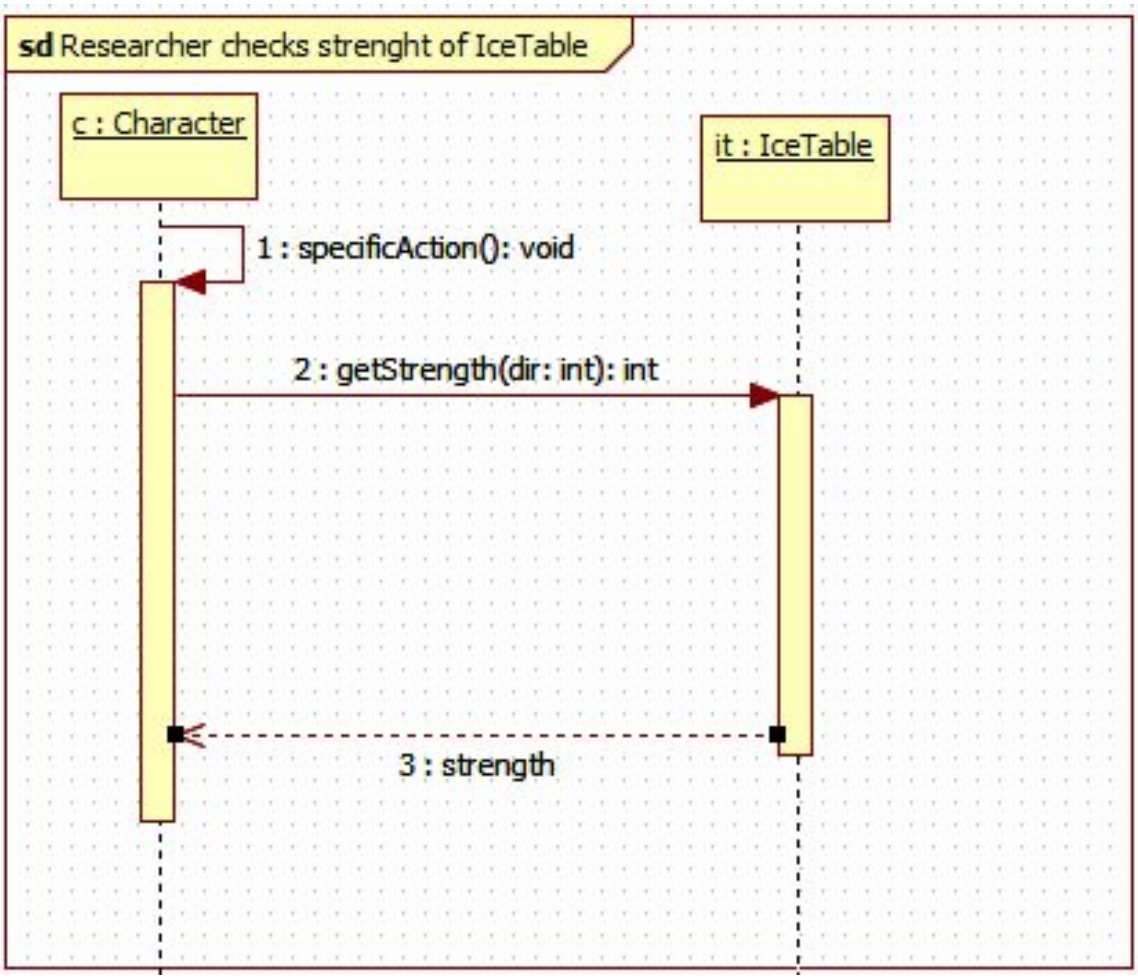
3.4.7 Picking up Item



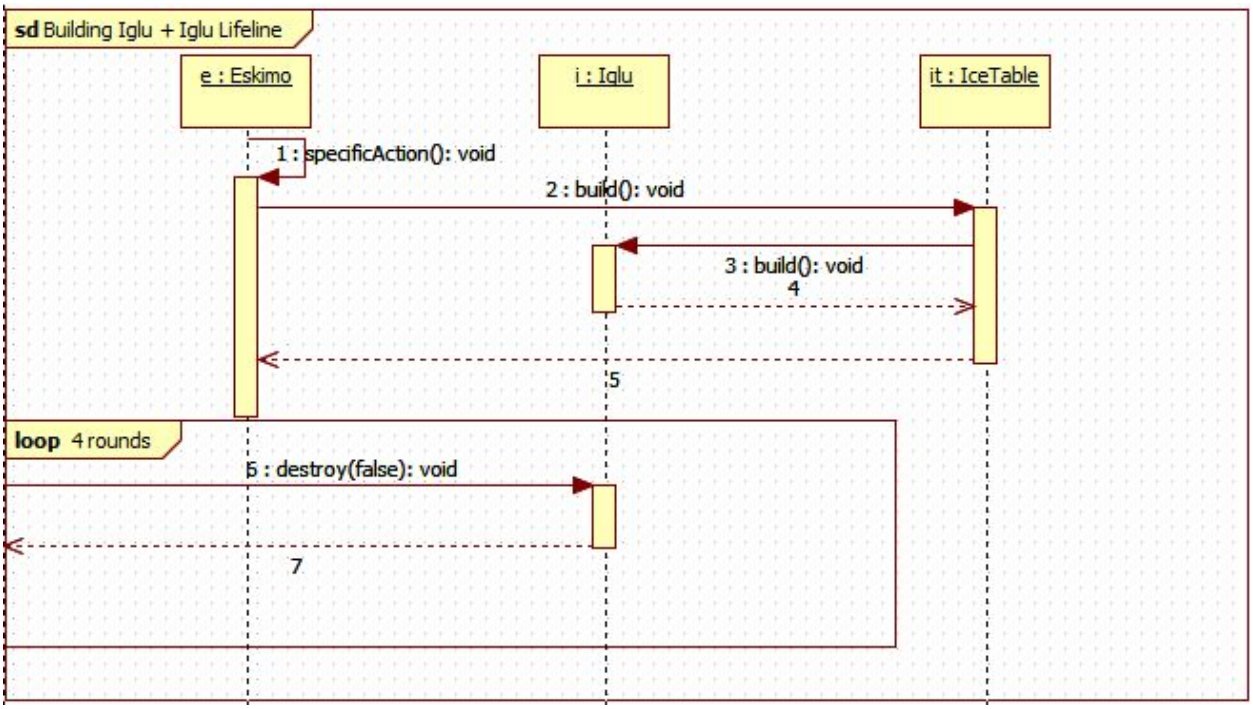
3.4.8 Picking up RocketItem



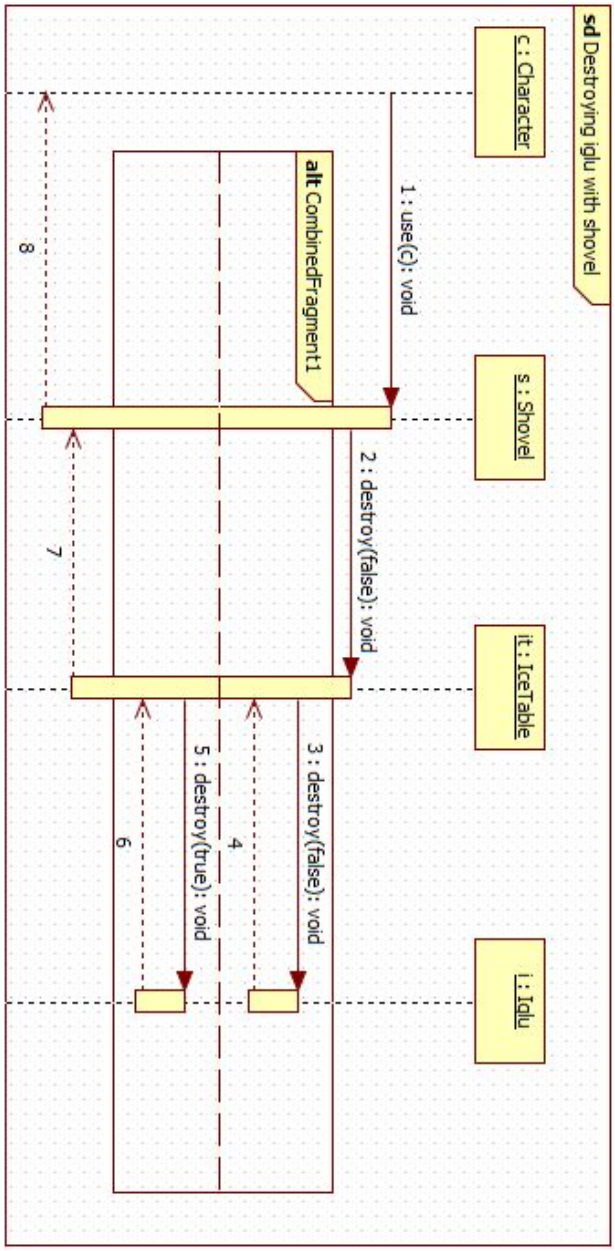
3.4.9 Researcher checks strength of IceTable



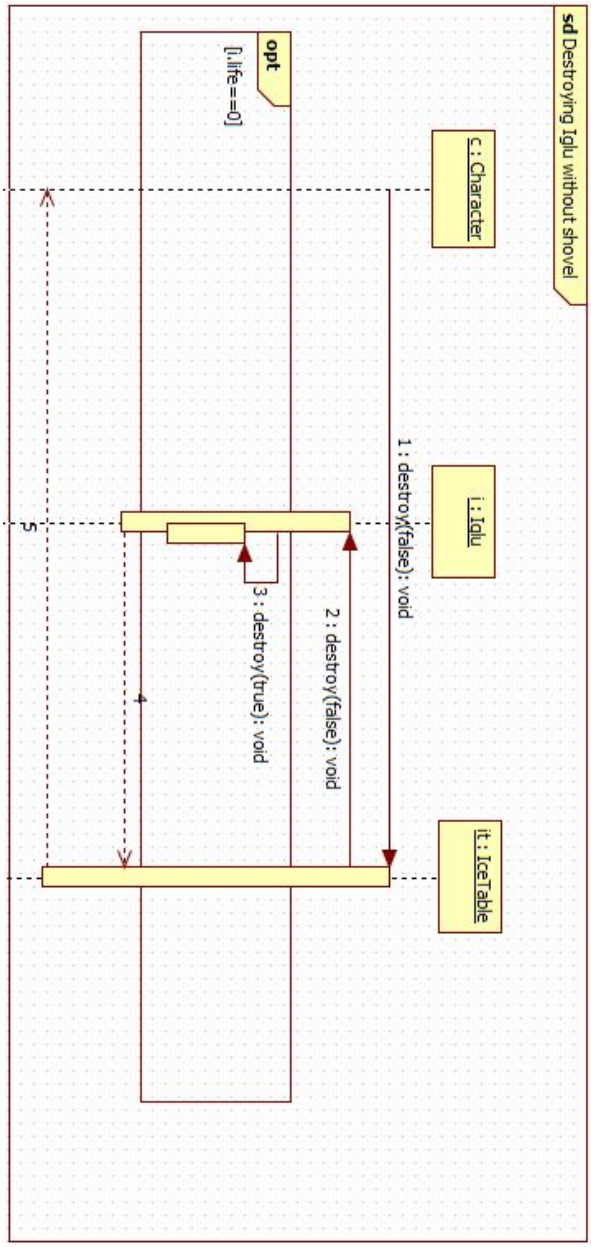
3.4.10 Building Iglu + Iglu Lifeline



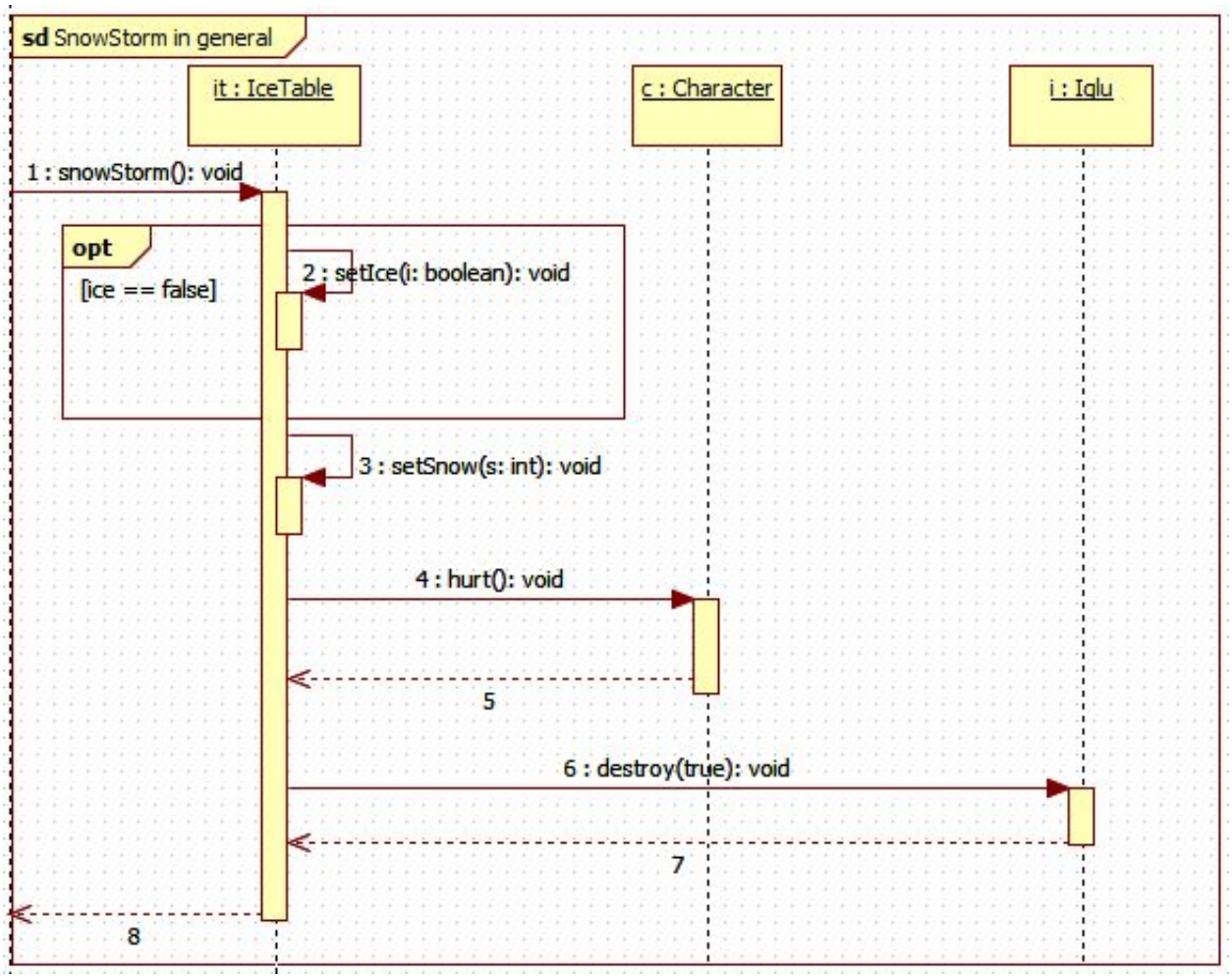
3.4.11 Destroying Iglu with shovel



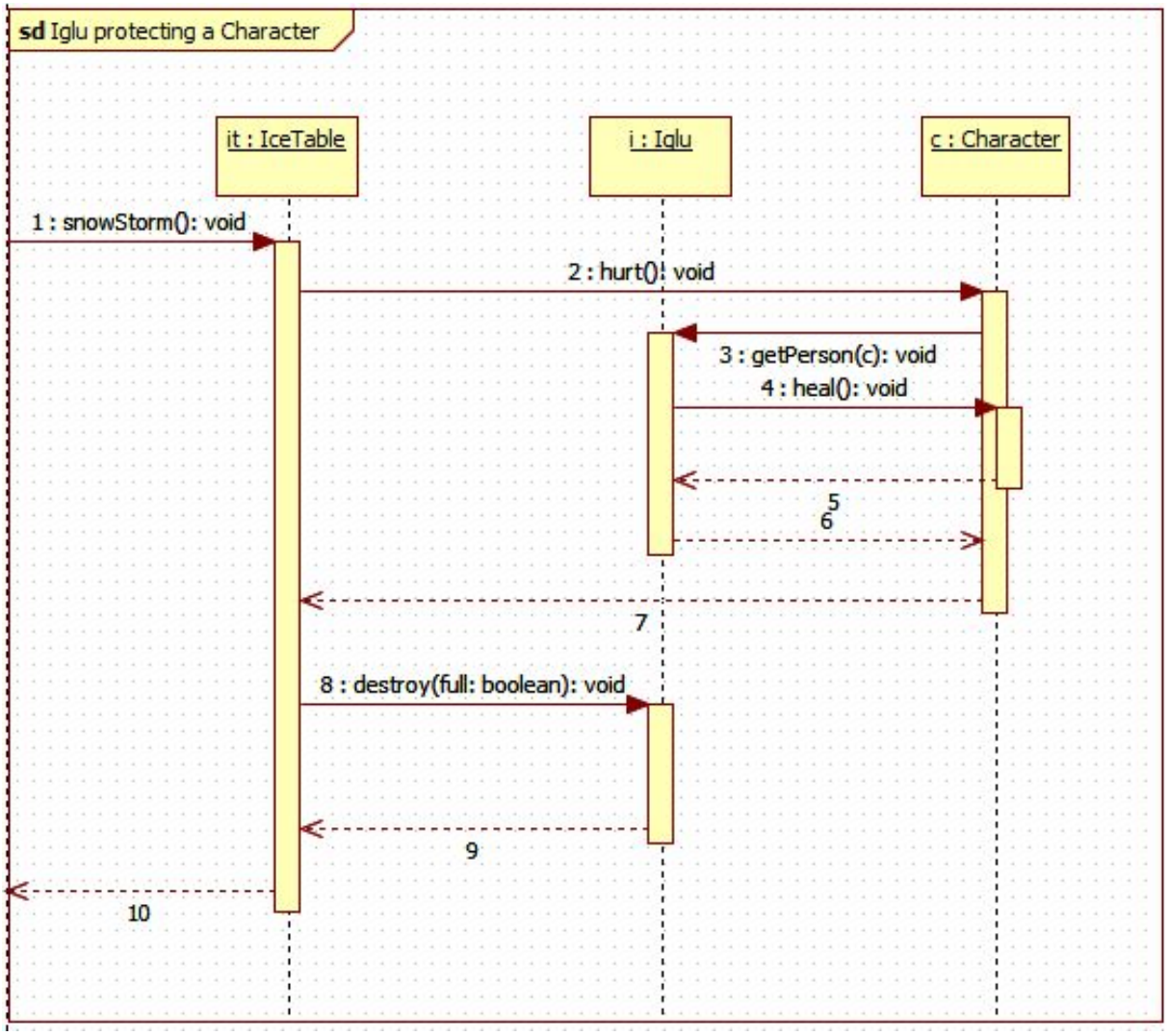
3.4.12 Destroying Iglue without shovel



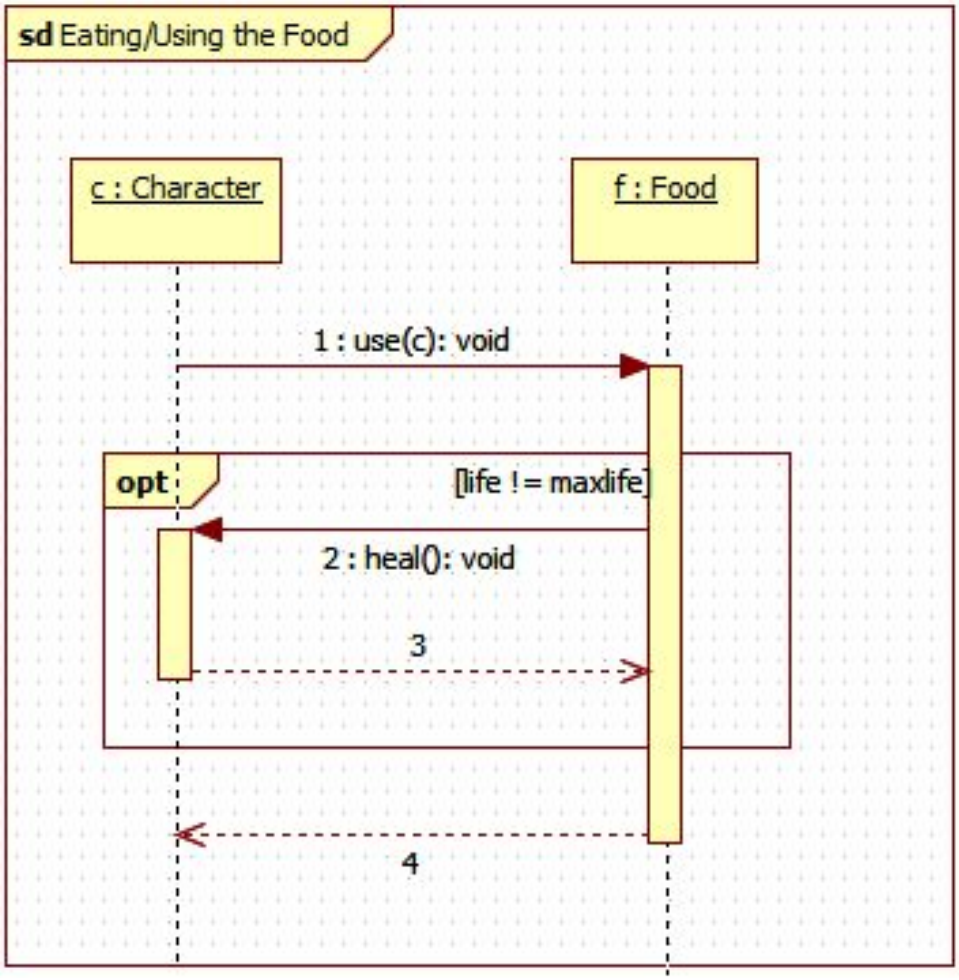
3.4.13 SnowStorm in general



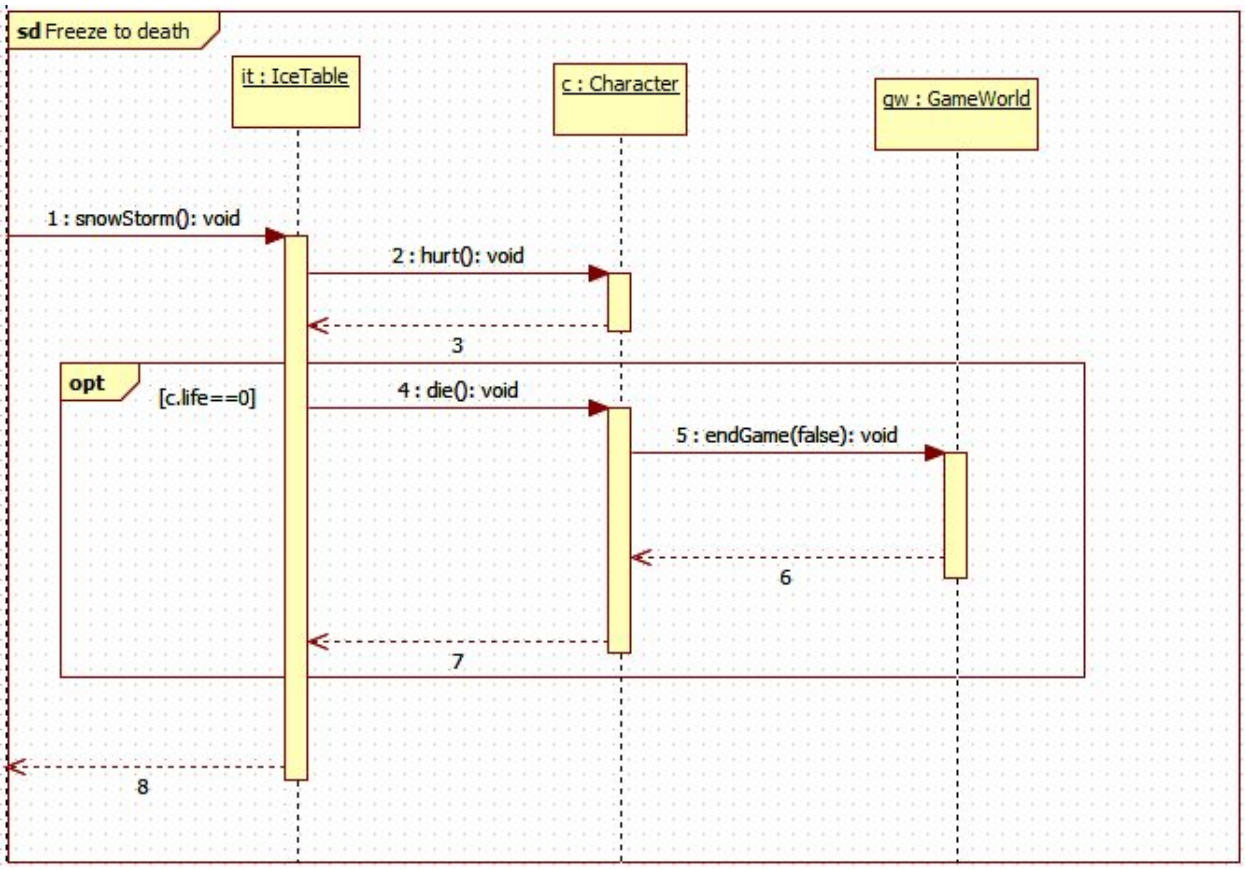
3.4.14 Iglu protecting a Character



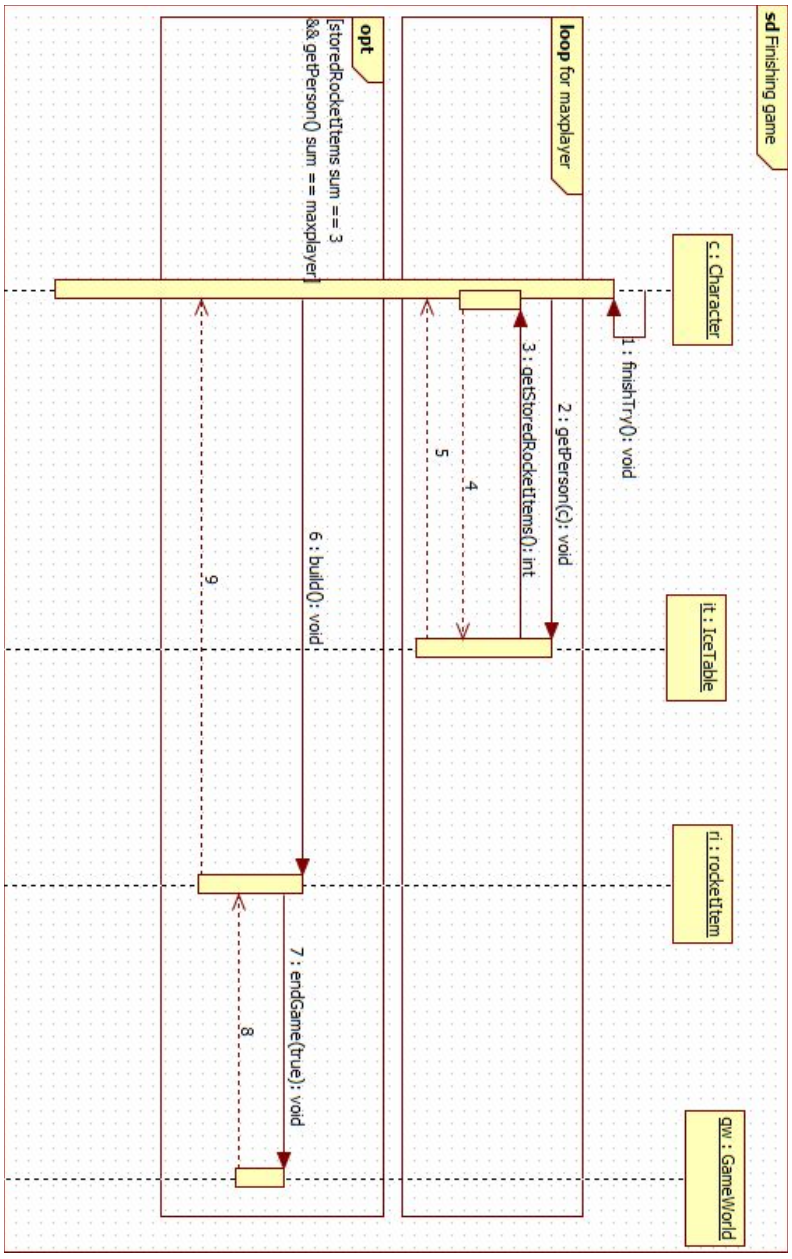
3.4.15 Eating/Using the Food



3.4.16 Freeze to death



3.4.17 Finishing game



3.5 Napló

Kezdet	Időtartam	Résztvevők	Leírás
2020.02.23. 17:00	2,5 óra	Ziaja	Osztálydiagram előzetes változata, főbb osztályok és metódusok megtervezése.
2020.02.25. 11:00	1 óra	Szepesi-Nagy Ziaja	Mindenki: Osztálydiagram átbeszélése, Objektum katalógus megtervezése és megírása
2020.02.25. 12:00	2 óra	Csoma Sipula Szepesi-Nagy Varga	Mindenki: Osztálydiagram kiegészítése, fontosabb módosítások átbeszélése. Objektum katalógus kiegészítése. Fontosabb szekvencia diagramok kigyűjtése
2020.02.26. 12:00	2 óra	Sipula Varga Ziaja	Mindenki: Konzultáción részvétel, csapat által felmerült kérdésekre válaszok keresése.
2020.02.26. 16:00	2 óra	Csoma Sipula Szepesi-Nagy Varga Ziaja	Mindenki: Közös meeting a konzultáción szerzett információk megbeszélésére, szekvencia diagramok és későbbi feladatok kiosztása. Osztálydiagrammal kapcsolatos fontosabb egyeztetések, javítások.
2020.02.28. 12:00	4 óra	Csoma Sipula Szepesi-Nagy Varga Ziaja	Mindenki: A kiosztott szekvencia diagramok elkészítése, azok megbeszélése.
2020.02.29. 9:00	2 óra	Ziaja	Osztálydiagram javítása, osztályok dokumentálása
2020.02.29. 12:00	5 óra	Varga	Szekvencia diagramok átnézése, hibásak javítása, újonnan felmerültek elkészítése.
2020.02.29. 21:00	2 óra	Sipula Varga	Szekvencia diagramok átnézése, véglegesítése.
2020.03.01. 20:00	3 óra	Sipula Szepesi-Nagy	Mindenki: Inicializálás szekvencia elkészítése, szekvencia diagramok dokumentumba helyezése
2020.03.02. 00:00	1 óra	Sipula	Dokumentum átnézése, végleges szerkesztése szerkesztése, napló megírása.

4. Analízis modell kidolgozása 2

4.0 Változtatások

4.0.1 Egy Character csak kézzel tud Iglut bontani, lapáttal nem.
4.0.2 Megszűnt a RocketItem osztály, a belőle leszármazó Gun, Stencil és Light is már usabelItem.
4.0.2.1 Ezáltal kikerültek a rocketItem-et kezelő függvények
4.0.3 Megszűnt az Obstacle interface, a függvényei átkerültek az Iglu-ba vagy az IceTable-be.
4.0.4 Kikerült a specificAction(), mivel a megvalósításhoz más paraméterek kellenek a leszármazottaknak, helyettük a checkStrength() és a buildIglu() függvény került a megfelelő osztályba.
4.0.5 Egy Characternél már bármennyi usableItem lehet.
4.0.6 A hóvihar a játékvilág felelőssége lett.

4.1 Objektum katalógus

4.1.1 Szereplő

Különböző képességű szereplőknek kell haladni a jégtáblákon, használni a tárgyakat és elsütni a jelzőrakétát, ezzel megnyerve a játékot.

4.1.1.1 Eszkimó

Egyfajta szereplő, akinek felelőssége alá tartoznak a tevékenységek végrehajtása, eszközök kezelése, a testhő és munkaráfordítás nyilvántartása és igluk építése.

4.1.1.2 Sarkkutató

Egyfajta szepelő, akinek felelőssége alá tartoznak a tevékenységek végrehajtása, eszközök kezelése, a testhő és munkaráfordítás nyilvántartása és a jégmezők teherbírásának vizsgálata.

4.1.2 Jégtábla

A jégtábla felelőssége alá tartozik, a rajtuk keletkezett hó mennyiségének nyilvántartása, az épült igluk megsemmisítése és hátralévő idejük számolása, illetve a rajtuk álló karakterek tárolása. Egy jégtábla lehet stabil vagy instabil, ha túl sokan állnak rajta (lyukason 1 is), akkor át kell fordulnia.

4.1.3 Jelzőrakéta

Az összeszerelt jelzőrakétával lehet megnyerni a játékot, az alkatrészek összeszereléséért felelős.

4.1.4 Pisztoly

A jelzőrakéta 3/1 (pisztoly) alkatrésze, ami szükséges a jelzőpisztoly elsütéséhez.

4.1.5 Patron

A jelzőrakéta 3/2 (patron) alkatrésze, ami szükséges a jelzőpisztoly elsütéséhez.

4.1.6 Jelzőfény

A jelzőrakéta 3/3 (jelzőfény) alkatrésze, ami szükséges a jelzőpisztoly elsütéséhez.

4.1.7 Élelem

Létezik élelem, aminek elfogyasztásával a szereplők az élethőjüket tudják növelni.

4.1.8 Kötél

A kötél segítségével tud egy karakter egy másik karaktert megmenteni a vízbefulladástól. Ennek felügyelete a Rope feladata.

4.1.9 Lapát

A lapáttal egy karakter két egység havat tud eltakarítani egy munkaráfordítással. Ennek felügyelete a lapát feladata.

4.1.10 Búvárruha

Buvárruhát viselő karakter vízbeesés esetén nem fullad meg. Ennek felügyelete a Swimsuit feladata.

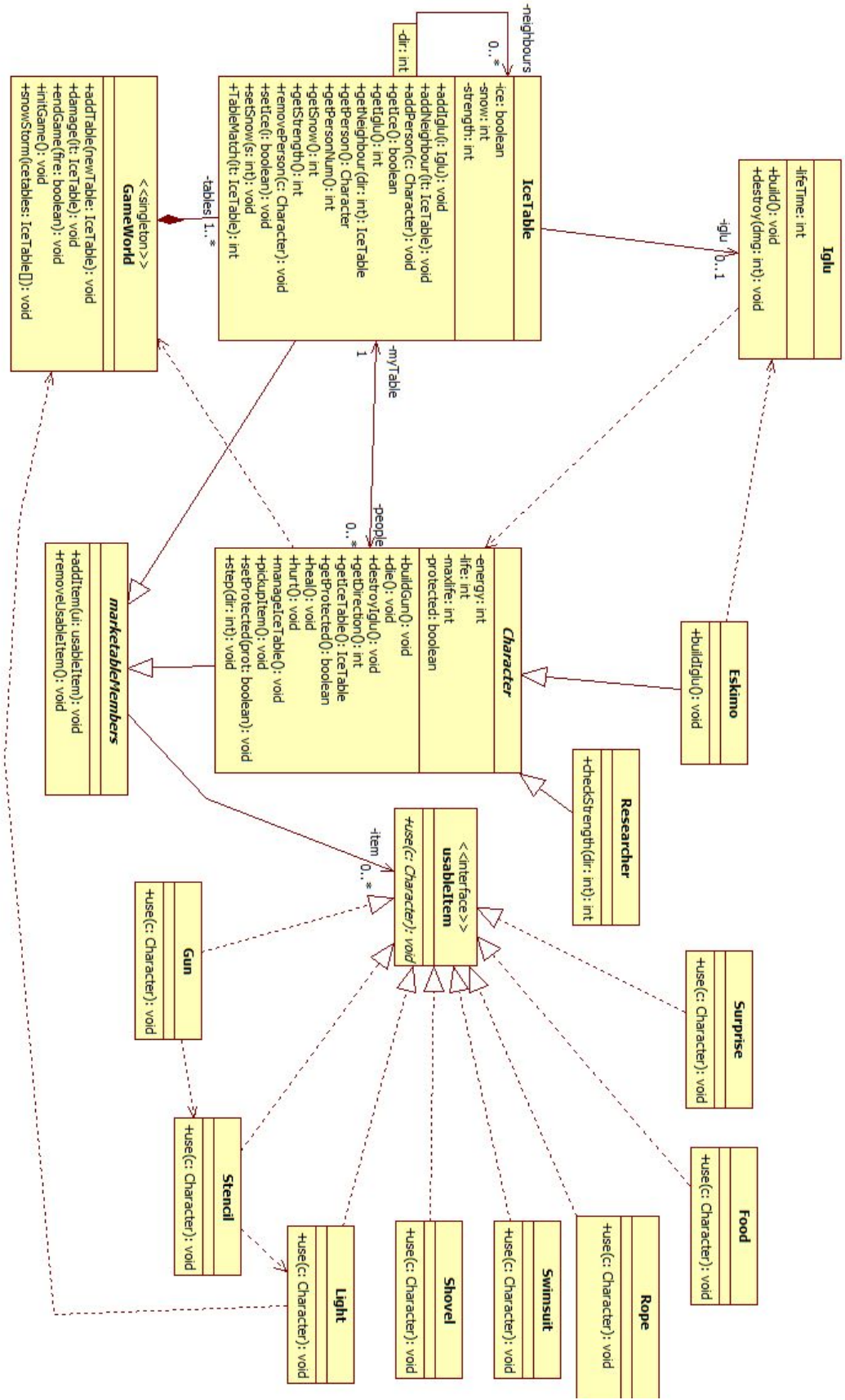
4.1.11 Iglu

Iglu állhat egy jégtáblán, a benne lévő szereplőket megvédi hóvihar esetén.

4.1.12 Meglepetés

Ez az entitás felelős a játék menetét befolyásoló jó vagy rossz kimenetelekért. Ennek használata felelős a játék körinek hóviharon kívüli befolyásolásáért (plusz cselekedet, körből kimaradás, testhő feltöltés stb.).

4.2 Statikus struktúra diagramok



4.3 Osztályok leírása

4.3.1 Character

- **Felelősség**

  A felelőssége alá tartozik a szomszédos mezőre lépés végrehajtása, saját testhő és energia nyilvántartása. Végül a fegyver megépítésére való próbálkozás is a feladata. Megtisztít jégtáblát hótól és jégtől egyaránt.

- **Ősosztályok**

  marketableMembers

- **Attribútumok**

  energy: int: A munkaegységek tárolására szolgál
  life: int: A testhő tárolására szolgál
  maxlife: int A karakter életének maximális száma
  myTable: IceTable: A saját jégtábla tárolására szolgál
  protected: boolean Vízbeesés esetén tárolja, hogy menthető-e

- **Metódusok**

  void buildGun(): A játékos megpróbálja összeépíteni a jelzőrakétát.
  void destroyIglu(): A játékos lebont egy életnyit az igluból.
  void die(): A szereplő vízbe esett, meg fog fulladni, ha nincs nála búvárruha. Ha megfullad, akkor játéknak vége, vesztett a csapat.
  int getDirection(): Getter függvény a kért irányra.
  IceTable getIceTable(): Getter függvény az IceTable-re
  boolean getProtected(): Getter függvény a protectedre.
  void heal(): A játékos kap egy testhőt.
  void hurt(): A játékos testhője eggyel csökken.
  void manageIceTable(): A játékos havat vagy jeget távolít el a jégtábláról.
  void pickupItem(): A játékos felvesz egy itemet a jégtábláról.
  void setProtected(boolean prot): Setter függvény a protectedre.
  void step(int dir): A játékos átlép az adott irányban, elkéri az adott irányú szomszédot és szól a tábláknak, hogy átlépett.

4.3.2 Eskimo

- **Felelősség**

  Eszkimónak kell tudnia iglut építeni és ezen kívül megörökli a Character összes felelősségét.

- **Ősosztályok**

  Character marketableMembers

- **Metódusok**
  void buildIglu(): Iglu megépítése.

4.3.3 Food

- **Felelősség**
  A felhasználójának egy testhőt biztosít.
- **Interfészek**
  usableItem
- **Metódusok**
  void use(Character c): Az adott karakternek eggyel kell növelni a testhőjét.

4.3.4 GameWorld

- **Felelősség**
  A játék felépítése, jégtáblák tárolása, hóviharok keltése, egy kör leteltének következményeinek kezelése, és a játéknak véget vetni.
- **Attribútumok**
  tables: IceTable[ ]: A táblák tárolása, hogy azokon hóvihar keletkezhessen, hogy bomoljanak a táblák iglujai.
- **Metódusok**
  void addTable(IceTable newTable): Új mező hozzáadásáért felelős.
  void damage(IceTable it): Adott mezőben kárt okoz, azaz a rajta lévő iglut megsemmisíti vagy karakterek életét megcsökkenti.
  void endGame(boolean fire): Ha a fire true, akkor nyertek a játékosok, vége a játéknak. Ha false, akkor vesztettek, de ugyanúgy vége a játéknak.
  void initGame(): A játék létrehozásáért felelős függvény.
  void snowStrom(IceTable[ ] icetables): Hóvihart kelt a mezőkön.

4.3.5 Gun

- **Felelősség**
  A pisztoly entitása, összeépülés elindítása.
- **Interfészek**
  usableItem

- **Metódusok**

  void use(Character c): Pisztoly beleépül a rakétába, a patron beépítésének intéz

4.3.6 IceTable

- **Felelősség**

  Egy jégtáblán lévő igluk, karakterek, szomszédok kezelése.

- **Ősosztályok**

  marketableMembers

- **Attribútumok**

  ice: boolean: A jégtáblán lévő jég tárolása.
  iglu: Iglu: A jégtáblán álló igluk nyilvántartása.
  neighbors: IceTable: Szomszédos táblák tárolása
  people: Character: A jégtáblán álló szereplők nyilvántartása.
  snow: int: A jégtáblán lévő hó tárolása
  strength: int: A jégtábla teherbírásának tárolására használandó

- **Metódusok**

  void addIglu(Iglu i): Kapott iglu eltárolása sajátként.
  void addNeighbour(IceTable it): Egy mező szomszédjának a regisztrálásáért felelős függvény.
  void addPerson(Character c): c karakter elfogadása, azaz elmenteni people közé. Ha nem sikeres az átlépés (lyuk vagy instabil mező), akkor arról tájékoztat mindenkit, hogy vízbe esett.
  boolean getIce(): Getter függvény a jégre.
  int getIglu(): Getter függvény az iglu mennyiségére.
  IceTable getNeighbour(int dir): Getter függvény az kapott irányú szomszédra.
  Character getPerson(): Visszatér egy jégtáblán lévő játékossal (a Jégtábla embereket tároló list első elemével.)
  int getPersonNum(): Getter függvény a táblán álló karakterek mennyiségére.
  int getSnow(): Getter függvény a hóra.
  int getStrength(): Getter függvény a teherbírásra.
  void removePerson(Character c): Eltávolítja a kapott karaktert a tárolt karakterei közül.
  void setIce(boolean i): Setter függvény a jégre.
  void setSnow(int s): Setter függvény a hóra.
  int TableMatch(IceTable it): Visszatér azzal, hogy a kapott jégtábla hol van hozzá képest.

4.3.7 Iglu

- **Felelősség**
  Hátralévő élet számontartása, benne álló karakter(ek) megvédése hóvihar esetén.

- **Attribútumok**
  lifeTime: int: Hátralévő idő nyilvántartása

- **Metódusok**
  void build(): Megépül és szól a jégtáblának, hogy megépült.
  void destroy(int dmg): Kapott érték levonása a hátralévő időből.

4.3.8 Light

- **Felelősség**
  A jelzőfény nyilvántartása és megvillantása.

- **Interfészek**
  usableItem

- **Metódusok**
  void use(Character c): Fény beépül a rakétába, így elkészült a fegyver, felvillant a fény, tehát véget vet a játéknak, a játékosok nyertek.

4.3.9 marketableMembers

- **Felelősség**
  Rakétaalkatrészek és tárgyak átruházásához szükséges ős.

- **Attribútumok**
  item: usableItem: Tárgyak tárolása.

- **Metódusok**
  void addItem(usableItem ui): Átvesszük a tárgyat
  void removeUsableItem(): Leadjuk a tárgyat

4.3.10 Researcher

- **Felelősség**
  Sarkkutatók derítik fel a mezők teherbírását és ezen kívül megörökli a Character összes felelősségét.

- **Ősosztályok**
  Character marketableMembers

- **Metódusok**
  int checkStrength(): Egy szomszédos mező felderítése, a saját táblán keresztül, visszatér az kért teherbírással.

4.3.11 Rope

- **Felelősség**
  Egy karakter kötéllel való kimentésének levezénylése.

- **Interfészek**
  usableItem

- **Metódusok**
  void use(Character c): Kimenti a vízbe esett karaktert a vízből az általa kapott ember jégtáblájára.

4.3.12 Shovel

- **Felelősség**
  Használatakor elvárt, hogy két mennyiségű hó eltűnjön a tábláról.

- **Interfészek**
  usableItem

- **Metódusok**
  void use(Character c): A kapott karakternek mezőjéről eltűntet két mennyiségű havat.

4.3.13 Stencil

- **Felelősség**
  Patron a jelzőrakéta egy alkatrésze, összeépíthető.

- **Interfészek**
  usableItem

- **Metódusok**

  void use(Character c): A patron beszerelése a pisztolyba, majd a jelzőfény beépítésének intézése.

4.3.14 Surprise

- **Felelősség**

  A játék befolyásolása meglepetésszerűen

- **Interfészek**

  usableItem

- **Metódusok**

  void use(Character c): Meglepetés végrehajtása.

4.3.15 Swimsuit

- **Felelősség**

  Játékos megmentése, ha lyukra lép.

- **Interfészek**

  usableItem

- **Metódusok**

  void use(Character c): A karakter protected-jét igazra állítja, így ha az lyukba esik, úgy viselkedik, mintha mezőn állna, tovább tud lépni (kimászik a lyukból).

4.3.16 usableItem

- **Felelősség**

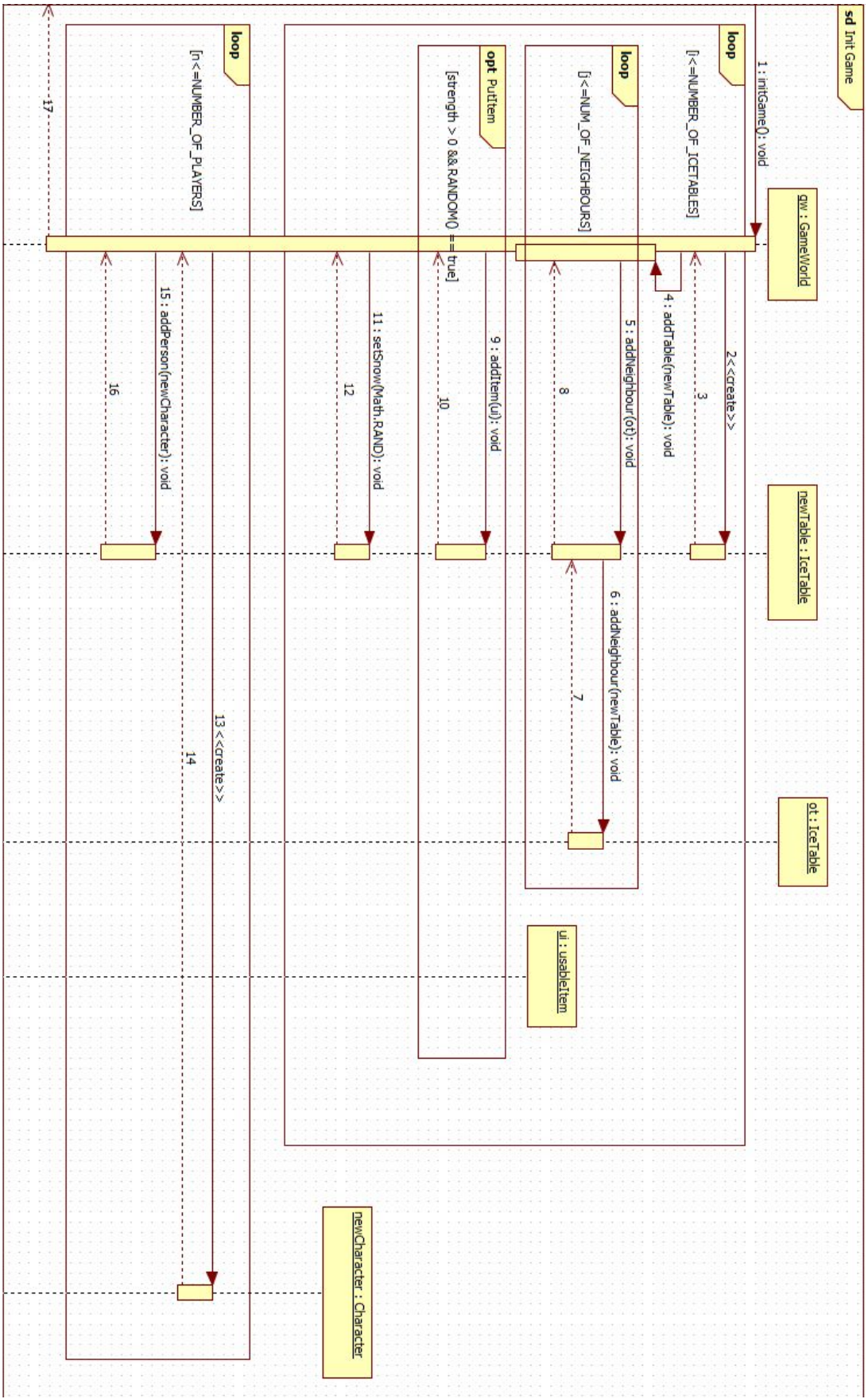
  Használható tárgyak interfésze, minta a tárgyaknak.

- **Metódusok**

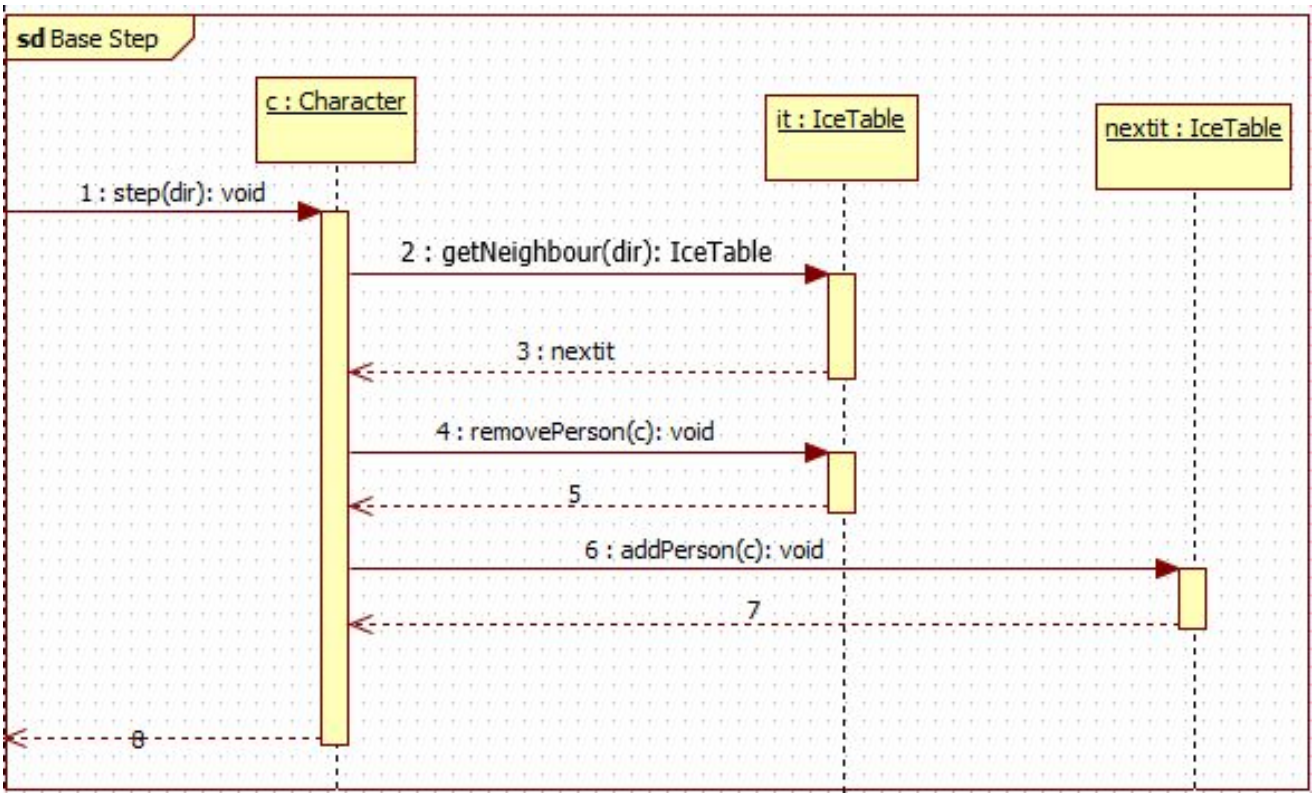
  void use(Character c): Absztrakt metódus, minta, hogy hogyan kell kinéznie egy tárgynak.

4.4 Szekvencia diagramok

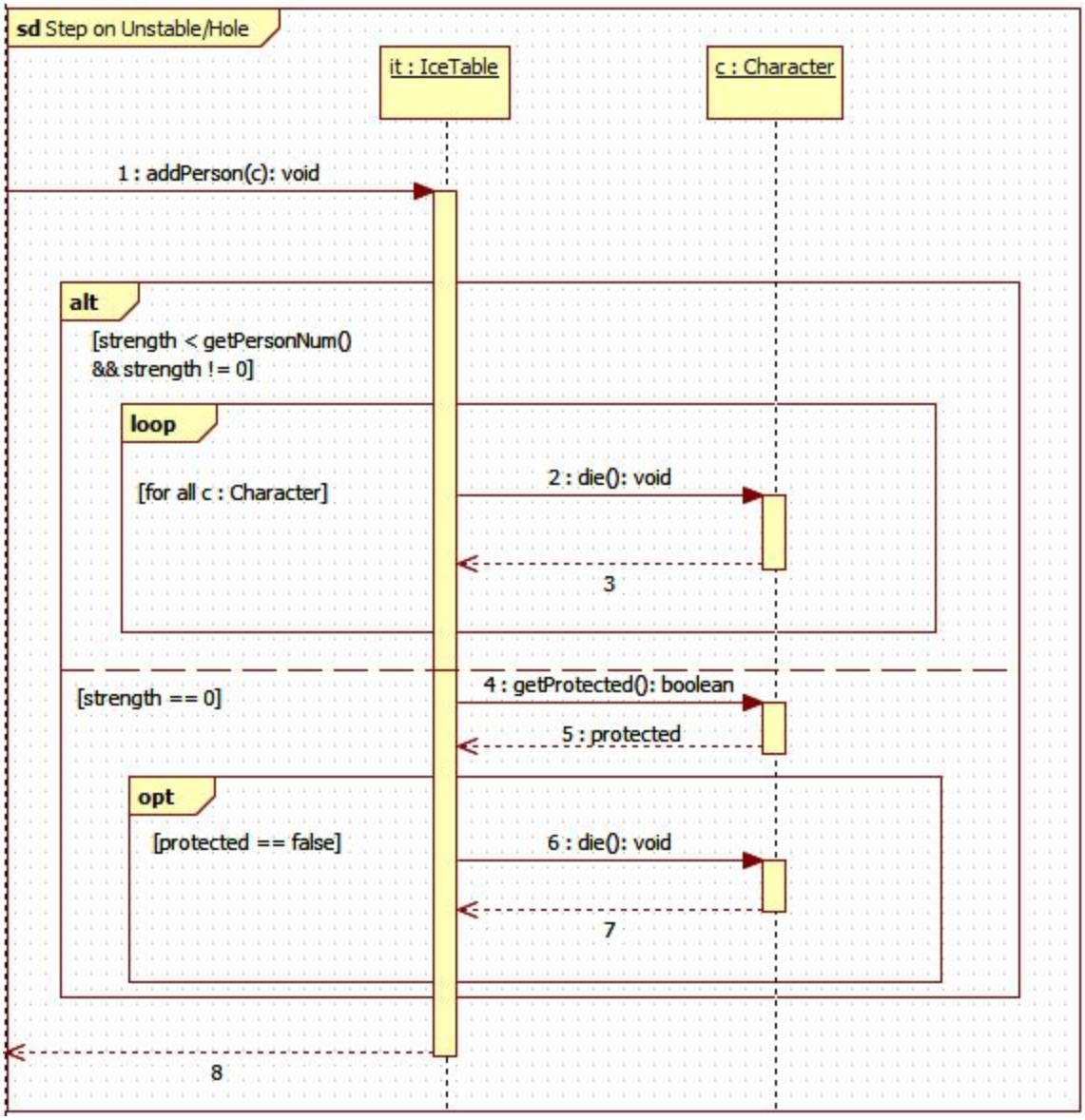
4.4.1 Game initialization



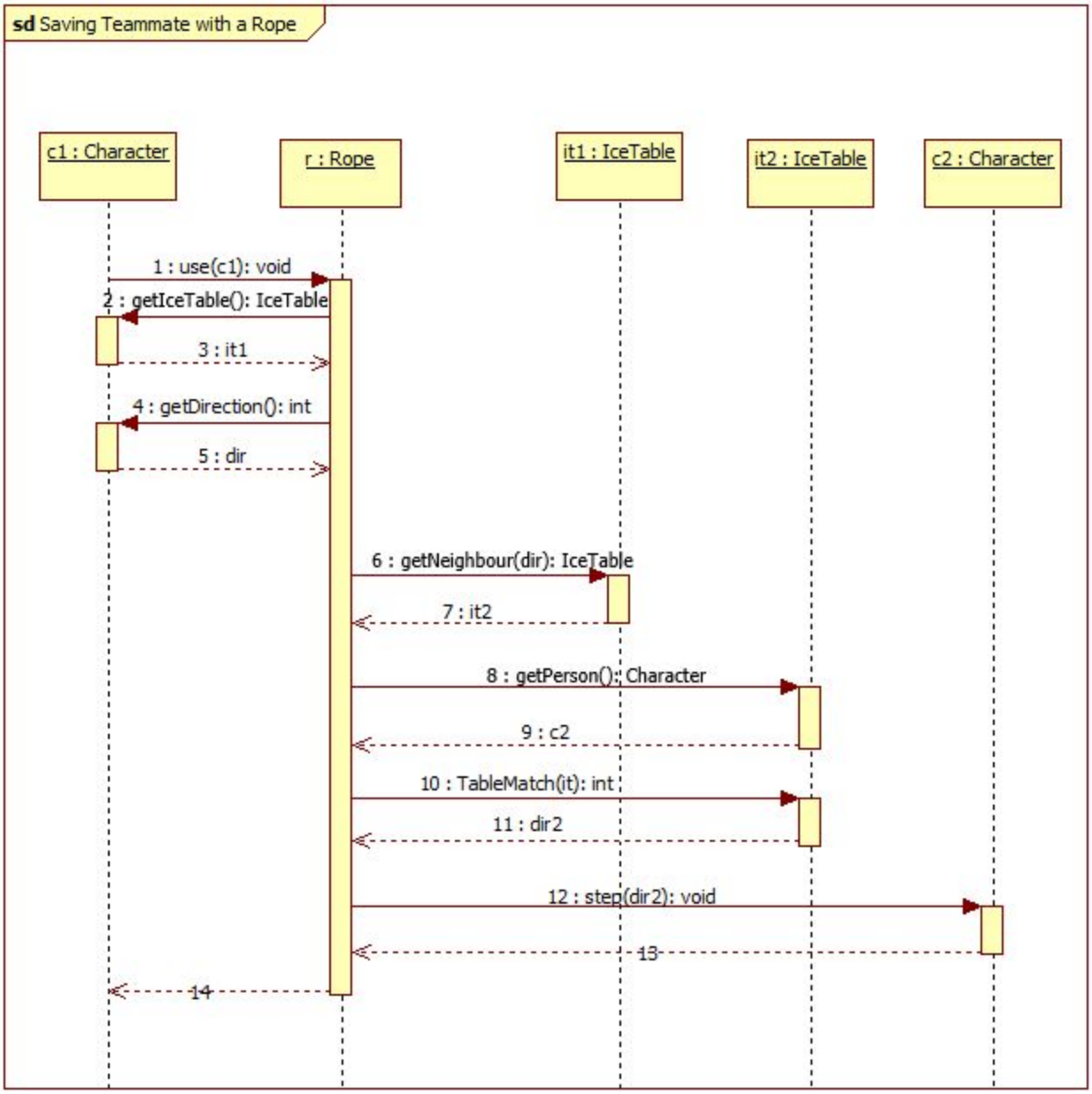
4.4.2 Stepping



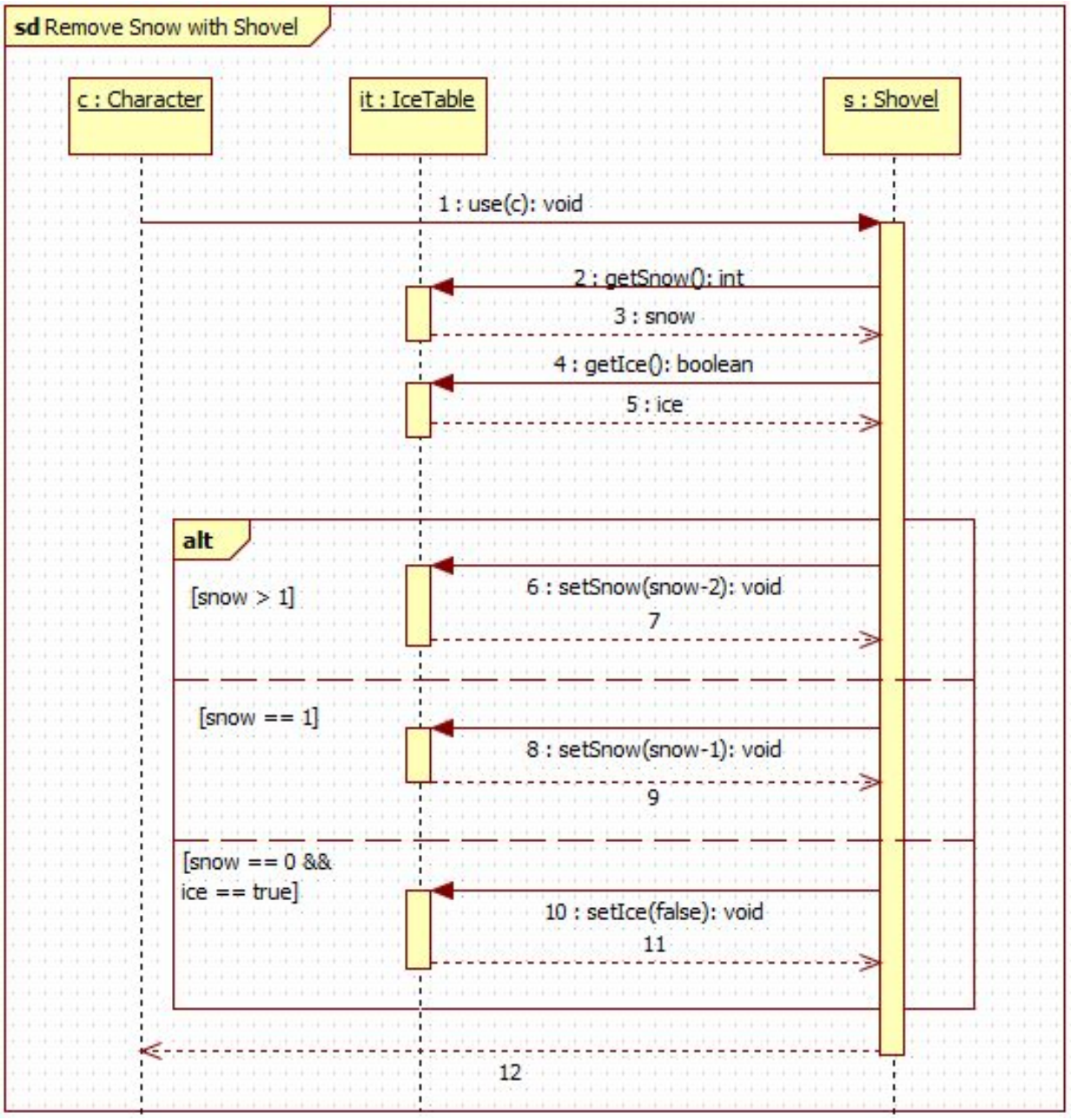
4.4.3 Stepping onto unstable or hole



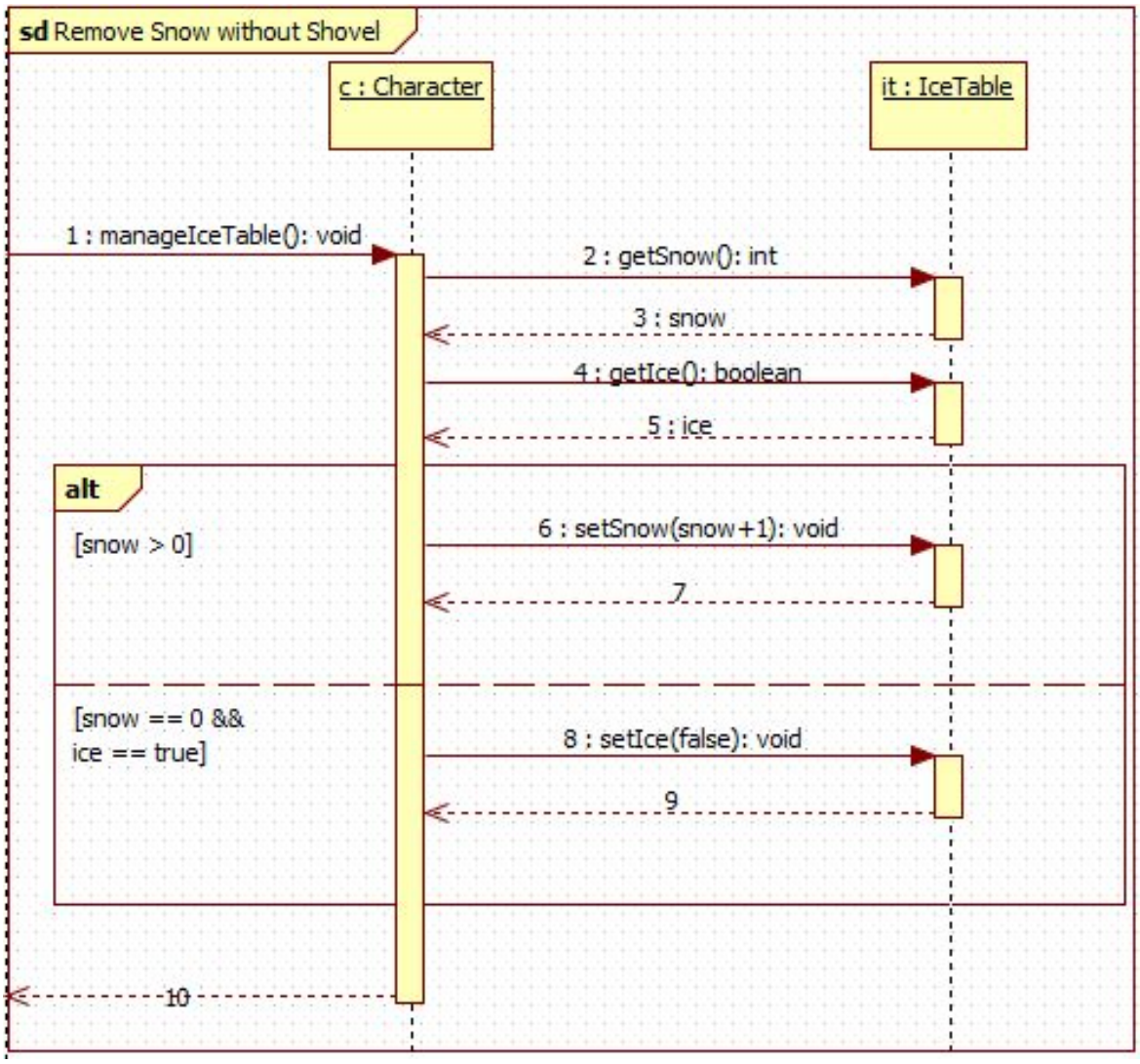
4.4.4 Saving Teammate with a Rope



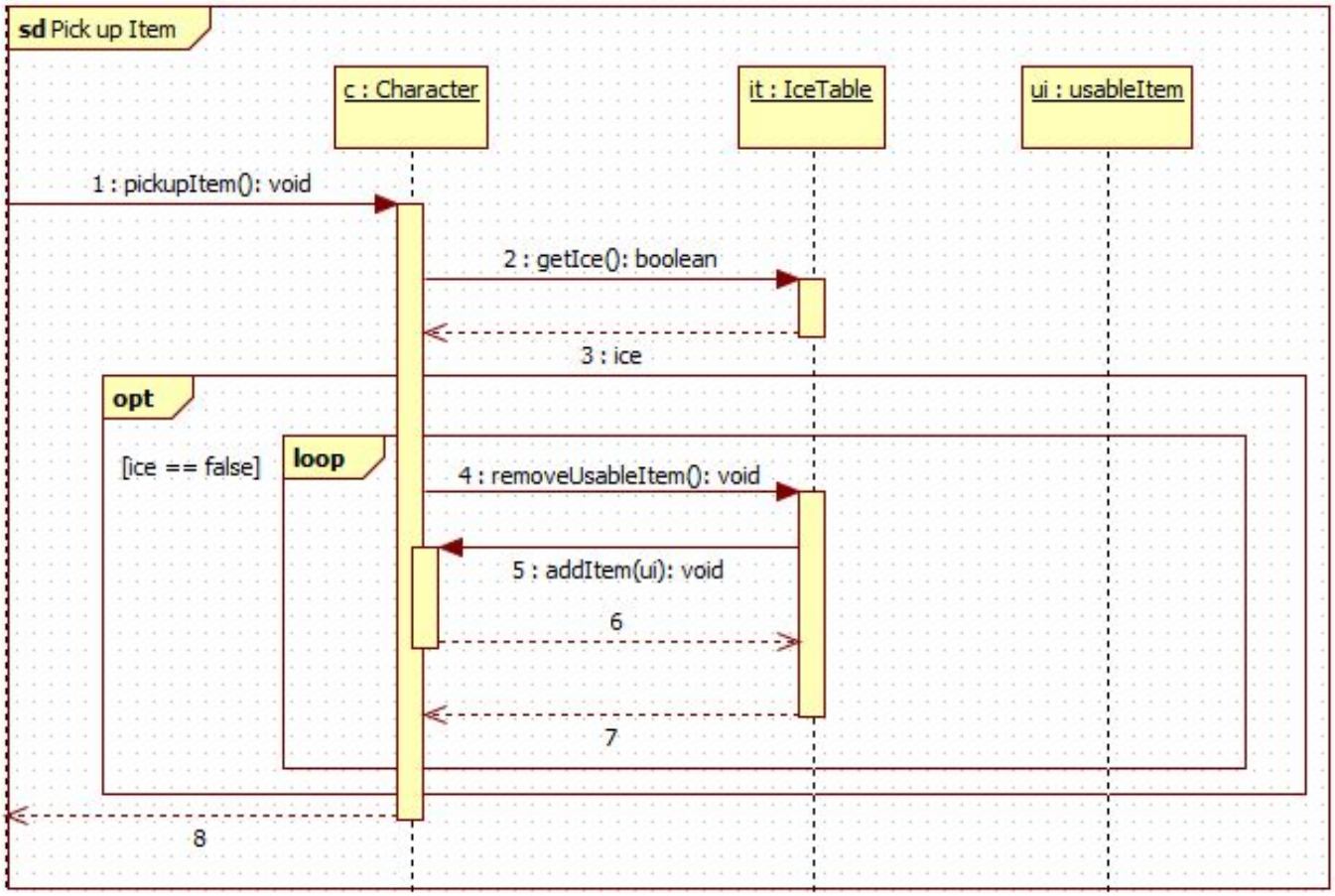
4.4.5 Removing snow with shovel



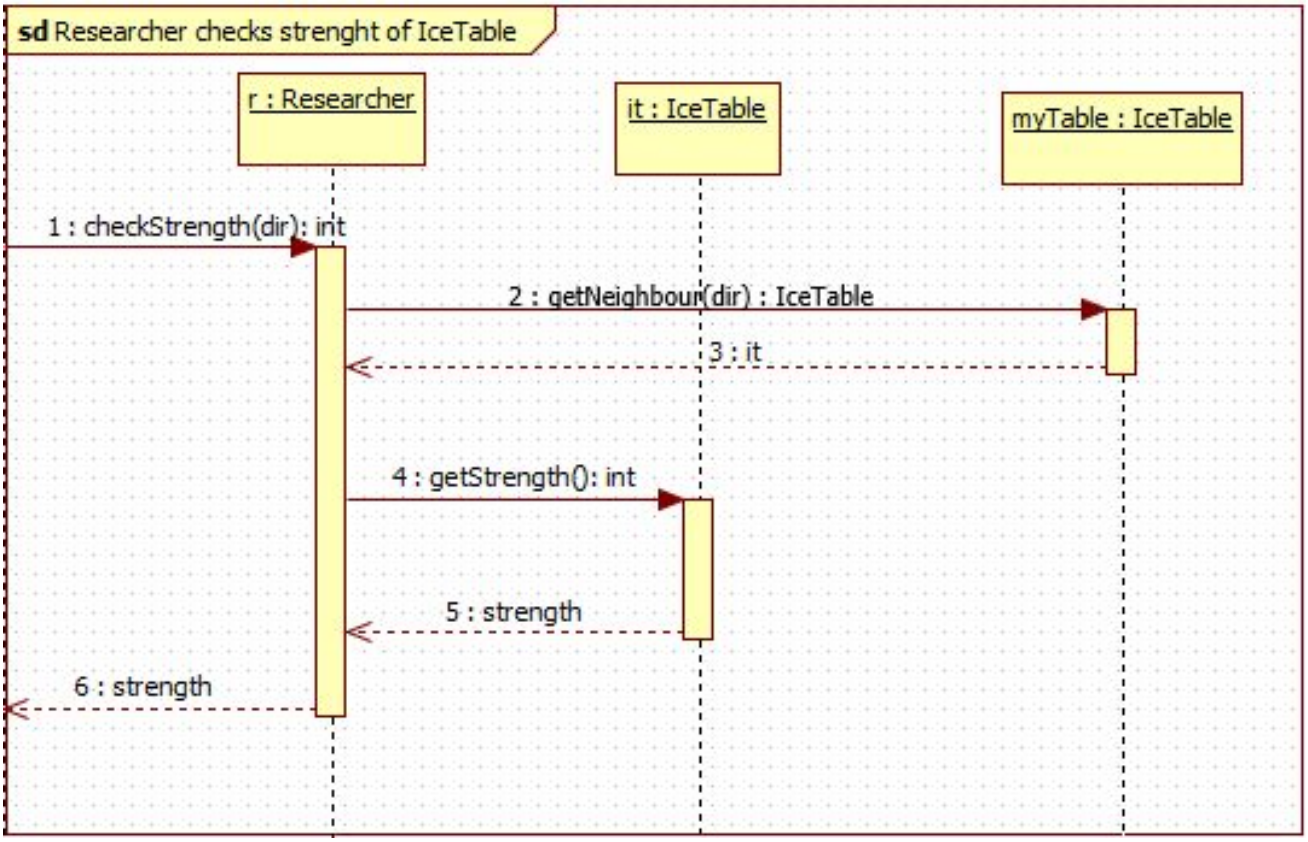
4.4.6 Removing snow without shovel



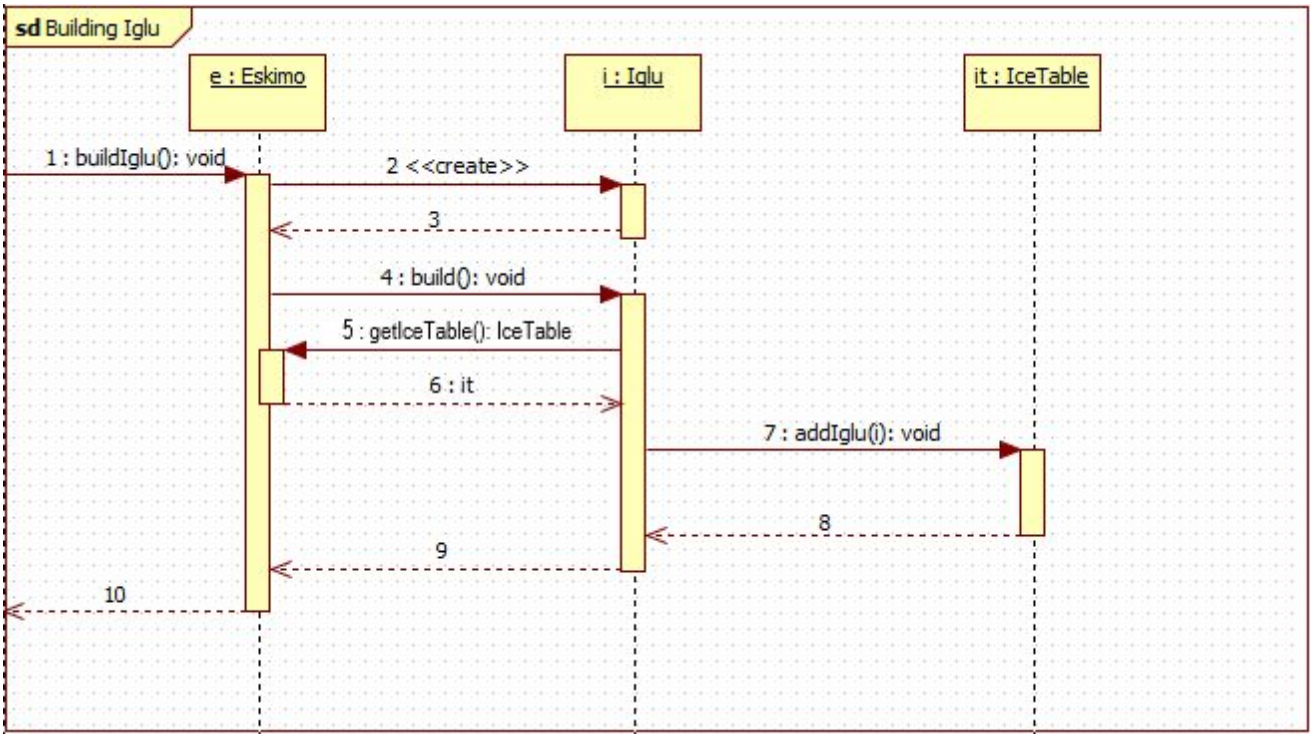
4.4.7 Picking up Item



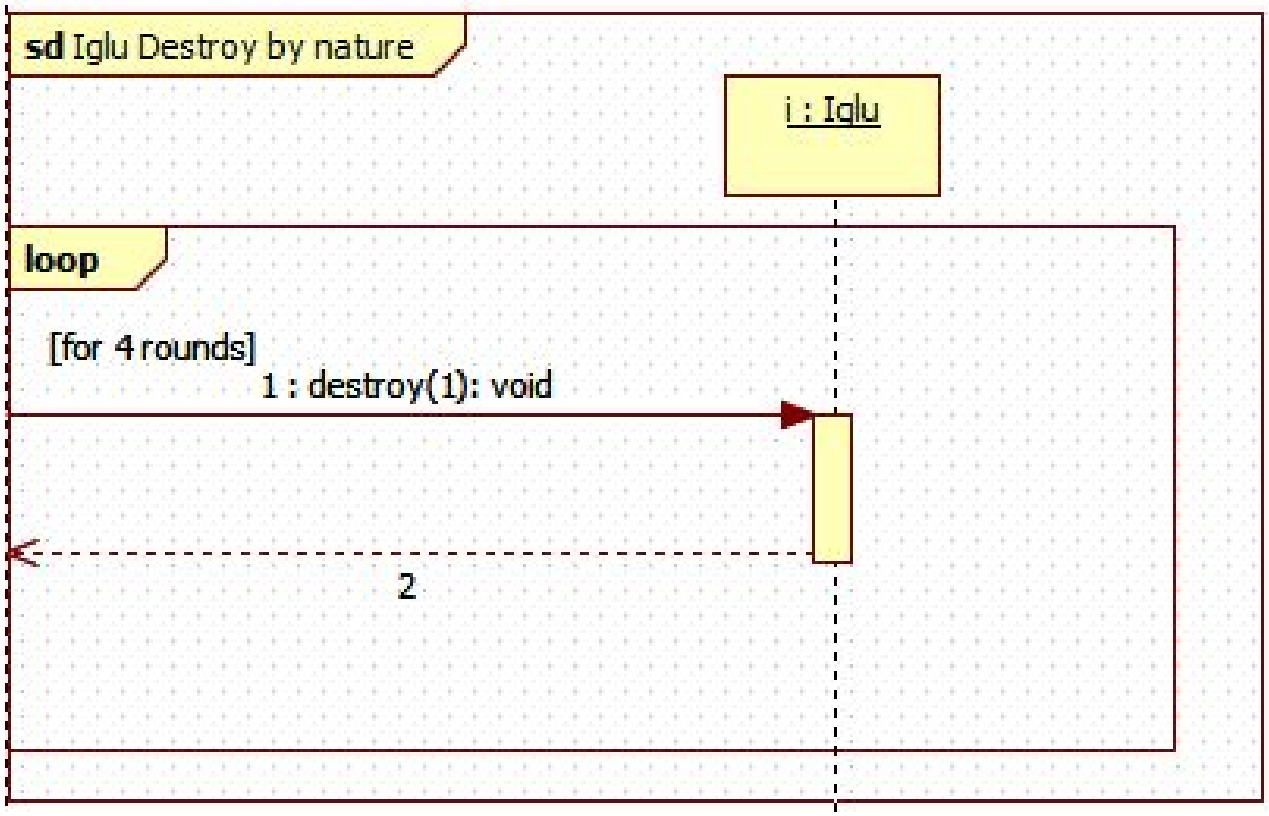
4.4.8 Researcher checks strength of IceTable



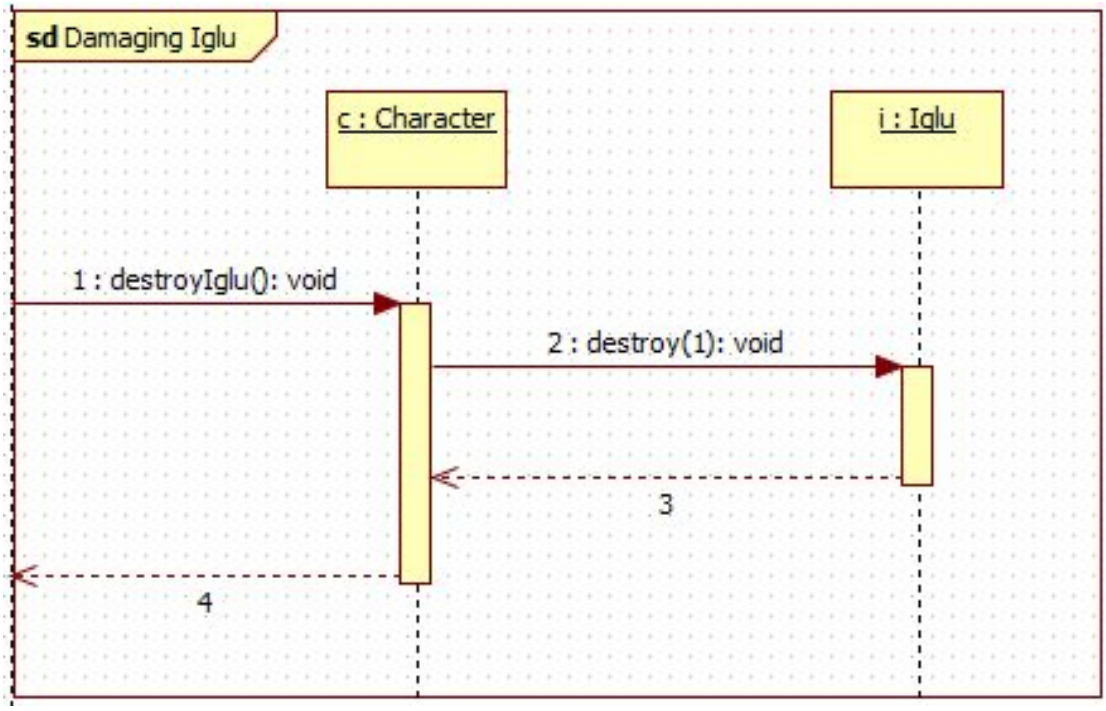
4.4.9 Building Iglu + Iglu Lifeline



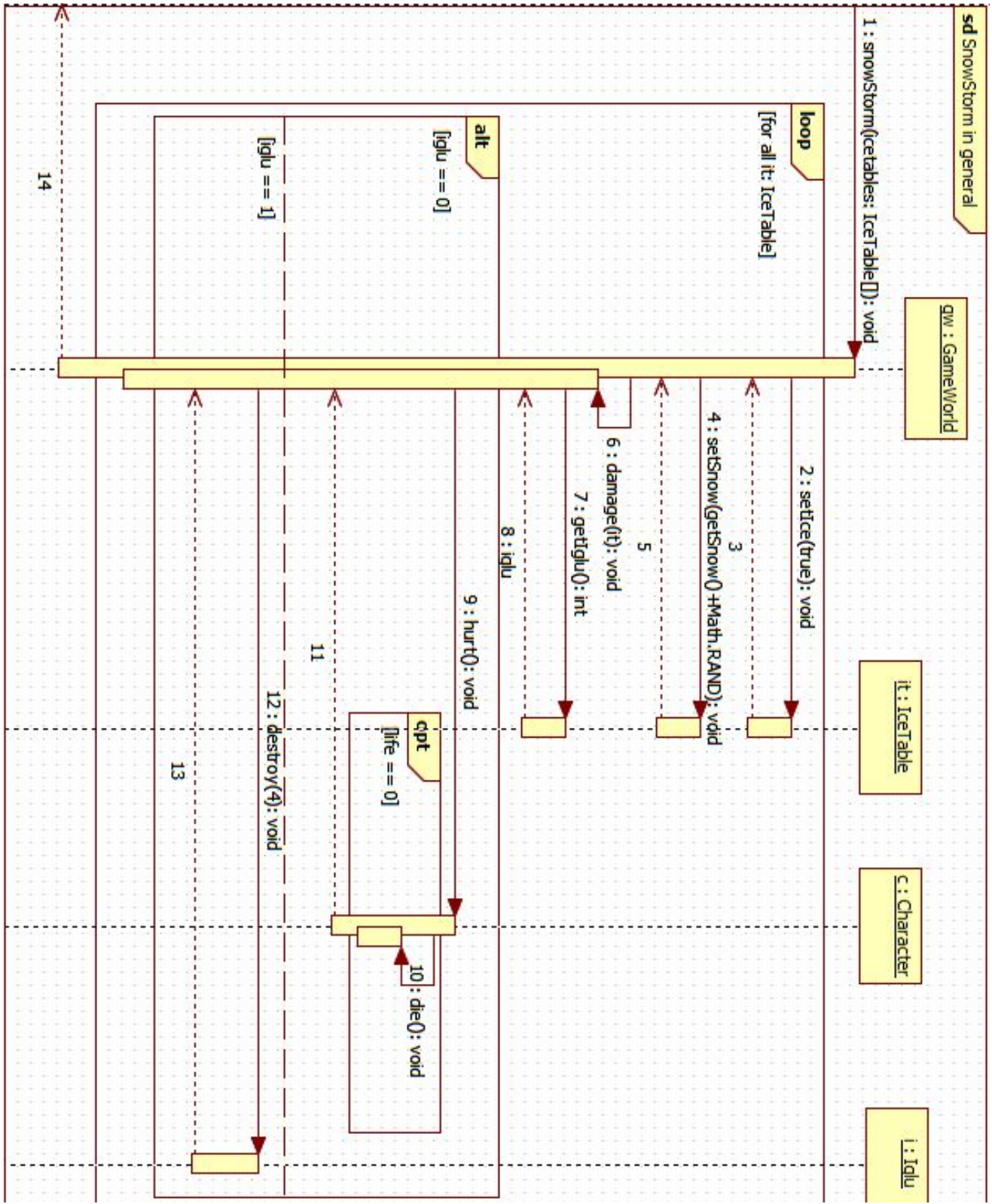
4.4.10 Destroying Iglu by nature



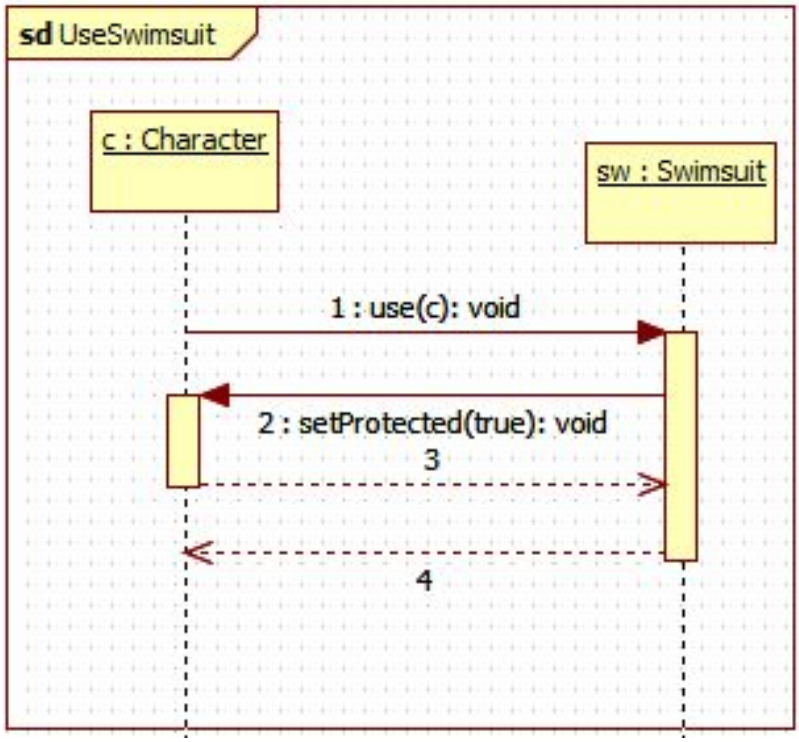
4.4.11 Destroying Iglu



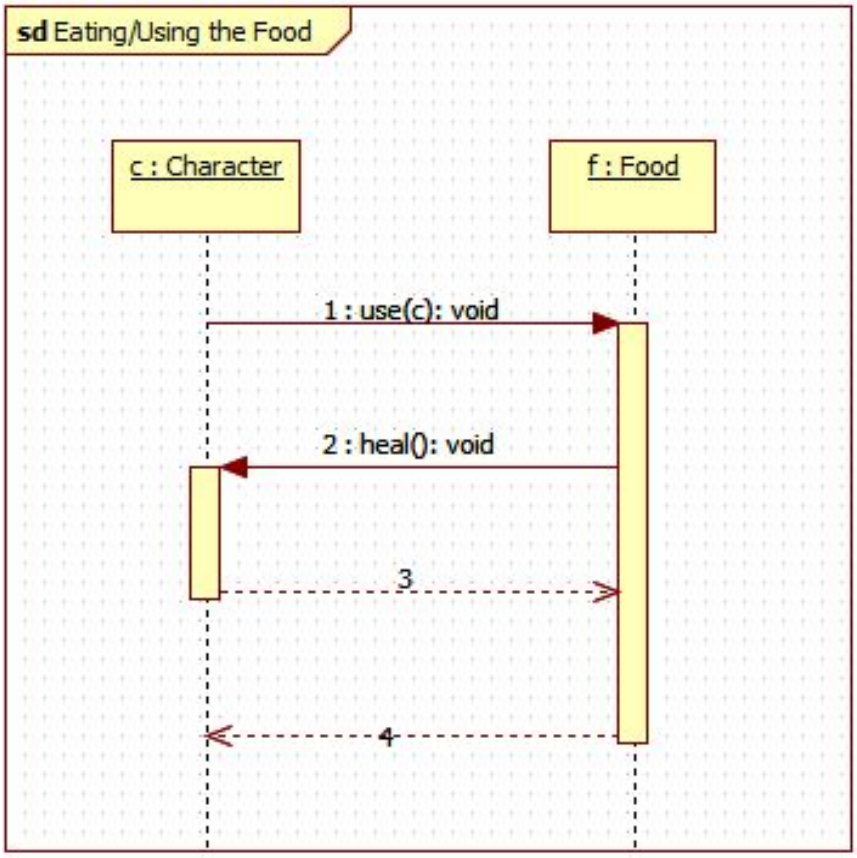
4.4.12 SnowStorm in general



4.4.13 Use Swimsuit

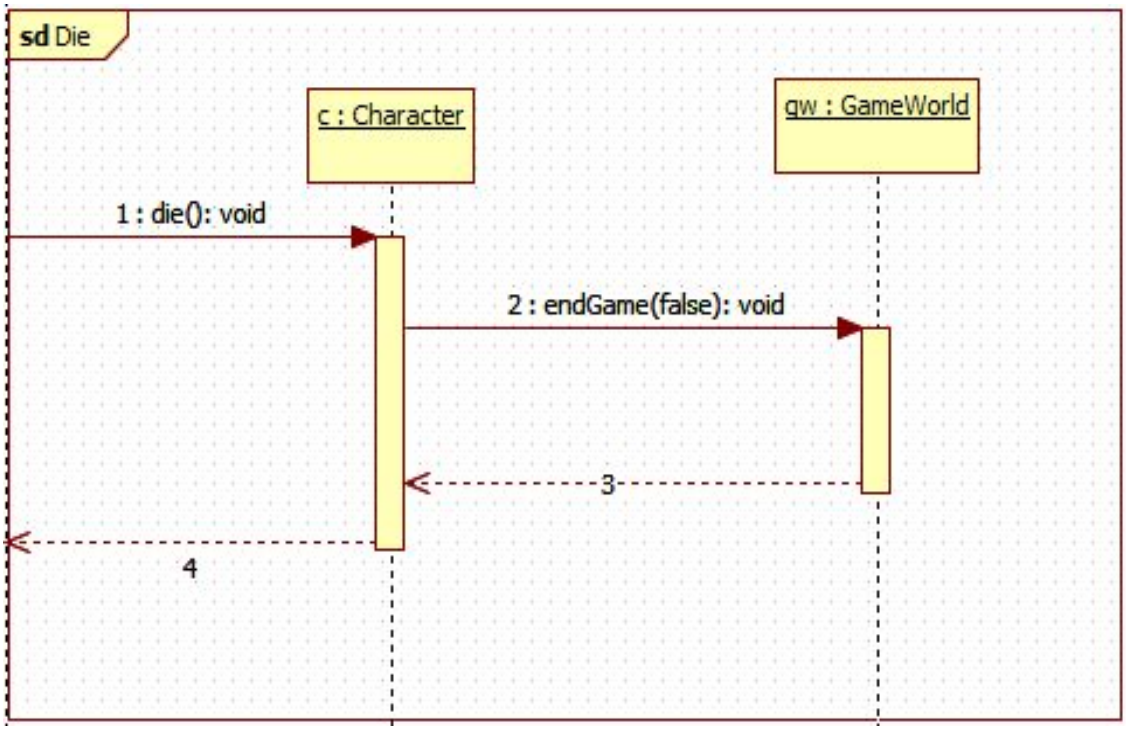


4.4.14 Using the Food

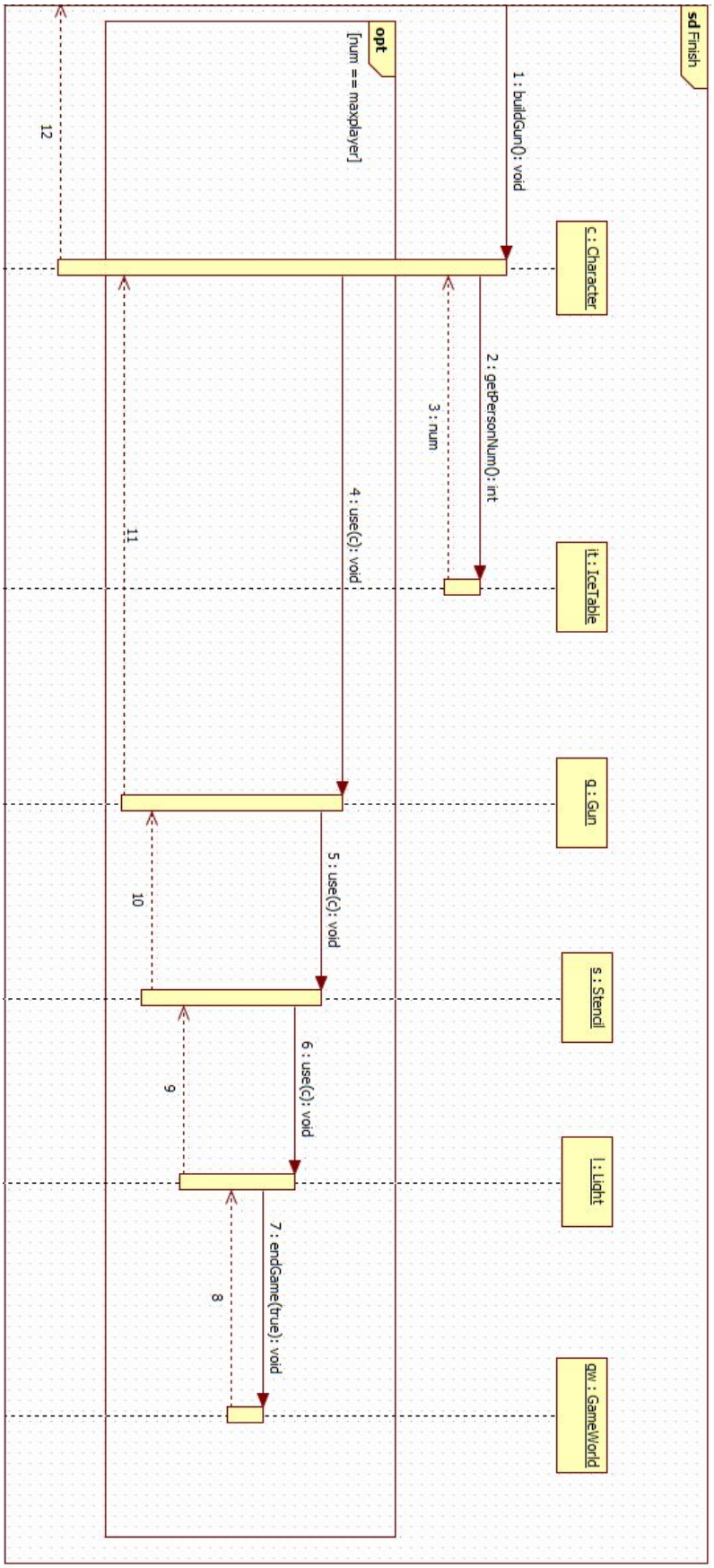


4.4.15 Surprise

4.4.16 Die


sd Die
c : Character
gw : GameWorld
1 : die(): void
2 : endGame(false): void
3
4

4.4.17 Finish


sd Finish
opt
[num == maxplayer]
1 : buildGun(): void
2 : getPersonNum(): int
3 : num
4 : use(c): void
5 : use(c): void
6 : use(c): void
7 : endGame(true): void
8
9
10
11
12
c : Character
it : IceTable
g : Gun
s : Stencil
l : Light
gw : GameWorld

4.5 Napló

Kezdet	Időtartam	Résztvevők	Leírás
2020.03.04. 12:00	2 óra	Sipula Varga	Mindenki: Konzultáción részvétel, információk begyűjtése a javításhoz.
2020.03.04. 16:00	2 óra	Csoma Sipula Szepesi-Nagy Varga Ziaja	Mindenki: Közös meeting a konzultáción szerzett információk megbeszélésére. Osztálydiagrammal kapcsolatos fontosabb egyeztetések, javítások.
2020.03.06. 12:00	6 óra	Sipula Varga	Mindenki: Szekvencia diagramok újratervezése és elkészítése a meetingen megbeszéltek szerint. Osztálydiagram módosítása.
2020.03.07. 9:00	4 óra	Sipula Varga Ziaja	Mindenki: Osztálydiagramnál felmerült kérdések egyeztetése, szekvencián felmerült kérdések megbeszélése, hibák kijavítása, osztálydiagram véglegesítése.
2020.03.07. 17:00	2 óra	Csoma Szepesi-Nagy	Mindenki: Szekvencia diagramok és osztálydiagram átnézése és ellenőrzése.
2020.03.08. 08:00	1 óra	Ziaja	Osztályok dokumentációjának szerkesztése, újítások dokumentálása.
2020.03.08. 16:00	2 óra	Sipula Varga	Mindenki: Talált hibák javítása, diagramok dokumentumba helyezése.
2020.03.08. 18:00	1 óra	Sipula	Dokumentum végleges módosítása, napló megírása.
2020.03.08. 19:00	1 óra	Csoma Szepesi-Nagy	Dokumentum átnézése.

6. Szkeleton beadás

6.1 Fordítási és futtatási útmutató

6.1.1 Fájllista

Fájl neve	Méret(Byte)	Keletkezés ideje	Tartalom
Character.java	8750	2020.03.29. 21:55	A karakterek ősosztálya, közös függvények megvalósítása.
Eskimo.java	1543	2020.03.29. 21:55	A játékban szereplő egyik karaktertípust, az eszkimó és annak speciális képességének(iglu építés) megvalósítása.
Food.java	1417	2020.03.29. 21:55	Az étel használatának megvalósítása.
GameWorld.java	4905	2020.03.29. 21:55	A játékkezelő megvalósítása.
Gun.java	1283	2020.03.29. 21:55	A fegyver megvalósítása, ami a jelzőpisztoly első alkatrésze.
IceField.bat	34	2020.03.29. 21:55	A parancssori indítást leegyszerűsítő bat fájl.
IceTable.java	7808	2020.03.29. 21:55	Egy jégmező és annak függvényeinek megvalósítva.
Iglu.java	1670	2020.03.29. 21:55	Az iglu megvalósítása.
Item.java	266	2020.03.29. 21:55	Az eszközök interfésze.
Light.java	451	2020.03.29. 21:55	A jelzőpisztoly harmadik alkatrésze.
Main.java	4171	2020.03.29. 21:55	A program fő osztálya, ahol a menü került megvalósításra.
marketableMembers.java	1484	2020.03.29. 21:55	Az IceTable és Character absztrakt őse. Eszközkezelést valósít meg.
Reserchер.java	1276	2020.03.29. 21:55	A játékban szereplő egyik karaktertípust, a sarkkutató és annak speciális képességének megvalósítása.
Rope.java	1307	2020.03.29. 21:55	A kötél megvalósítása.
Shovel.java	1856	2020.03.29. 21:55	Az ásó megvalósítása.
Stencil.java	926	2020.03.29. 21:55	A patron megvalósítása.
Surprise.java	838	2020.03.29. 21:55	A meglepetés megvalósítsa.
Swimsuit.java	967	2020.03.29. 21:55	Az úszóruha megvalósítása.

6.1.2 Fordítás

Az IceField.bat fájl tartalmazza a fordításhoz szükséges lépést (így elég azt futtatni és meg is jelenik a program a parancssorban), ami a "javac Main.java" parancs parancssorba való beírása. Ekkora létrejönnek a "java" paranccsal futtatható fájlok.

6.1.3 Futtatás

Miután az előző pontban használt parancsot lefuttattuk, ki kell adni a "java Main" parancsot és már el is indul a program a parancssorban.
Ezeket a lépéseket egyszerűsíti nekünk le az IceField.bat fájl, amire duplán kattintva egyszerűen megnyílik a parancssorban az alkalmazás. Lehet, hogy a bat fájl indításánál először a Windows Defender jelzést dob, hogy nem megbízható fájl, itt nyomjunk a "Futtatás mindenképpen" lehetőségre.

Megjegyzés: Az előző lépések és a bat fájl a cloud-on létrehozott virtuális gépen lett tesztelve és létrehozva, azokkal a fájlokkal amik véglegesen is beadásra kerültek.

6.2 Értékelés

Tag neve	Tag neptun	Munka százalékban
Csoma Zoltán	G0IOFT	20%
Sipula László Márk	A1D4QD	20%
Szepesi-Nagy István	K45SFS	20%
Varga Zsombor	AKCJOP	20%
Ziaja Bálint	HICYBO	20%

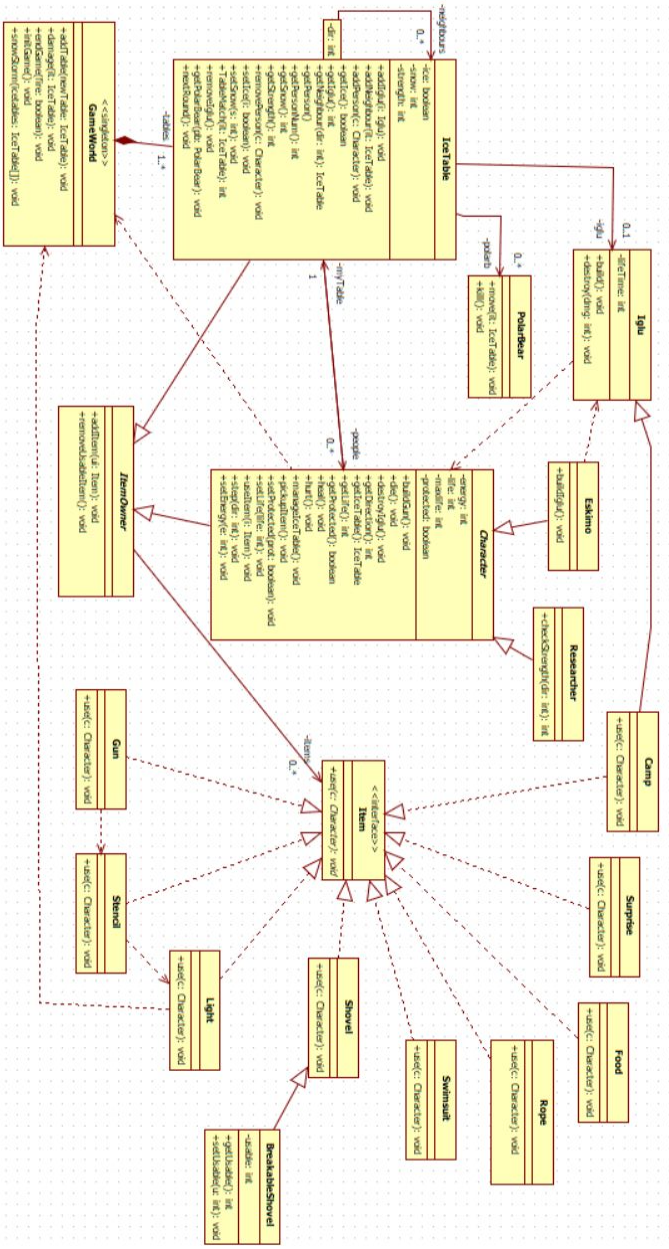
6.3 Napló

Kezdet	Időtartam	Résztvevők	Leírás
2020.03.25	2,5 óra	Ziaja Sipula Varga Csoma Szepesi-Nagy	Alapok átbeszélése, feladatok kiosztása.
2020.03.26	2 óra	Szepesi-Nagy Ziaja	5.4-es kommunikációs diagramok javítása.
2020.03.27	4 óra	Varga	Szkeleton elkészítése.
2020.03.27	1 óra	Sipula	Dokumentáció és a bat fájl elkészítése.

7. Prototípus koncepciója

7.0 Változás hatása a modellre

7.0.1 Módosult osztálydiagram



7.0.2 Új vagy megváltozó metódusok

7.0.2.1 PolarBear

- **Felelősség:** Ez az osztály felelős a jegesmedve mozgásáért és a karakterek megtámadásáért.

- **Metódusok:**
  - **move(it: IceTable): void -** Adott jégtáblára átlép a jegesmedve, ahol egyből le is kérdezi, hogy van-e karakter rajta, mert ha igen, akkor őket megeszi, feltéve, ha nem igluban vannak.
  - **kill(): void -** A karakterek megtámadását levezénylő függvény.

7.0.2.2 BreakableShovel

- **Felelősség:** A törékeny ásó osztály, egy olyan ásó működéséért felelős, ami három használat után eltörik, használhatatlan lesz.

- **Attribútumok:**
  - **usable: int -** Az eddigi használatok száma.

- **Metódusok:**
  - **use(c: Character): void -** Ezt örökli a Shovel osztálytól.
  - **setUsable(u: int): void -** Setter az usable attribútumhoz.
  - **getUsable(): int -** Getter az usable attribútumhoz.

- **Ősosztályok:**
  - Shovel

7.0.2.3 Camp

- **Felelősség:** A sátor osztály felelős a játékosok védelméért, hóvihar esetén.

- **Metódusok:**
  - **use(c: Character): void -** Sátor megépítését levezénylő függvény a c paraméterként kapott karakter aktuális jégtáblájára épül.

- **Ősosztályok:**
  - Iglu

7.0.2.4 Character (csak módosult)

- **Metódusok:**
  - **setEnergy(e: int): void -** Setter függvény az energy attribútumhoz.

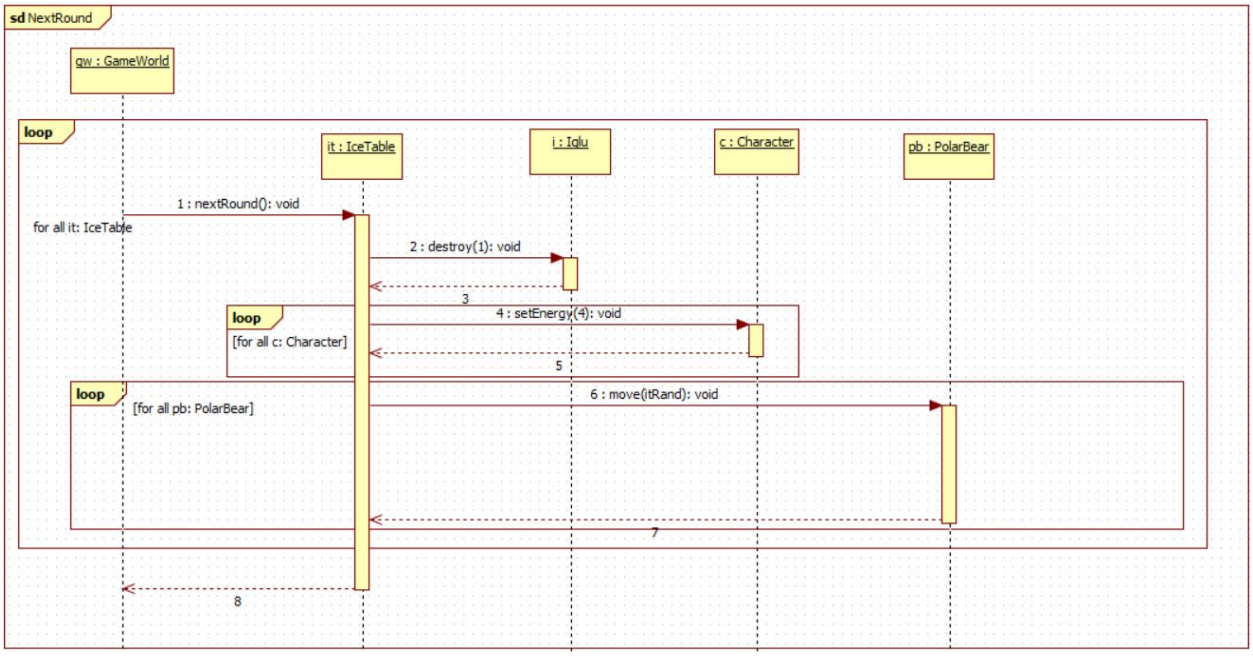
7.0.2.5 IceTable (csak módosult)

- **Attribútumok:**
  - **polarb: PolarBear -** Az eddigi használatok száma.

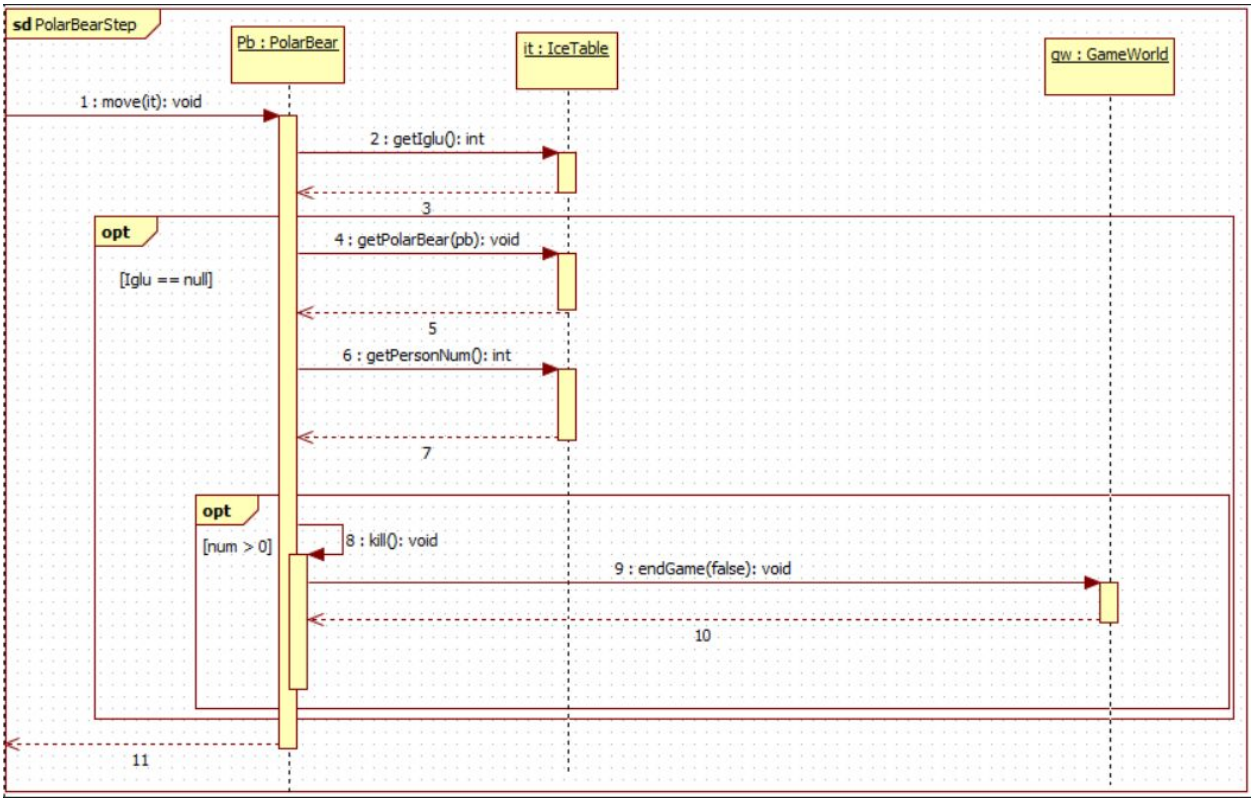
- **Metódusok:**
  - **getPolarBear(pb: PolarBear): void -** Egy jegesmedve átvétele, a polarb attribútumhoz fel kell venni.
  - **nextRound(): void -** Új kört megvalósító függvény, karakterek energiájának növelése, jegesmedvék továbbléptetése, igluk/sátrak “öregítése”.

7.0.3 Szekvencia-diagramok

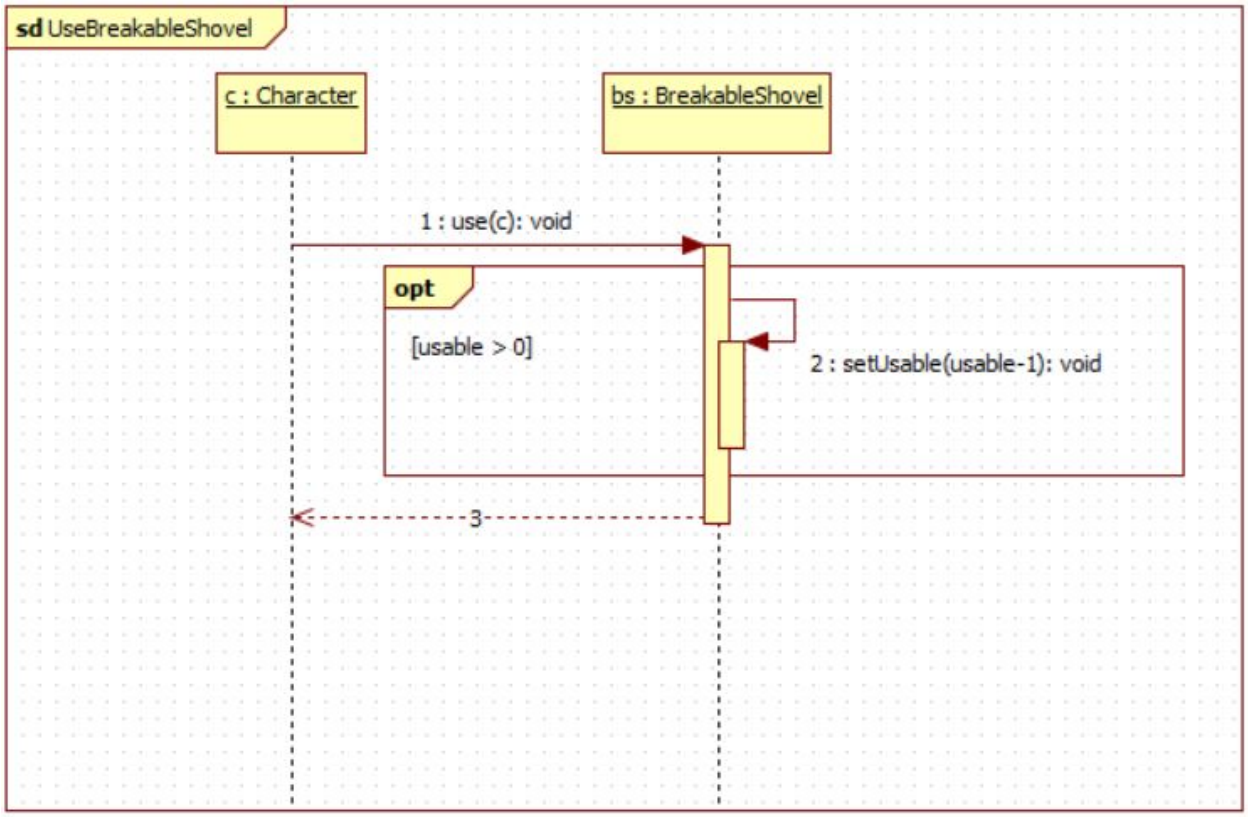
7.0.3.1 Next Round



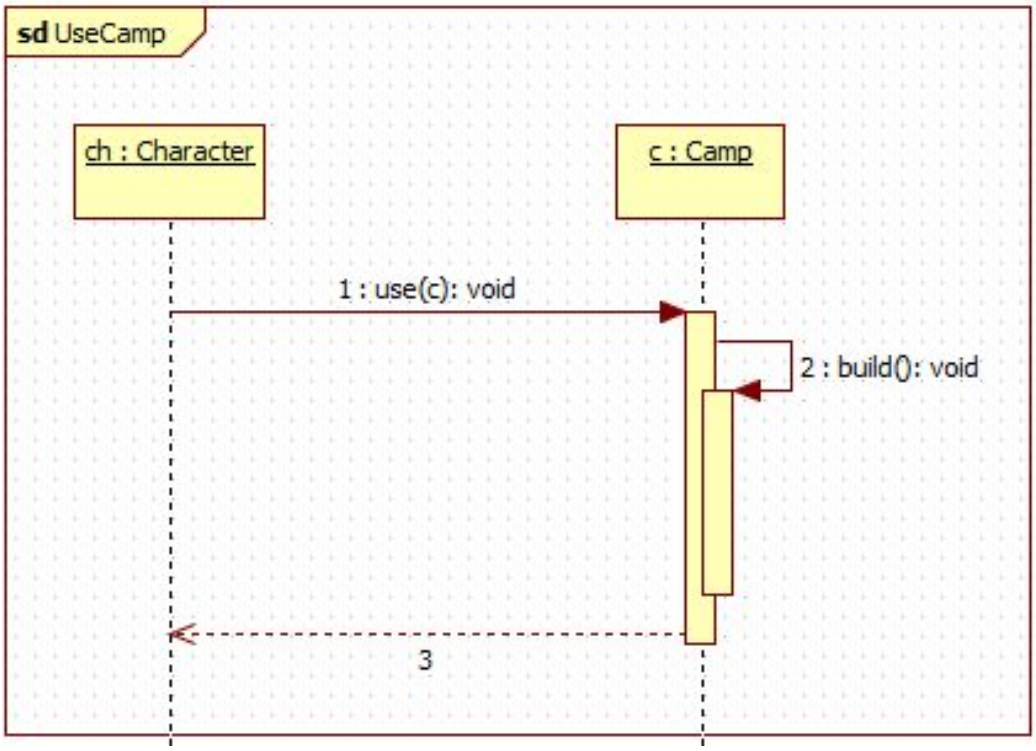
7.0.3.2 PolarBearStep



7.0.3.3 Use Breakable Shovel



7.0.3.4 Use Camp



7.1 Prototípus interface-definíciója

7.1.1 Az interfész általános leírása

A prototípus parancssoros kezelőfelülettel fog működni. A felhasználó parancsokat adhat a program számára, amelyek hatására a játéktéren különböző műveletek hajtódnak végre. A parancsok különböző opciókkal rendelkeznek, amelyek segítségével pontosítható az elvárt működés. A parancsokat és a feltételeket szóközök választják el egymástól. A parancsok listáját a 7.1.2-es pont tartlamazza. Ha bármely parancs feltétele hibás, azt jelezzük a felhasználónak a kimeneten.

7.1.2 Bemeneti nyelv

`start <num>[, <size>]`

Leírás: A pálya létrehozását előidéző parancs. A ‘num’ paraméter segítségével lehet megadni a játékosok számát. A program ennek beolvasása után létrehoz egy pályát, játékosokkal, jégtáblákkal és tárgyakkal. Ha a ‘num’ paraméter nulla, akkor a ‘size’ paraméter megadása lehetséges és kötelező is, ilyenkor egy üres pálya jön létre a beolvasott mérettel, rajta semmilyen más objektummal és minden tábla stabil lesz.

Opciók: ‘num’ - [0; ∞)
‘size’ - ha ‘num’ = 0, akkor - [1; ∞)

`step <dir>`

Leírás: A soron lévő karakterrel lépés a ‘dir’ paraméterrel átadott irányba, ahol lehet stabil, instabil vagy lyukas jégtábla is, ez a pálya építésekor véletlenszerűen létrehozott jégtábláktól függ vagy a manuálisan beállítható.

Opciók: ‘dir’ - [0; 6.283185]

`use <item>[, <character>]`

Leírás: A paraméterként megadott tárgy használata, ha kötelet (‘Rope’) használunk, meg kell adni a megmentendő karaktert is a ‘character’ paraméter segítségével. A buvárruha (‘Swimsuit’) használata felvétel után egyből ajánlott, hiszen lyukra lépés esetén csak így menekülhet meg a karakter. A fegyver (‘Gun’) használata, a játék győzelméhez vezethet, ha teljesül a fegyver elsütéséhez minden követelmény.

Opciók: ‘item’ - {Camp; Surprise; Food; Rope; Swimsuit; Shovel; Gun}
‘character’ - karakter pontos neve

`buildIglu`

Leírás: Egy iglu megépítése, Eszkimó karakterrel lehetséges csak. Az iglu az adott táblára épül, az építő karakter pedig belekerül azonnal, a táblán állókkal együtt.

`destroyIglu`

Leírás: A karakter bontja a jégtábláján lévő iglut.

`checkNeighbour <dir>`

Leírás: A sarkkutató karakter megvizsgálja a paraméterként kapott irányban lévő mezőt.
Opciók: ‘dir’ - [0; 6.283185]

`pickUpItem`

Leírás: Az aktuális jégtábláról felvesz egy tárgyat.

`getSnowStorm [<iceTable>[, <iceTable2>[, …]]]`

Leírás: A játék pillanatnyi állásában egy hóvihart generál, ami véletlenszerűen érinti az egyes jégtáblákat. A véletlenszerűséget elkerülve, explicit is megadhatjuk a hóviharral érinteni kívánt jégtáblákat.
Opciók: ‘iceTable’ - Érintett jégtábla megadása.

`removeSnow`

Leírás: A karakter lapát segítsége nélkül tisztít jégtáblát.

`info [<type>]`

Leírás: A megjelenítést kiküszöbölő parancs. Segítségével bármely objektumról lekérhető az az információ, ami megjelenítés során rendelkezésünkre állna. A ‘type’ paraméter megadása nélkül a játék alapinformációit kapjuk meg, játékosok száma, jégtáblák száma, körök száma. Az ‘IceTables’ megadásával lekérdezhető minden jégtábla azonosítója. A ‘Characters’ segítségével a játékosok azonosítója tudható meg, míg ha egy konkrét azonosítót adunk meg, részletes információt kapunk az adott objektumról.
Opciók: ‘type’ - { IceTables; Characters; CharacterID; IceTableID }

`load <file>`

Leírás: Egy korábban mentett játékmenet betöltése.
Opciók: ‘file’ - A file elérési útja és neve.

`save <file>`

Leírás: Jelenlegi játékmenet elmentése.
Opciók: ‘file’ - A file elérési útja és neve.

`nextPlayer`

Leírás: A jelenleg soron lévő karakter végzett, a következő jön a sorban.

`exit`

Leírás: A program azonnali leállítása.

`addObject <IceTable>, <type>`

Leírás: Hozzáadhatóak a játékhoz további objektumok. Meg kell adni, hogy melyik jégtáblához, milyen típusú objektumot akarunk hozzáadni.

Opciók: ‘IceTable’ - jégtábla beazonosítása
‘type’ - {Researcher; Eskimo; Iglu; PolarBear; Camp; Surprise; Food; Rope; Swimsuit; Shovel; Gun; Stencil; Light}

`setObject <object>`

Leírás: Az adott táblára egy már létező objektum helyezhető át.

Opciók: ‘object’ - {Researcher; Eskimo; Iglu; PolarBear; Camp; Surprise; Food; Rope; Swimsuit; Shovel; Gun; Stencil, Light}

`setTable <type>[, <strength>]`

Leírás: Az adott tábla típusának beállítása, hogy az instabil, stabil vagy lyukas legyen-e. Instabil esetén meg kell adni a teherbírást.

Opciók: ‘type’ - {stable; unstable; hole}
‘strength’ - [1; x] - x = Karakterek számának háromnegyedének alsó egész része

`actualTable <IceTable>`

Leírás: A kiválasztott tábla, amit a fenti parancsok segítségével módosíthatunk, ezt a kiválasztott, módosítandó táblát lehet beállítani ezzel a paranccsal, amikre a fenti parancsok kihatnak.

Opciók: ‘IceTable’ - jégtábla beazonosítása

7.1.3 Kimeneti nyelv

bemenet: start <num>
kimenet: Icefield created with <num> characters

bemenet: start 0 <size>
kimenet: Icefield created of <size> IceTables with no Characters

bemenet: step <dir>
kimenet: <CharaterID> moves to <IceTableID>

bemenet: use <item>
kimenet: <CharacterID> used <ItemID>

bemenet: use <Gun>
kimenet: ***ha sikeres:*** You won the game!
ha sikertelen: You didn't win the game! Try again!

bemenet: buildIglu
kimenet: ***ha Eskimo hívja meg:*** An iglu created on <IcaTbaleID>
ha Researcher: You can not build an iglu!

bemenet: destroyIglu
kimenet: <CharacterID> is destroying iglu on <IceTableID>

bemenet: CheckNeighbour <dir>
kimenet: ***ha Researcher hívja meg:*** Strength of <IceTableID> is <strength>
ha Eskimo: You can not check neighbour strength!

bemenet: pickUpItem
kimenet: ***ha sikeres:*** You picked up <ItemID> from <IceTableID>
ha be van fagyva: You couldn't pick up <ItemID> from <IceTableID>
ha nincs tárgy: There's no item in <IceTableID>

bemenet: getSnowStorm
kimenet: Snow storm generated on the icefields.

bemenet: getSnowStorm <IceTableID>
kimenet: Snow storm generated on <IceTableID>

bemenet: removeSnow
kimenet: IceTable has cleaned.

bemenet: info

kimenet: Number of players: <Characters>
Number of IceTables: <IceTables>
Round number: <numOfRounds>

bemenet: info <Characters>
kimenet: ***minden character-re:*** <CharacterID>

bemenet: info <IceTables>
kimenet: ***minden IceTable-re:*** <IceTableID>

bemenet: info <CharacterID>
kimenet: Actual IceTable: <IceTableID>
Characters item: <ItemID>
Life: <life>
Type: <type>

bemenet: info <IceTableID>
kimenet: Number of characters: <n>
Type: <stable, unstable, hole>
Strength: <strength>
Iglu: <Built/No>

bemenet: load <file>
kimenet: File has loaded

bemenet: save <file>
kimenet: File has saved.

bemenet: nextPlayer
kimenet: It is your turn: <CharacterID>

bemenet: Exit
kimenet: Bye.

bemenet: addObject <IceTable>, <type>
kimenet: ***ha karakter:*** <IceTableID> got a <CharacterID>
ha eszköz: <IceTable> got an <ItemID>

bemenet: setObject <object>
kimenet: ***ha karakter:*** <CharacterID> moved to <IceTableID>
ha eszköz: <ItemID> was put to <IceTableID>

bemenet: setTable <type>
kimenet: The <type> of the <IceTableID> is set.

bemenet: setTable ‘unstable’, <strength>
kimenet: The Strength of IceTable <IceTableID> is set <strength>.

bemenet: actualTable <IceTable>
kimenet: The <IceTableID> got an impulse.

7.2 Összes részletes use-case

Use-case neve	StartGame
Rövid leírás	A játék indítása
Aktorok	User
Forgatókönyv	A felhasználó a `start` parancs kiadásával betölt egy random generált méretű pályát, annyi játékossal, amennyit megadunk neki.

Use-case neve	EndGame
Rövid leírás	A játék befejezése és bezárása.
Aktorok	User
Forgatókönyv	A felhasználó az `exit` parancs kiadásával kilép a programból, amiről szöveges visszajelzést is kap.

Use-case neve	SaveGame
Rövid leírás	A játék aktuális állásának mentése.
Aktorok	User
Forgatókönyv	A `save` parancs kiadásával a felhasználó elmentheti az aktuális játékállapotot.

Use-case neve	LoadGame
Rövid leírás	Játékállás betöltése.
Aktorok	User
Forgatókönyv	A felhasználó a `load` paranccsal betölt egy korábban elmentett pályaállapotot.

Use-case neve	MoveCharacter
Rövid leírás	Karakterek (Eskimo és Researcher) léptetése egy kiválasztott mezőre.
Aktorok	User
Forgatókönyv	A felhasználó a `step` parancs kiadásával egy adott irányba lépteti a karaktert.

Use-case neve	UseItem
Rövid leírás	Tárgy használata.
Aktorok	User
Forgatókönyv	A `use` parancs kiadásával használhatóak a már felvett tárgyak. Rope esetén külön meg kell adnunk, hogy kit mentünk ki.

Use-case neve	BuildIglu
Rövid leírás	Iglu építése egy táblára.
Aktorok	User
Forgatókönyv	A felhasználó a `buildIglu` paranccsal építhet egy iglut, amennyiben éppen egy Eskimo karakter aktív. A megépült igluba minden, a táblán álló játékos automatikusan beléptetésre kerül, így egy esetleges hóvihar esetén védve lesznek.

Use-case neve	DestroyIglu
Rövid leírás	A táblán lévő iglu lebontása.
Aktorok	User
Forgatókönyv	A felhasználó a `destroyIglu` parancs kiadásával le tudja bontani bármelyik karakterrel az iglut.

Use-case neve	CheckNeighbour
Rövid leírás	Szomszédos mező teherbírásának ellenőrzése.
Aktorok	User
Forgatókönyv	A felhasználónak amennyiben egy Researcher karaktere van, akkor a `checkNeighbour` paranccsal le tudja kérdezni az adott irányban lévő szomszédos mező teherbírását.

Use-case neve	PickUpItem
Rövid leírás	Már kiásott tárgy felvétele a jégtábláról.
Aktorok	User
Forgatókönyv	A `pickUpItem` parancs kiadásával a karakter felvesz egy olyan tárgyat a jégtábláról, amit nem fed hó és nincs befagyva.

Use-case neve	SnowStorm
Rövid leírás	Hóvihar generálása.
Aktorok	User
Forgatókönyv	A `getSnowStorm` parancs kiadásával a felhasználó generálhat egy hóvihart, melynek paraméterként átadhatja a jégtáblákat, amiket érint a hóvihar.

Use-case neve	RemoveSnow
Rövid leírás	Hó eltakarítása a tábláról kézzel.
Aktorok	User
Forgatókönyv	A `removeSnow` parancs megadásával a karakter kézzel takarít el havat a tábláról.

Use-case neve	GetInfo
Rövid leírás	Objektumok információnak a lekérdezése.
Aktorok	User
Forgatókönyv	A felhasználó az `info` parancs segítségével lekérdezheti az adott játék alap információt, amennyiben megad paramétert, úgy jégtáblák, karakterek adatait kaphatja meg.

Use-case neve	NextPlayer
Rövid leírás	Soron következő játékosra lépés.
Aktorok	User
Forgatókönyv	A felhasználó a `nextPlayer` paranccsal továbbadhatja a stafétát a következő játékosnak.

Use-case neve	AddObject
Rövid leírás	Objektum hozzáadása egy táblához.
Aktorok	User
Forgatókönyv	Egy jégmezőhöz különböző objektumok (karakter, tárgyak, Iglu, Jegesmedve) hozzáadása lehetséges az `addObject` parancs megadásával.

Use-case neve	SetObject
Rövid leírás	Objektum áthelyezése egy másik tábláról.
Aktorok	User
Forgatókönyv	A `setObject` parancs segítségével egy objektum helyezhető át.

Use-case neve	SetTable
Rövid leírás	Pálya strapabírásának megadása.
Aktorok	User
Forgatókönyv	A felhasználó a `setTable` paranccsal megadhatja paraméterként, hogy milyen strapabírású az adott tábla (stabil, instabil vagy lyuk). Amennyiben instabil táblát állít be, úgy a teherbírást is meg kell adnia.

Use-case neve	EditTable
Rövid leírás	A fenti parancsok segítségével módosítani kívánt tábla kiválasztása.
Aktorok	User
Forgatókönyv	Amennyiben a felhasználó módosítani szeretne a fent látott parancsokkal egy jégtáblát, úgy először az `actualTable` paranccsal meg kell adnia, hogy melyik táblát szeretné módosítani, és ezután tudja kiadni a módosító parancsokat.

7.3 Tesztelési terv

Teszt-eset neve	Pálya létrehozása
Rövid leírás	Pálya létrehozása, majd a kezdőállapot kiírása
Teszt célja	Ellenőrzi, hogy a pályát a megfelelő paraméterekkel lett-e létrehozva.

Teszt-eset neve	Lépés
Rövid leírás	Egy Stepable objektum lépése egyik mezőről a másikra.
Teszt célja	Ellenőrzi, hogy a lépés során az új jégtábla valóban tartalmazza-e az objektumot, a régi már nem, és az objektum tudja-e hogy melyiken áll.

Teszt-eset neve	Item használata..
Rövid leírás	Bármilyen Item használása
Teszt célja	Ellenőrzi, hogy egy Item használata csökkenti-e a Character energiáját.

Teszt-eset neve	Food használata..
Rövid leírás	Élet növelése a Food-dal.
Teszt célja	Ellenőrzi, hogy a Food használata után tényleg növekedett e a használója élete, ha az nem volt max életen.

Teszt-eset neve	Camp használata..
Rövid leírás	Character letesz egy Campet, ami egy körig megvédi
Teszt célja	Ellenőrzi, hogy a jégmező tartalmazza-e a sátrat, és hóvihar esetén megvédi-e a benne tartózkódó embert.

Teszt-eset neve	Rope használata..
Rövid leírás	Szomszéd lyukas mezőből kihúzunk egy csapattársat.
Teszt célja	Ellenőrzi, hogy lyukas mezőről tényleg a saját mezőnkre húztuk a csapattársat, és a csapattárs saját mezője a minékre változott, valamint a lyukas mező már tudja hogy nem “áll” rajta senki.

Teszt-eset neve	Swimsuit használata..
Rövid leírás	Egy karakter magára vesz egy Swimsuitot
Teszt célja	Ellenőrzi, hogy a Swimsuit beállítja-e a Character protect változóját true-ra

Teszt-eset neve	Shovel használata
Rövid leírás	Hó lapátolása/Jég törése
Teszt célja	Ellenőrzi, hogy lapát használatára tényleg annyi havat lapátol, amennyit kell neki, vagy ha nem tud lapátolni akkor betöri-e a jeget.

Teszt-eset neve	Gun használata..
Rövid leírás	A Gun teljes megépítésével nyerhető meg a játék.
Teszt célja	Ellenőrzi, hogy a Gun teljes megépítése után a játék véget ér-e WIN-nel.

Teszt-eset neve	Iglu építése..
Rövid leírás	Egy eszkimó építhet Iglut egy jégmezőre.
Teszt célja	Ellenőrzi, hogy a megépített Iglut tartalmazza-e az Eskimo jégmezője.

Teszt-eset neve	Iglu rombolása
Rövid leírás	A játékosok kézzel is pusztíthatják az Iglut, ogy kiszedjék alóla a tárgyakat.
Teszt célja	Ellenőrzi, hogy az Iglu rombolása után tényleg csökkent-e egy egységgel az Iglu élete.

Teszt-eset neve	Iglu teljes elpusztítása
Rövid leírás	A játékosok addig rombolják az iglut, amíg el nem pusztul.
Teszt célja	Ellenőrzi, hogy ha az Iglut 1 életerőn ütik meg, tényleg eltűnik-e a tábláról.

Teszt-eset neve	Strapabírás ellenőrzése
Rövid leírás	A Researcher ellenőrzi egy szomszédos mező strapabírását.
Teszt célja	Ellenőrzi, hogy tényleg a jó szomszéd tényleges strapabírását olvasta-e ki.

Teszt-eset neve	Item felvétele
Rövid leírás	Item felvétele egy jégmezőből.
Teszt célja	Ellenőrzi, hogy a felvett Item bekerül-e a Character inventoryjába/”táskájába”.

Teszt-eset neve	Hóvihar Igluval
Rövid leírás	Amikor hóvihar ér egy mezőt elpusztítja az iglut, ami megvédi a benne álló(ka)t.
Teszt célja	Ellenőrzi, hogy elpusztítja-e az Iglut, ami a mezőn van úgy, hogy a benne lévő karakter(eke)t ne sebezze meg.

Teszt-eset neve	Hóvihar iglu nélkül
Rövid leírás	Amikor hóvihar Iglu nélküli mezőt ér, megsebesíti a rajta álló(ka)t.
Teszt célja	Ellenőrzi, hogy Iglu nélküli jégtáblán a hóvihar után a rajta álló karakternek csökkentette-e az életét/testhőjét 1-gyel.

Teszt-eset neve	Hóvihar Hó
Rövid leírás	Amikor hóvihar ér egy mezőt, havat tehet rá, befagyasztja és elpusztítja az iglut, ami megvédi a benne álló(ka)t.
Teszt célja	Ellenőrzi, hogy a jégtábla a hóvihar után begfagyott-e.

Teszt-eset neve	Hóvihar Jég
Rövid leírás	Amikor hóvihar ér egy mezőt, havat tehet rá.
Teszt célja	Ellenőrzi, a jégtáblán a hóvihar után nőtt-e a random értékkel a hó mennyisége.

Teszt-eset neve	Hótakarítás
Rövid leírás	Kézzel való hótakarítás a mezőről.
Teszt célja	Ellenőrzi, hogy ha van hó a mezőn, a művelet befejeztével biztosan eggyel kevesebb lesz-e a hó mennyisége.

Teszt-eset neve	Jégtörés
Rövid leírás	Ha nincs hó a mezőn, betörjük a jegét.
Teszt célja	Ellenőrzi, hogy ha nincs hó a jégmezőn, mégis interakcióba lépünk vele, akkor betöri-e a jegét.

Teszt-eset neve	Pályaállapot
Rövid leírás	Lekérjük a pálya jelenlegi állapotát.
Teszt célja	Ellenőrzi, hogy a kimenetre a pálya valódi állapotát sikerült e kiírni.

Teszt-eset neve	Pálya betöltése
Rövid leírás	Egy mentett pálya betöltése.
Teszt célja	Ellenőrzi, hogy az elmentett, vagy generált pálya valóban úgy tölt e be, ahogy annak be kell.

Teszt-eset neve	Állás mentése
Rövid leírás	A játék szüneteltetése kilépéssel
Teszt célja	Ellenőrzi, hogy a játékállást tényleg helyesen lett-e elmentve.

Teszt-eset neve	Következő játékos
Rövid leírás	Amikor egy játékosnak elfogy a 4 energiája, a soron következő cselekedhet.
Teszt célja	Ellenőrzi, hogy 4 energia után valóban cselekvőképtelen egy Character/Játékos, és a következő ugyanúgy 4 energiával fog-e kezdeni.

Teszt-eset neve	Objektum hozzáadása a jégtáblához
Rövid leírás	Lépéskor, vagy a pálya létrehozásakor hozzáadjuk a karaktert vagy jegesmedvét a jégtáblához, amin áll.
Teszt célja	Ellenőrzi, hogy a hozzáadás lefutása után tartalmazza-e az IceTable a hozzáadott objektumot.

Teszt-eset neve	Item hozzáadása a jégtáblához
Rövid leírás	Pályalétrehozáskor “belefagyasztjuk” az Itemet egy mezőbe.
Teszt célja	Ellenőrzi, hogy tényleg tartalmazza-e az IceTable az itemet, amit hozzáadtak.

Teszt-eset neve	Instabilra lépés
Rövid leírás	Egy Character az instabil mezőre.
Teszt célja	Ellenőrzi, hogy ha kevesebb ember van az instabil mezőn, mint a a kapacitása, nem szakad-e be.

Teszt-eset neve	Instabil beszakadása
Rövid leírás	Amikor több játékos van egy mezőn, mint amennyit elbír, a mező beszakad és a játék véget ér
Teszt célja	Ellenőrzi, hogy ha a rajta álló játékosok száma meghaladja a mező strapabírását, a játék LOSE-zal véget ér-e.

Teszt-eset neve	Lyukra lépés
Rövid leírás	Ha egy játékos lyukra lép, és nem mentik ki, meghal.
Teszt célja	Ellenőrzi elsősorban, hogy a játékos rendelkezik e búvárruhával. Ezután ellenőrzi, hogy van e játékos a közelében, aki kimentheti, majd ellenőrzi, hogy a következő körben tényleg véget ér-e a játék LOSE-zal.

Teszt-eset neve	Karakter megfagyása
Rövid leírás	Ha elfogy a karakter élete/testhője meghal.
Teszt célja	Ellenőrzi, hogy ha egy karakter 1 élettel védtelenül hóviharba kerül, a játék véget ér-e LOSE-zal.

Teszt-eset neve	Jegesmedve ölése
Rövid leírás	A jegesmedve megöli a vele egy mezőn álló Charactert.
Teszt célja	Ellenőrzi, hogy ha a mezőn, amire lépett a jegesmedve áll legalább egy Character, akkor a játék LOSE-zal véget ér-e.

Teszt-eset neve	Játék befejezése
Rövid leírás	A fegyver összeszerelése és elsütése
Teszt célja	Ellenőrzi, hogy véget ér-e a játék, ráadásul WIN-nel.

Teszt-eset neve	Meglepetés használata
Rövid leírás	Véletlenszerű pozitív vagy negatív hatás egy Characterre.
Teszt célja	Ellenőrizzük, hogy a véletlen esemény tényleg olyan kimenetelű-e, mint ahogy a random választó szerette volna.

Teszt-eset neve	Törékeny ásó használata
Rövid leírás	A törékeny ásó 3 használat után eltörik, és használhatatlan lesz.
Teszt célja	Ellenőrzi, hogy az ásó háromszori használhat után használhatatlan lesz-e, a hómennyiség ugyanannyi marad-e.

Teszt-eset neve	Sátor elpusztulása
Rövid leírás	A sátor csak egy körig életképes.
Teszt célja	Ellenőrzi, hogy a kör végén a sátor elpusztul-e.

7.4 Tesztelést támogató segéd- és fordítóprogramok specifikálása

A teszteléshez segédprogram használata nem lesz szükséges. A tesztesetekhez elvárt bemenet a parancssoros interfészen, vagy fájl útvonal megadása esetén fájlból betöltve is megadható. A kimenet szintén kerülhet a szöveges kimenetre, vagy fájlba.

7.5 Napló

Kezdet	Időtartam	Résztvevők	Leírás
2020.04.01 18:00	2 óra	Csoma Sipula Szepesi-Nagy Varga Ziaja	Újítások átbeszélése, megtervezése.
2020.04.01 20:00	2 óra	Varga	Osztálydiagram és meglévő szekvencia diagramok átalakítása, új szekvencia diagram megírása.
2020.04.03 14:00	1 óra	Csoma Sipula Szepesi-Nagy Varga Ziaja	Hátralévő módosítások átbeszélése, eddigiek kiegészítése, feladatok felosztása.
2020.04.04 10:00	2 óra	Csoma Sipula Szepesi-Nagy Varga Ziaja	Csoma és Szepesi-Nagy: Kimeneti nyelv megírása Sipula: Use-casek megírása Varga: Tesztesetek megírása Ziaja: Bemeneti nyelv megírása.
2020.04.05 12:00	0.5 óra	Sipula	Dokumentum véglegesítése

8. Részletes tervek

8.1. Osztályok és metódusok tervei.

8.1.1. Iglu

- **Felelősség**

Hátralévő élet számontartása, benne álló karakter(ek) megvédése hóvihar esetén. Egy kör elteltével a hátralévő ideje az iglunak csökken, nullát elérve lebomlik teljesen. A karakterek megvédése passzív módszerrel történik, azaz nem egyesével megvédi a karaktereket, hanem a hóvihar egyáltalán nem tudja a karaktereket bántani, ha a jégtáblán épült iglu.

- **Ősosztályok**

  -

- **Interfészek**

  -

- **Attribútumok**
  - **-lifeTime: int:** Az iglu hátralévő idejének számontartására használja.
  - **-myTable: IceTable:** A jégtábla tárolása, a lebomláskor és épüléskor ezen keresztül éri el a jégtábláját.

- **Metódusok**
  - **+void build()**: Az iglu megépülése. A myTable-n keresztül eléri a jégtáblát, ezen meghívja a void addIglu(this)-t, aminek paraméterébe magát adja át.

  - **+void destroy(int dmg)**: Az iglu lebontása. A paraméter értékét levonjuk a lifeTime-ból, ha így nulla lett, akkor lebomlik az iglu. A lebomlás a myTable attribútumon keresztül a void removeIglu() meghívásával történik.
    Pseudo:
    - ❑ *lifeTime értékéből kivonjuk a dmg paramétert*
    - ❑ *IF lifeTime így 0 vagy kisebb*
      - ❑ *myTable 'removeIglu' függvény hívása.*
    - ❑ *ENDIF*

  - **+IceTable getMyTable()**: Getter függvény, amely visszatér a tárolt jelenlegi jégtáblával.

  - **+void setMyTable(IceTable it)**: Setter függvény, a myTable attribútumot beállítja a paraméterként kapott táblára.

8.1.2. PolarBear

- **Felelősség**

Ez az osztály felelős a jegesmedve mozgásáért és a karakterek megtámadásáért. A jegesmedve minden körben egy másik táblára lép, a lépés levezénylése és az új táblán álló karakterek megtámadása a feladata. Lyukra lépve vagy jégtábla átfordulása esetén meghal, ezt is ez az osztály intézi.

- **Ősosztályok**
  - **Stepable**
- **Interfészek**
  -
- **Attribútumok**
  -
- **Metódusok**
  - **+void kill()**: A jegesmedve megtámadja a jégtáblán álló összes karaktert, ha talál karaktert, vége a játéknak. Az őse getMyTable() függvényén keresztül eléri a tábláját, a Stepable[ ] getPersons() segítségével eléri a rajta álló karaktereket, amiknek a void die() függvényével véghezvitte a támadást.
    Pseudo:
    - ❑ *Saját tábla megszerzése, örökölt getMyTable() fügvénnyel*
    - ❑ *Ezen lévő léptethető objektumok megszerzése, getPersons() fügvénnyel*
    - ❑ *Ezen objektumokon 'die' függvény meghívása*
  - **+void die():** A jegesmedve meghal. A saját jégtábláját eléri az ős függvényeinek segítségével, ezen void removePerson(this), itt paraméterben önmagát adja át.
  - **+void step(double dir)**: Az adott irányba történő lépés. Az ős step függvény meghívása, ezután a saját void kill() függvényének meghívásával támad.
  - **+void nextRound()**: Egy új kör következik, ekkor léptetjük magunkat a saját void step(double dir) függvény segítségével, a paraméter itt egy random szám [0; 6.283].

8.1.3. Eskimo

- **Felelősség**

Ez az osztály felelős az igluk megépítéséért, ezen kívül örökli a karakter összes metódusát.

- **Ősosztályok**
  - **Stepable → Character** → Eskimo
- **Interfészek**

  -
- **Attribútumok**

  -
- **Metódusok**
  - **+void buildIglu()**: Létrehoz egy iglut az eszkimó aktuális jégtábláján. Az örökölt getter függvények segítségével eléri a saját jégtábláját. Létrehoz egy Iglu típusú objektumot a jégtábla azonnali beállítását (setMyTable(it)) követően a void build() függvényt meghívja, aminek segítségével végleg megépül az iglu.
    Pseudo:
    - ❑ *Egy új Iglu objektum létrehozása.*
    - ❑ *Megszerezni a saját jégtáblát, örökölt 'getMyTable' függvénnyel.*
    - ❑ *Létrehozott Iglu objektum 'setMyTable' függvény hívása, ezen jégtáblával.*
    - ❑ *Iglu 'build' függvényének hívása.*

8.1.4. Researcher

- **Felelősség**

Ez az osztály felelős szomszédos jégtáblák teherbírásának kutatásáért, ezen kívül örökli a karakter összes metódusát.

- **Ősosztályok**
  - **Stepable → Character** → Researcher
- **Interfészek**

  -
- **Attribútumok**

  -
- **Metódusok**
  - **+int checkStrength(double dir)**: Lekérdezi az adott irányban található jégtáblának az teherbírását. Az örökölt getter függvények segítségével eléri a saját jégtábláját. A jégtábláján a megfelelő függvény (IceTable getNeighbour(dir)) függvény és a paraméterként kapott irány segítségével lekérdezi a mérni kívánt jégtáblát, amin már csak a megfelelő (int getStrength()) függvényt kell meghívnia a teherbírás kikutatása érdekében.
    Pseudo:
    - ❑ *Jégtábla megszerzése, örökölt 'getMyTable' függvény segítségével.*
    - ❑ *Saját tábla 'getNeighbour' függvény hívása, kapott dir paraméterrel.*
    - ❑ *A kapott táblán 'getStrength' függvény hívása.*
    - ❑ *Visszatérés ezen értékkel.*

8.1.5. Stepable

- **Felelősség**

Ez az ősosztálya mindazon osztályoknak, akik jégtáblákon keresztül tudnak lépkedni.

- **Ősosztályok**
  -
- **Interfészek**
  -
- **Attribútumok**
  - **-myTable: IceTable:** Az jelenlegi jégtábla tárolása.
- **Metódusok**
  - **+IceTable getMyTable()**: Getter függvény, amely visszatér a tárolt jelenlegi jégtáblával.
  - **+void setMyTable(IceTable it)**: Setter függvény, amely beállítja a jelenlegi jégtáblát a paraméterként kapott jégtáblára. Ennek lefutása során az aktuális jégtábla megfelelő függvénye segítségével (void removePerson(this)) el kell távolítani az objektumot és a paraméterként kapott új jégtábla megfelelő függvénye (void addPerson(this)) segítségével szólni kell, hogy már azon a jégtáblán áll az adott objektum.
    Pseudo:
    - ❏ *myTable 'removePerson' függvényének hívása aktuális objektum átadásával.*
    - ❏ *myTable átállítása a paraméterként kapott it-re.*
    - ❏ *Az új myTable 'addPerson' függvény hívása, aktuális objektum átadásával.*
  - **+void step(double dir)**: Az adott irányba történő lépés. A jelenlegi táblán a (IceTable getNeighbour(dir)) függvény meghívása, majd a visszatérési értéket paraméterként a saját setMyTable(it) függvényének kell átadni.
  - **+void die()**: Megsemmisül az adott objektum, örökölt osztályok definiálják.
  - **+void nextRound()**: Egy új kör következik, örökölt osztályok definiálják.

8.1.6. ItemOwner

- **Felelősség**

  Tárgyak felvételéért és lerakásáért felelős interfész.

- **Ősosztályok**

  -

- **Interfészek**

  -

- **Attribútumok**

  -

- **Metódusok**
  - **+void addItem(Item ui):** Character esetén a paraméterként kapott Item-et hozzáadja a Character Itemeit tartalmazó listához, IceTable esetén pedig beállítja az Item változóhoz a paramétert.
  - **+ void removeUsableItem():** Character esetén kiveszi a lista utolsó Item elemét, IceTable esetén pedig a változó értéke NULL lesz.

8.1.7. Item

- **Felelősség**

Ez az osztály a játékban található különböző használható tárgyak interfésze.

- **Ősosztályok**

  -

- **Interfészek**

  -

- **Attribútumok**

  -

- **Metódusok**
  - **+void use(c: Character):** Egy tárgy használatát megvalósító függvény. Minden tárgynál felülírásra kerül.

8.1.8. GameWorld

- **Felelősség**

  A játék felépítése, jégtáblák tárolása, hóviharok keltése, egy kör leteltének következményeinek kezelése, és a játéknak véget vetni.

- **Ősosztályok**

  -

- **Interfészek**

  -

- **Attribútumok**
  - tables: IceTable[]

- **Metódusok**
  - **+ void addTable(IceTable newTable):** Új mező hozzáadásáért felelős. Az új IceTable-t elmentjük a tables Objektum tömbben.
  - **+ void damage(IceTable it):** Adott mezőben kárt okoz, azaz a rajta lévő iglut megsemmisíti vagy karakterek életét csökkenti. Szóval meghívja az IceTable removeIglu() függvényét de ha nincs a jégtáblán iglu akkor a táblán tartózkodó Characterec hurt() függvényét hívja.
  - **+ void endGame(boolean fire):** Ha a fire true, akkor nyertek a játékosok, vége a játéknak. Ha false, akkor vesztettek, de ugyanúgy vége a játéknak.
  - **+ void initGame():** A játék létrehozásáért felelős függvény. Létrehozza a jégtáblákat és eltárolja egy listában. Annyi jégtáblát hoz létre, amennyit egy előre meghatározott szorzószámmal előállít a játékosok számából. Pl.: játékos szám * 15 darab jégtáblát hoz létre.
    Ciklusban végighaladva megadjuk a jégtáblák típusait random és ha a jégtábla éppen nem lyuk akkor random havat (pl.:0-2 közötti mennyiség) teszünk rá és befagyasztjuk (IceTable ice attribútuma). Véletlenszerűen hozzáadunk a táblákhoz Itemeket, PolarBeareket és Charactereket. Majd szintén véletlenszerűen meghatározzuk a jégtáblák szomszédosságait.
    Egy előre meghatározott max szomszéd számot használva (bekérve inputon), kitöltjük a jégtáblák szomszédossági mátrixát random mennyiségű szomszéddal. Pl.: ha max szomszéd szám 6 akkor egy jégtáblának [2-6] szomszédja lehet. Majd a jégtábla tömbön végighaladva egymáshoz rendeljük a szomszédokat.
    Pseudo:
    FOR(player * 15)
    addTable(IceTable it)
    FOR(végigmegyünk a tömbön)
    Megadjuk a jégtáblák RAND típusát (stable, unstable, hole)

```
            IF != hole
                    it.setIce()
            setSnow(RANDOM(0-2))
            it.addPerson(Stepable s) //Random
            IF!= hole
                    it.addItem(Item i)
    FOR(végigmegyünk a szomszédossági tömbön)
            kitöltjük random (pl.: 2-6 szomszéd)
    FOR(végigmegyünk a jégtábla tömbön)
            it.addNeighbour(aki a szomszédossági mátrixban hozzá tartozik)
```

- **+ void snowStrom(IceTable[ ] icetables):** Hóvihart kelt a mezőkön**.** Végighaladunk az icetables elemein és mindegyikre helyezünk egy random egységnyi havat. A random egység előre meghatározott érték alapján számolódik.

8.1.9. Character

- **Felelősség**

A karakterek ősosztálya. Minden itt valósul meg, amit a játékosok által irányítható karakterek általánosságban tudnak, vagy jellemzőjük. Az osztály absztrakt, virtuális függvénye nincs, az itt megvalósított függvények használatra készek.

- **Ősosztályok**
  - **Stepable** → Character
- **Interfészek**
  - **ItemOwner**
- **Attribútumok**
  - **-energy: int**: A körönként felhasználható energia mennyisége. Minden cselekedet után csökken az értéke, ha ez eléri a 0-t, a következő játékos jön.
  - **-life: int**: A játékos jelenlegi élete. Kezdetben megegyezik a max életével. Ha ez 0, a játék véget ér.
  - **-maxlife: int**: A játékos maximális élete. Ezt a maximumot még healeléssel sem lépheti túl.
  - **-protect: boolean**: A játékos búvárruhájával való védettséget jelzi. Ha ez true, a vízbe esés nem okozza a játék végét.
  - **-items: ArrayList<Item>**: A játékosnál tárolt Itemek listája. Csak innen használhat itemeket a játékos. Mérete végtelen.
- **Metódusok**
  - **+void nextRound()**: Körönként meghívandó függvény, beállítja az energyt újra 4-re. Ha egy játékos a vízben van a kör kezdetén, a játék véget ér.
  - **+void heal()**: A játékos testhőjének/életének növelése. A Food (és a Surprise) használata tudja kiváltani az eseményt. A játékos testhője/élete csak akkor nőhet, ha az élete nem a karakteréhez képesti maximum. Egyébként nem csinál semmit.
  - **+void hurt()**: Egy játékos testhőjének elvesztése. Ha hóvihar van, és nem védi a játékost iglu, akkor csökken eggyel a testhője. Ha a testője 0-ra csökken, meghal, és a játék végetér.
  - **+void destroyIglu()**: A játékosok lekérik a saját jégtáblájuk igluját és csökkenthetik az iglu élettartalmát eggyel az iglu destroy(1) meghívásával, mert addig nem áshatnak ki Itemet alóla, amíg rajta van. Egy energiába kerül.
  - **+void buildGun()**: A játék végéhez össze kell szerelni a pisztolyt. Ennek a függvénynek a meghívásával kezdhetik meg a játék megnyeréséhez tartozó folyamatot.
    Ellenőrzi hogy minden játékos egy táblán áll-e. ha igen: Elkezdi használni a Gun-t, ha nem, nem csinál semmit. Egy energiába kerül összesen,
    Pszeudo:
    -Elkérjük a jégtáblán álló emberek számát

-Ha ez a maxjátékos számával egyenlő
Meghívja a jégtábla azon függvényét, ami ellenőrzi, hogy valakinél megvan-e a Gun (checkItem(Gun))
Ha megvan
Meghívja a use-át

- **+void manageIceTable()**: A jégtábla takarításához szükséges függvény. Ha a jégtáblán van hó, eggyel csökkenti az értékét, ha nincs, de van jég, akkor betöri, ha nincs se hó se jég akkor pedig nem csinál semmit. Egy energiába kerül.
- **+int getLife()**: Getter függvény az életre. Visszaadja a játékos jelenlegi életét. Ha ez 0, a játék véget ér.
- **+void setLife(int l)**: Setter függvény az életre. A paraméterül kapott értéket beállítja az új életnek.
- **+bool getProtect()**: Getter függvény a protect változóra. Visszaadja hogy a játékos védve van-e a víztől.
- **+void setProtect(bool p)**: Setter függvény a protectre. Truera állítja, hogy a játékos használ egy fürdőruhát.
- **+void pickupItem()**: Megkísérli felvenni az itemet a jégtábláról. Sikerül, ha a jégtáblában van Item és a jégtáblán nincs se hó se jég. Ezeket az adatokat előtte el kell kérni a jégtáblától. Egy energiába kerül.
- **+void useItem(Item i)**: Használja a paraméterül kapott Itemet. A paraméterül kapott Itemnek a saját tárolójából kell kikerülnie, nem használhat olyat, ami nincs nála. Egy energiába kerül.
- **+void addItem(Item i)**: A paraméterül kapott Itemet hozzáadja a saját tárolójához.
- **+void removeItem(Item i)**: Eltávolít egy paraméterül kapott itemet a tárolójából. A kapott itemet kikeressük a listájából, majd kitöröljük onnan, ha már nem használhatjuk többet. Ilyen item például a Food.
- **+void setEnergy(int e)**: Setter függvény, beállítja a játékos energiaszintjét minden cselekedet után vagy kör elején a kapott értékkel.
- **+int getEnergy()**: Getter függvény, visszaadja a játékos hátralévő energiáját.
- **+void die()**: Egy játékos halálakor hívódó függvény. Lényegtelen, hogy a halált jegesmedve, fulladás vagy kihűlés okozta, értesíti a játékvilágot hogy a játék befejeződött, vereséggel véget ért.

8.1.10. IceTable

- **Felelősség**
  Egy jégtáblán lévő igluk, karakterek, szomszédok kezelése.
- **Ősosztály**
  -
- **Interfészer**

  ItemOwner

- **Attribútumok**
  - **-ice: boolean:** A jégtáblán lévő jég tárolása.
  - **-snow: int:** A jégtáblán lévő hó tárolása.
  - **-strength: int:** A jégtábla terhelhetőségének tárolása
  - **-neighbours: ArrayList<IceTable>**: A jégtábla szomszédainak tárolása a hozzáadásuk sorrendjében.
  - **-item: Item**: A jégtáblába befagyott Item.
  - **-stepables: ArrayList<Stepable>**: A jégtáblán álló játékosok vagy medvék száma.
  - **-iglu: Iglu**: A jégtáblára épített Iglu típusú objektum (Sátor vagy Iglu).
  - **gameworld: GameWorld**: A játékvilág tárolása, amiben benne vannak. A játék végének jelzése céljából van rá szükség.
- **Metódusok**
  +addIglu(i: Iglu): void: Kapott iglu tárolás sajátként.
  - **+addNeighbour(it: IceTable): void:** Az aktuális jégtábla szomszéd jégtáblájának a regisztrálása a paraméterül kapott IceTable alapján.
  - **+addPerson(s: Stepable): void:** Az adott karakter hozzáadása a mezőhöz. Ha instabil, ellenőrzi, hogy az új karakterrel nem lépték-e át a mező strapaírását. Ha lyuk, a játékos nem léphet többet. Ha lyukból nem húzzák ki egy körön belül, a játék véget ér, kivéve ha van rajta búvárruha.
    Pszeudo:
    -A kapott karaktert hozzáadjuk a stepable listánkhoz.
    -Ha a mező instabil
    Megnézzük hogy a lista mérete nagyobb-e mint a strapabírás
    Ha nagyobb a lista mérete
    Játék vége
    Ha a mező lyuk
    A játékos energiája 0ra csökken
    Ha nem húzzák ki egy kör alatt
    Játék vége
  - **+getIce(): boolean:** Getter függvény a befagyás állapotára.
  - **+getIglu(): int:** A jégtábla megkapja azt, hogy az adott mezőn van-e iglu.
  - **+getNeighbour(dir: double): IceTable:** A paraméterként kapott irány alapján visszaadja az adott irányban lévő szomszédját. Az irány radiánban értendő, a mezőhöz képesti elhelyezkedés és a szomszédok száma alapján határozza meg.

(Pl.: Ha 3 szomszédja van, akkor 2.094 radinánonként helyezkednek el, ami 120 fok. Eszerint a 3. szomszéd az 4.188 és 6.28 radián közötti terület.)

- **+getPersons(): Stepable:** Megkapja a jégtáblán lévő karaktereket.
- **+getPersonNum(): int:** Megkapja a jégtáblán lévő karakterek számát.
- **+getSnow(): int:** Getter függvény a snow változóra.
- **+getStrength(): int:** Getter függvény a strength változóra.
- **+removePerson(s: Stepable): void:** Egy játékos lépésekor már ne tárolnunk, hogy rajtunk áll, így eltávolítjuk a listánkból.
- **+setIce(i: boolean): void:** Setter függvény az ice változóra.
- **+setSnow(s: int): void:** Setter függvény a snow változóra. A paraméterül kapott értéket állítja be a snow értékének.
- **+TableMatch(it: IceTable): double:** Azzal az értékkel tér vissza, hogy a paraméterül kapott jégtábla, hol van hozzá képest.
- **+removeIglu(): void:** A jégtáblán lévő Iglu eltávolítása.
- **+nextRound(): void:** Egy új kör következik, ekkor minden Játékos nextRoundját meghívjuk, leromboljuk a sátrakat és csökkentjük a rajtuink álló Iglu élettartamát.
- **+getGameWorld(): GameWorld:** Getter függvény a gameworld attribútumra.
- **+checkItem(Item item): void**: Ellenőrzi, hogy a kapott Item megvan-e valamelyik rajta álló játékosánál. A fegyver összeszereléséhez szükséges függvény.

8.1.11. Camp

- **Felelősség**

Ez az osztály valósítja meg a sátrat a játékban. A sátor egy kiásható tárgy, mely hasonlóan viselkedik mint az Iglu, viszont csak egy karakter fér el benne, nem véd a medvétől és egy kör után megsemmisül.

- **Ősosztályok**

  -

- **Interfészek**

  Item

- **Attribútumok**

  -

- **Metódusok**
  - **+void use(c: Character):** Meghívja egy Camp példány build függvényét, mely által megépül egy sátor a karakter táblájára.

8.1.12. Surprise

- **Felelősség**

A játékban lévő meglepetés tárgyat megvalósító osztály. Ennek a megtalálásával a játékos különböző “képességet” kap.

- **Ősosztályok**

  -

- **Interfészek**

  Item

- **Attribútumok**

  -

- **Metódusok**
  - **+void use(c: Character):** A függvény hívásánál a paraméterként kapott karakterre fejt ki egy különleges hatást.

8.1.13. Food

- **Felelősség**

Ez az osztályt valósítja meg a játékban megtalálható ételt. Felhasználásával a karakter testhője növekszik eggyel.

- **Ősosztályok**

  -

- **Interfészek**

  Item

- **Attribútumok**

  -

- **Metódusok**
  - **+void use(c: Character):** Meghívja a paraméterként kapott karakter heal függvényét, ezzel annak testhőjét növeli.

8.1.14. Rope

- **Felelősség**

Ez az osztály valósítja meg a játékban a kötelet, melynek segítségével egy játékos kimentheti egy vízbeesett társát.

- **Ősosztályok**

  -

- **Interfészek**

  Item

- **Attribútumok**

  -

- **Metódusok**
  - **+void use(c: Character):** Meghívás után lekérdezi a paraméterként kapott karakter jégtábláját (*IceTable getIceTable()* függvény hívása) és lekérdezi, hogy melyik irányban(*int getDirection()* függvény hívása) található jégtáblán lévő karaktert kell kihúzni a vízből. Ezután az aktuális jégtáblától lekérdezi az előzőleg lekérdezett irányban lévő szomszédot(*IceTable getNeighbour(int dir)* függvény hívása), majd az azon lévő karaktert(*Person getPerson()* függvény hívása). Ezután kihúzandó karakter jégtáblájától lekérdezzük, hogy a kötelet használó karakter táblája milyen irányban van hozzá képest(*int TableMatch(IceTable it)* függvény hívása), majd ebbe az irányba léptetjük a kihúzandó karaktert(*void step(int dir)* függvény hívása).

PSEUDOCODE:

függvény (use)

    Lekérdezzük a paraméterként kapott karakter jégtábláját
    Lekérdezzük a jégtábla irányát, ahol a kimenekítendő van
    Lekérdezzük az ebben az irányban lévő szomszédot
    Lekérdezzük az ezen a táblán lévő karaktert
    Lekérdezzük, hogy a kötelet használó karakter táblája milyen irányban van.
    Léptetjük a kihúzandó karaktert az előbb lekért irányba.

8.1.15. Swimsuit

- **Felelősség**

A játékban megtalálható búvárruhát megvalósító osztály. Aki ezzel rendelkezik, az megmenekül a vízbeesésnél a megfulladástól.

- **Ősosztályok**

  -

- **Interfészek**

  Item

- **Attribútumok**

  -

- **Metódusok**
  - **+void use(c: Character):** Meghívja a paraméterként kapott karakter void *setProtected(boolean prot)* függvényét, mellyel a karakter boolean protected változóját igazra állítja.

8.1.16. Shovel

- **Felelősség**

A játékban szereplő lapátot megvalósító osztály. Ennek a segítségével egy játék egy energia felhasználásával két egység havat lapátolhat el.

- **Ősosztályok**

  -

- **Interfészek**

  Item

- **Attribútumok**

  -

- **Metódusok**
  - **+void use(c: Character):** A függvény lekérdezi a paraméterként kapott karakter jégtábláját(*IceTable getIceTable()* függvény), majd lekérdezésre kerül az adott jégtábla snow és ice attribútuma (*int getSnow()* és *boolean getIce()* függvények hívása). Ezután amennyiben a hó legalább két egységnyi vastag, úgy kettővel csökkentjük a jégtábla *snow* attribútumának értékét(*void setSnow()* függvény segítségével), ha csak egy egységnyi vastag, akkor egyel csökkentjük a változó értékét. Amennyiben nincs hó a jégtáblán(*snow* értéke 0) és a tábla be van fagyva(*ice* értéke *true*), akkor feltörjük a jeget(*ice* értékét *false*-ra állítjuk a *void setIce(boolean ice)* függvény segítségével).

PSEUDO:

```
függvény (use)
        Lekérdezzük a karakter jégtábláját
        Lekérdezzük a jégtábla snow attribútumát
        Lekérdezzük a jégtábla ice attribútumát
        HA snow >= 2
                AKKOR
                        csökkentjük a snow értékét kettővel
        HA snow == 1
                AKKOR
                        csökkentjük a snow értékét eggyel
        HA snow == 0 ÉS ice == true
                AKKOR
                        beállítjuk az ice attribútum értékét false-ra
```

8.1.17. BreakableShovel

- **Felelősség**

Ez az osztály felel a törékeny lapát játékban való megvalósításáról. Egy törékeny lapát hasonlóan működik, mint egy sima lapát, ugyanúgy 2 egység hó lapátolható el vele egy egységnyi energia felhasználásával, viszont három használat után eltörik és így már nem használható.

- **Ősosztályok**

  Shovel

- **Interfészek**

  -

- **Attribútumok**
  - **-usable: int - A hátralévő lapátolások számát nyilvántartó változó.**

- **Metódusok**
  - **+void use(c: Character):** Amennyiben a *usable* attribútum értéke több, mint nulla, akkor a függvény lekérdezi a paraméterként kapott karakter jégtábláját(*IceTable getIceTable()* függvény), majd lekérdezésre kerül az adott jégtábla snow és ice attribútuma (*int getSnow()* és *boolean getIce()* függvények hívása). Ezután amennyiben a hó legalább két egységnyi vastag, úgy kettővel csökkentjük a jégtábla *snow* attribútumának értékét(*void setSnow()* függvény segítségével), ha csak egy egységnyi vastag, akkor egyel csökkentjük a változó értékét. Amennyiben nincs hó a jégtáblán(*snow* értéke 0) és a tábla be van fagyva(*ice* értéke *true*), akkor feltörjük a jeget(*ice* értékét *false*-ra állítjuk a *void setIce(boolean ice)* függvény segítségével). Ezután a *usable* értékét eggyel csökkentjük.

PSEUDO:

```
függvény (use)
        HA usable != 0
                AKKOR
                        Lekérdezzük a karakter jégtábláját
                        Lekérdezzük a jégtábla snow attribútumát
                        Lekérdezzük a jégtábla ice attribútumát
                        HA snow >= 2
                                AKKOR
                                        csökkentjük a snow értékét kettővel
                        HA snow == 1
                                AKKOR
                                        csökkentjük a snow értékét eggyel
                        HA snow == 0 ÉS ice == true
                                AKKOR
                                        beállítjuk az ice attribútum értékét false-ra
                csökkentjük a usable attribútum értékét egyel
```

- **+int getUsable():** Segédfüggvény, hogy le tudjuk kérdezni a usable attribútum értékét.
- **+void setUsable():**Segédfüggvény, hogy be tudjuk állítani a usable attribútum értékét.

8.1.18. Gun

- **Felelősség**

Ez az osztály valósítja meg a pisztolyt a játékban. A pisztoly a játék megnyeréséhez szükséges a patron és a jelzőfény mellett.

- **Ősosztályok**

  -

- **Interfészek**

  Item

- **Attribútumok**

  -

- **Metódusok**
  - **+void use(c: Character):** Elkéri a paraméterként kapott karakter *Stencil* példányát és meghívja a *Stencil void use()* függvényét.

8.1.19. Stencil

- **Felelősség**

A játékban megtalálható patron megvalósításáért felelős osztály. A pisztolyba töltve a jelzőfénnyel együtt megnyerhető a játék.

- **Ősosztályok**

  -

- **Interfészek**

  Item

- **Attribútumok**

  -

- **Metódusok**
  - **+void use(c: Character):** Elkéri a paraméterként kapott karakter *Light* példányát és meghívja a *Light void use()* függvényét.

8.1.20. Light

- **Felelősség**

Ez az osztály valósítja meg a játékban lévő jelzőfényt. A pisztolyba szerelve a patronnal együtt, majd elsütve megnyerhető a játék.

- **Ősosztályok**

  -

- **Interfészek**

  Item

- **Attribútumok**

  -

- **Metódusok**
  - **+void use(c: Character):** Jelez a *GameWorld*-nek, hogy vége a játéknak az *endGame(boolean fire)* függvény meghívásával.

8.2. A tesztek részletes tervei, leírásuk a teszt nyelvén

8.2.0.1 Bemeneti nyelv változásai

- `start <num>[, <size>[, <neighbourCnt>]]`

  Leírás: A pálya létrehozását előidéző parancs. A ‘num’ paraméter segítségével lehet megadni a játékosok számát. A program ennek beolvasása után létrehoz egy pályát, játékosokkal, jégtáblákkal és tárgyakkal. Ha a ‘num’ paraméter nulla, akkor a ‘size’ paraméter megadása lehetséges és kötelező is, ilyenkor egy üres pálya jön létre a beolvasott mérettel, rajta semmilyen más objektummal és minden tábla stabil lesz. A ‘size’ paraméter megadása után, lehetséges egy szomszédok száma megadása is, amivel kikötjük, hogy egy jégtáblának hány szomszédja legyen általában.

  Opciók: ‘num’ - [0; ∞)

  ‘size’ - ha ‘num’ = 0, akkor - [1; ∞)

  ‘neighbourCnt’ - ha ‘size’ = 0, akkor [2, 9]

- `actualCharacter <character>`

  Leírás: A soron lévő karakter a paraméterként kapott legyen.

  Opciók: ‘character’- {CharacterID}

- `pickUpItem <ItemID>`
  Leírás: Az aktuális jégtábláról felvesz egy tárgyat.

  Opciók: ‘ItemID’ - A jégtáblán található tárgy id-ja.

- `addObjectCharacter <CharacterID>, <type>`
  Leírás: Egy új Item hozzárendelhető egy létező karakterhez.
  Opciók: ‘CharacterID’ - Character beazonosítása
  ‘type’ - {Camp; Surprise;
  Food; Rope; Swimsuit; Shovel; Gun; Stencil; Light}

8.2.0.2 Kimeneti nyelv változásai

- bemenet: `start 0 <size> <neighbourCnt>`

  kimenet: Icefield created of <size> IceTables with maximum number of neighbours <neighbourCnt>

- bemenet: `info <IceTableID>`

  kimenet: Number of characters: <n>
  Type: <"stable", "unstable", "hole">
  Strength: <strength>
  Iglu:<"Built"/"No Iglu">
  Characters: <CharacterID, CharacterID2…>
  Neighbours: <IceTableID, IceTableID2…>
  Item: <ItemID/"No Item">
  Iced: <"True"/"False">
  Snow: <Quantity of Snow>

- bemenet: `info <CharacterID>`

  kimenet: Actual IceTable: <IceTableID
  Characters items: <ItemID, ItemID2…>
  Life: <life>
  MaxLife: <maxLife>
  Energy: <energy>
  Type: <type>
  Swimsuit: <"True"/"False">

- bemenet: `actualCharacter <CharacterID>`

  kimenet: `<CharacterID>'s turn`

- bemenet: `setObject <object>`

  kimenet: **ha karakter:** `<CharacterID> moved to <IceTableID>`

  ha eszköz és sikeres: `<ItemID> was put to <IceTableID>`

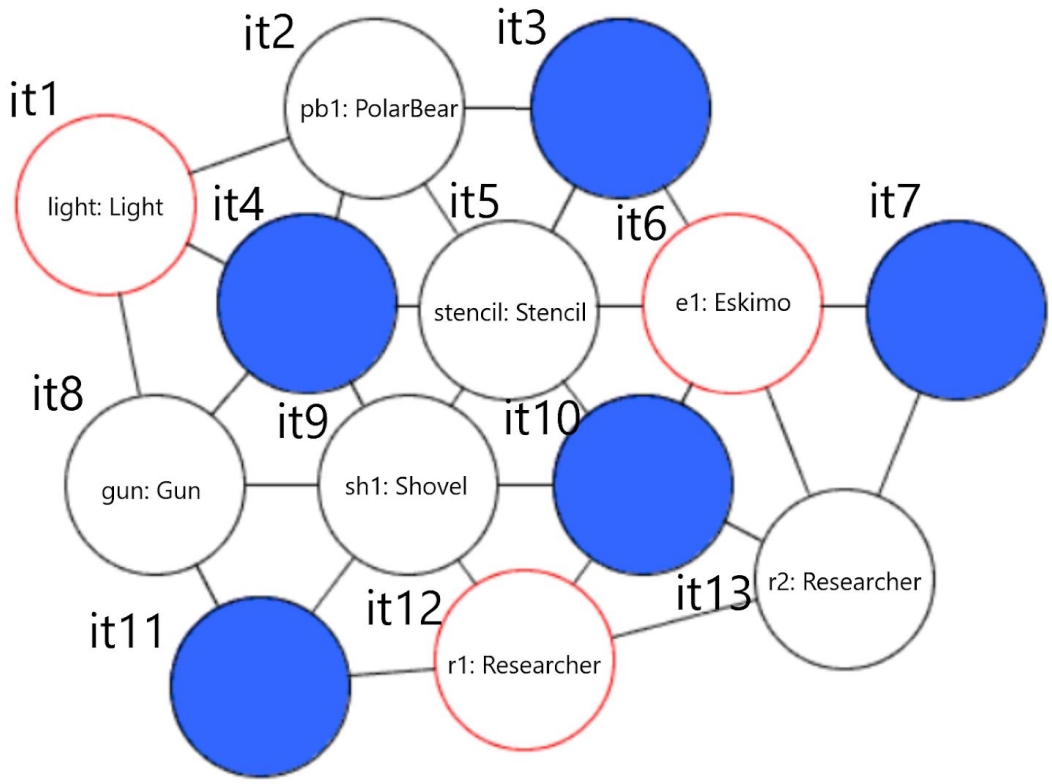
  ha eszköz és sikertelen: `<IceTableID> already owned an item`

- bemenet: addObjectCharacter <CharacterID>, <type>
  kimenet: <CharacterID> got an <ItemID>

8.2.0.3 Előre definiált pályák leírása

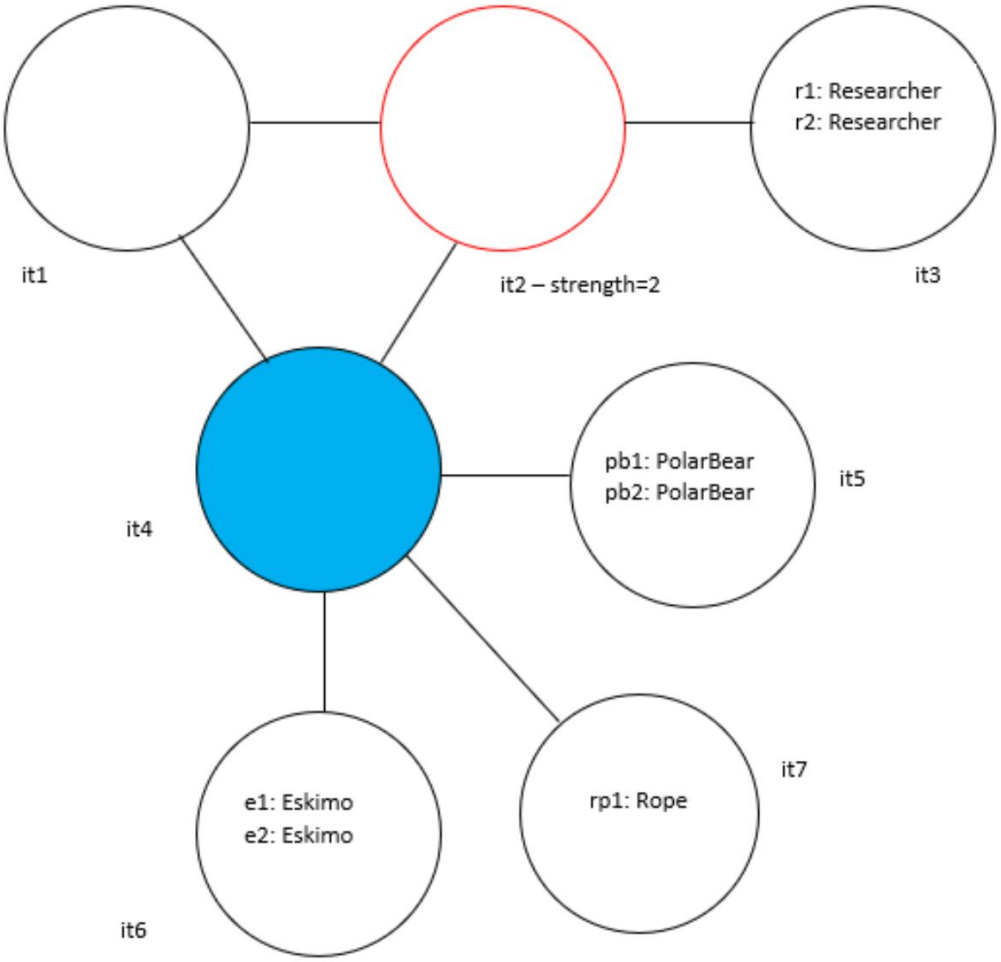
A teszteléshez létrehozunk előre definiált kisebb pályákat, melyeken könnyebb az egyes funkciók tesztelése. A pályákat a **`load`** bemeneti paranccsal fogjuk betölteni, majd a pályákon elvégezzük a teszteket. Néhány előre létrehozott pályával képesek leszünk az elvárt tesztek végrehajtására.

- *Első pálya*
  - **Leírás**: 13 jégtáblából álló pálya, amelyen egy Eskimo, kettő Researcher és egy PolarBear tartózkodik. Ezen felül egy ásó és a három jelzőrakéta-alkatrész. A jégtáblákon nincs hó.
  - **Jelölések:**
    - Fehér kör: stabil jégtábla
    - Kék kör: Lyuk
    - Fehér kör + piros szegély: instabil jégtábla
    - Jégtáblák ID-jai: it1, it2, …, it13
    - Fájl neve: `~/elsopalya`



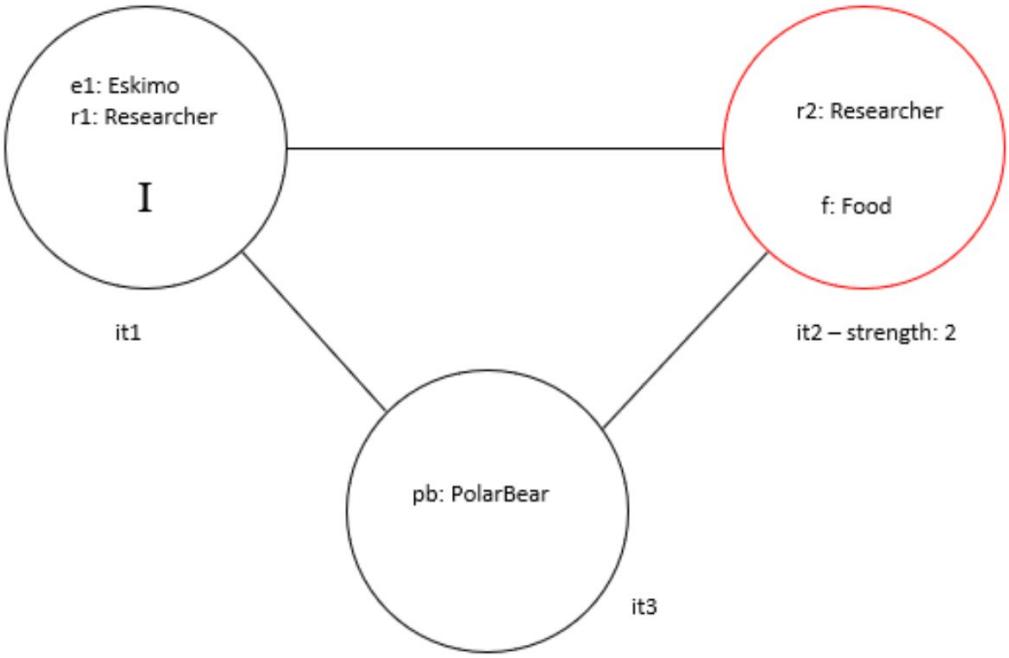
- *Második pálya*

- **Leírás**: 7 jégtáblából álló pálya, amelyen kettő Eskimo, kettő Researcher és kettő PolarBear tartózkodik. Ezen felül egy kötél található a pályán, ami nincs befagyva.
- **Jelölések:**
  - Fehér kör: stabil jégtábla
  - Kék kör: Lyuk
  - Fehér kör + piros szegély: instabil jégtábla
  - Jégtáblák ID-jai: it1, it2, ..., it7
  - Fájl neve: `~/masodikpalya`



- *Harmadik pálya*

- **Leírás**: 3 jégtáblából álló pálya, amelyen egy Eskimo, két Researcher és egy PolarBear tartózkodik. Ezen felül egy élelem és egy Iglu található a pályán. Az Igluban tartózkodik e1 Eskimo és r1 Researcher
- **Jelölések:**
  - Fehér kör: stabil jégtábla
  - Kék kör: Lyuk
  - Fehér kör + piros szegély: instabil jégtábla
  - Jégtáblák ID-jai: it1, it2, it3
  - Fájl neve: `~/harmadikpalya`



8.2.1. Pálya létrehozása

- **Leírás**
  Pálya létrehozása, azon néhány Character és Item hozzáadása
- **Ellenőrzött funkcionalitás, várható hibahelyek**
  Ellenőrzi, hogy a pálya megfelelő paraméterekkel lett-e létrehozva.
- **Bemenet**

```
start 0 10 4
addObject it1 Eskimo
addObject it2 Eskimo
addObject it3 Researcher
addObject it4 Researcher
addObject it9 PolarBear
addObject it8 Food
info
```

- Elvárt kimenet

```
Icefield created of 10 IcaTables with no Characters
it1 got a e1
it2 got a e2
it3 got a r1
it4 got a r2
it9 got a pb1
it 8 got a f1
Number of players: 4
Number of IceTables: 10
Round number: 1
```

8.2.2. Lépés

- **Leírás**
  Egy Stepable objektum lépése egyik mezőről a másikra.
- **Ellenőrzött funkcionalitás, várható hibahelyek**
  Ellenőrzi, hogy a lépés során az új jégtábla valóban tartalmazza-e a charactert, a régi jégtábla már nem és a character tudja, hogy melyiken áll.
- **Bemenet**

```
load ~/elsopalya
info it12
into r1
actualCharacter r1
step 2.1
info it12
info it9
info r1
```

- **Elvárt kimenet**

```
File has loaded

Number of characters: 1
Type: unstable
Strength: 2
Iglu: No Iglu
Characters: r1
Neighbours: it9, it10, it11, it13
Item: No item
Iced: False
Snow: 0

Actual IceTable: it12
Character items: -
Life: 4
Energy: 4
Type: Researcher
Swimsuit: False

r1’s turn
r1 moves to it9

Number of characters: 0
Type: unstable
Strength: 2
Iglu: No Iglu
Characters: r1
Neighbours: it9, it10, it11, it13
Item: No item
Iced: False
Snow: 0

Number of characters: 1
Type: unstable
Strength: 2
Iglu: No Iglu
Characters: r1
Neighbours: it4, it5, it8, it10, it11, it12
Item: sh1
Iced: False
Snow: 0

Actual IceTable: it9
Character items: -
```

```
Life: 4
Energy: 4
Type: Researcher
Swimsuit: False
```

8.2.3. Item használata

- **Leírás**
  Bármilyen Item használata. Egy karakter felvesz egy eszközt és saját funkciójának megfelelően használja.
- **Ellenőrzött funkcionalitás, várható hibahelyek**
  Ellenőrzi, hogy egy Item használata csökkenti-e a Character energiáját.
- **Bemenet**

```
load ~/elsopalya
actualTable it9
setObject r1
getSnowStorm it9
actualCharacter r1
info r1
pickUpItem sh1
use sh1
info r1
```

- **Elvárt kimenet**

```
File has loaded
The it9 got an impulse
r1 moved to it9
Snow storm generated on it9
r1's turn

Actual IceTable: it9
Character items: sh1
Life: 4
Energy: 4
Type: Researcher
Swimsuit: False

You picked up sh1 from it9
r1 used sh1

Actual IceTable: it9
Character items: sh1
Life: 4
Energy: 3
Type: Researcher
Swimsuit: False
```

8.2.4. Food használata

- **Leírás**
  Az élet növelése Food-dal character számára.
- **Ellenőrzött funkcionalitás, várható hibahelyek**
  Ellenőrzi, hogy a Food használata után tényleg növekedett-e a használója élet, ha még nem volt max élete.
- **Bemenet**

```
load ~/harmaspalya
actualCharacter r2
info r2
pickUpItem
use f
info r2
getSnowStorm it2
info r2
use f
info r2
```

- **Elvárt kimenet**

```
File has loaded
r2's turn

Actual IceTable: it2
Character items: -
Life: 4
Energy: 4
Type: Researcher
Swimsuit: False

You picked up f from it2
r2 used f

Actual IceTable: it2
Character items: f
Life: 4
Energy: 4
Type: Researcher
Swimsuit: False

Snow storm generated on it2

Actual IceTable: it2
Character items: f
Life: 3
Energy: 4
```

```
Type: Researcher
Swimsuit: False

r2 used f

Actual IceTable: it2
Character items: -
Life: 4
Energy: 4
Type: Researcher
Swimsuit: False
```

8.2.5. Camp használata

- **Leírás**
  Character letesz egy Camp-et, ami egy körig megvédi
- **Ellenőrzött funkcionalitás, várható hibahelyek**
  Ellenőrzi, hogy a Character IceTable-je tényleg tartalmaz-e egy Camp-et majd egy kör után már nem tartalmazza. Ellenőrzi, hogy tényleg megvédi-e a benne lévőt a hóviharban.
- **Bemenet**
  ```
  load ~/kettespalya
  actualTable it1
  setObject e1
  addObject it1 Camp
  actualCharacter e1
  pickUpItem c1
  use c1
  snowStorm it1
  info e1
  info it1
  nextPlayer
  info it1
  ```
- **Elvárt kimenet**
  ```
  File has loaded
  The it1 got an impulse
  e1 moved to it1
  it1 got a c1
  e1's turn
  You picked up c1
  e1 used c1
  Snow storm generated on it1

  Actual IceTable: it1
  Character items: c1
  Life: 4
  ```

```
Energy: 3
Type: Eskimo
Swimsuit: False

Number of characters: 1
Type: stable
Strength: -
Iglu: No Iglu
Characters: e1
Neighbours: it2, it4
Item: c1
Iced: False
Snow: 0

It is your turn: r1

Number of characters: 1
Type: stable
Strength: -
Iglu: No Iglu
Characters: r1
Neighbours: it2, it4
Item: No item
Iced: False
Snow: 0
```

8.2.6. Rope használata

- **Leírás**
  Szomszédos lyukas mezőből kihúzunk egy csapattársat
- **Ellenőrzött funkcionalitás, várható hibahelyek**
  Ellenőrzi, hogy lyukas mezőről tényleg a saját mezőnkre húztuk a csapattársat, és a csapattárs saját mezője a mienkre változott, valamint a lyukas mező tudja hogy nincs már rajta Character.
- **Bemenet**

```
load ~/elsopalya
actualTable it13
addObject it13 Swimsuit
setObject r1
actualCharacter r1
pickUpItem sw1
use sw1
actualTable it10
setObject r1
actualTable it13
```

```
info it10
info it13
addObject it13 Rope
actualCharacter r2
pickUpItem rp1
use rp1 r1
info it10
info it13
```

- **Elvárt kimenet**

```
File has loaded
The it13 got an impulse
it13 got a sw1
r1 moved to it13
r1’s turn
You picked up sw1 from it13
r1 used sw1
The it10 got an impulse
r1 moved to it10
The it13 got an impulse
//info it10//
Number of characters: 1
Type: hole
Strength: -
Iglu: No Iglu
Characters: r1
Neighbours: it5, it6, it9, it12, it13
Item: No item
Iced: False
Snow: 0
//info it13//
Number of characters: 1
Type: stable
Strength: -
Iglu: No Iglu
Characters: r2
Neighbours: it6, it7, it10, it12
Item: No item
Iced: False
Snow: 0
it13 got a rp1
r2’s turn
You picked up rp1 from it13
r2 used rp1
//info it10//
Number of characters: 0
```

```
Type: hole
Strength: -
Iglu: No Iglu
Characters: -
Neighbours: it5, it6, it9, it12, it13
Item: No item
Iced: False
Snow: 0
//info it13//
Number of characters: 2
Type: stable
Strength: -
Iglu: No Iglu
Characters: r1, r2
Neighbours: it6, it7, it10, it12
Item: No item
Iced: False
Snow: 0
```

8.2.7. Swimsuit használata

- **Leírás**
  Egy karakter magára vesz egy swimsuitot.
- **Ellenőrzött funkcionalitás, várható hibahelyek**
  Ellenőrzi, hogy a búvárruha felvétele beállítja-e a Character swimsuit változóját true-ra.
- **Bemenet**
  ```
  load ~/elsopalya
  actualTable it13
  addObject it13 Swimsuit
  actualCharacter r2
  info r2
  pickUpItem sw1
  use sw1
  info r2
  ```
- **Elvárt kimenet**
  ```
  File has loaded
  The it13 got an impulse
  it13 got a sw1
  r2's turn

  Actual IceTable: it9
  Character items: sh1
  Life: 4
  Energy: 4
  Type: Researcher
  ```

```
Swimsuit: False

You picked up sw1 from it9
r2 used sw1

Actual IceTable: it9
Character items: sh1
Life: 4
Energy: 4
Type: Researcher
Swimsuit: True
```

8.2.8. Shovel használata

- **Leírás**
  Hó ellapátolása, illetve jég törése.
- **Ellenőrzött funkcionalitás, várható hibahelyek**
  Ellenőrzi, hogy a lapát használatára tényleg annyi havat lapátol el, emnnyit kell neki, vagy ha nem tud lapátolni akkor betöri-e a jeget.
- **Bemenet**

```
load ~/harmadikpalya
actualCharacter r2
addObjectCharacter r2 Shovel
info it2
use sh1
info it2
pickUpItem
use sh1
pickUpItem
info it2
```

- **Elvárt kimenet**

```
File has loaded
r2's turn
r2 got a sh1
Number of characters: 1
Type: unstable
Strength: 2
Iglu: No Iglu
Characters: r2
Neighbours: it1, it3
Item: f
Iced: True
Snow: 1

r2 used sh1

Number of characters: 1
Type: unstable
Strength: 2
Iglu: No Iglu
Characters: r2
Neighbours: it1, it3
Item: f
Iced: True
Snow: 0
```

```
You couldn't pick up f from it2
r2 used sh1
You picked up f from it2
Number of characters: 1
Type: unstable
Strength: 2
Iglu: No Iglu
Characters: r2
Neighbours: it1, it3
Item: No item
Iced: False
Snow: 0
```

8.2.9. BreakeableShovel használata

- **Leírás**
  Törékeny ásó használata
- **Ellenőrzött funkcionalitás, várható hibahelyek**
  Ellenőrzi, hogy az ásó 3 használat után használhatatlan lesz-e, vagyis a hómennyiség ugyanannyi marad-e.
- **Bemenet**

```
load ~/elsopalya
actualCharacter r2
addObjectCharacter r2 BreakableShovel
getSnowStorm it13
info it13
use bsh1
use bsh1
use bsh1
info it13
use bsh1
info it13
```

- **Elvárt kimenet**

```
File has loaded
r2's turn
r2 got a bsh1
Snow storm generated on it13

Number of characters: 1
Type: stable
Strength: -
Iglu: No Iglu
Characters: r2
Neighbours: it6, it7, it10, it12
Item: No item
```

```
Iced: False
Snow: 8

r2 used bsh1
r2 used bsh1
r2 used bsh1

Number of characters: 1
Type: stable
Strength: -
Iglu: No Iglu
Characters: r2
Neighbours: it6, it7, it10, it12
Item: No item
Iced: False
Snow: 2

r2 used bsh1

Number of characters: 1
Type: stable
Strength: -
Iglu: No Iglu
Characters: r2
Neighbours: it6, it7, it10, it12
Item: No item
Iced: False
Snow: 2
```

8.2.10. Gun használat

- **Leírás**
  Gun használata.
- **Ellenőrzött funkcionalitás, várható hibahelyek**
  Ellenőrzi, hogy a jelzőpisztoly teljes megépítése után véget ér-e a játék WIN-nel.
- **Bemenet**

```
load ~/elsopalya
actualTable it13
setObject gun
setObject stencil
actualCharacter r2
pickUpItem gun
pickUpItem stencil
use gun
setObject light
pickUpItem light
```

```
use gun
```

- **Elvárt kimenet**

```
File has loaded
The it13 got an impulse
gun was put on it13
stencil was put on it13
r2's turn
You picked up gun from it13
You picked up stencil from it13
You didn't win the game! Try again!
light was put on it13
You picked up light from it13
You won the game!
```

8.2.11. Iglu építése

- **Leírás**
  Egy eszkimó építhet iglut egy jégmezőre.
- **Ellenőrzött funkcionalitás, várható hibahelyek**
  Ellenőrizzük, hogy az iglu tényleg létrejön-e a jégtáblán az Eszkimó által.
- **Bemenet**

```
load~/ elsopalya
actualTable it6
info it6
actualCharacter e1
buildIglu
info it6
```

- **Elvárt kimenet**

```
File has loaded
The it1 got an impulse
e1 moved to it1

Number of characters: 1
Type: unstable
Strength: 2
Iglu: No iglu
Characters: e1
Neighbours: it3, it5, it7, it10, it13
Item: No item
Iced: False
Snow: 0

e1's turn
An iglu created on it6

Number of characters: 1
```

```
Type: unstable
Strength: 2
Iglu: Built
Characters: e1
Neighbours: it3, it5, it7, it10, it13
Item: No item
Iced: False
Snow: 0
```

8.2.12. Iglu teljes elpusztítása

- **Leírás**

A játékosok addig rombolják az iglut, amíg el nem pusztul. Elpusztulás után, tilos, hogy a jégtábla iglut tartalmazzon.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Az iglu végleges megszűnését várjuk, ne lássa a jégtábla a bontás után az iglut.

- **Bemenet**

```
load ~/harmaspalya
actualCharacter r1
destroyIglu
destroyIglu
destroyIglu
info it1
destroy Iglu
info it1
```

- **Elvárt kimenet**

```
File has loaded
r1’s turn
r1 is destroying iglu on it1
r1 is destroying iglu on it1
r1 is destroying iglu on it1

Number of characters: 2
Type: stable
Strength: -
Iglu: Built
Characters: e1, r1
Neighbours: it2, it3
Item: No item
Iced: False
Snow: 0

r1 is destroying iglu on it1

Number of characters: 2
Type: stable
Strength: -
Iglu: No Iglu
Characters: e1, r1
Neighbours: it2, it3
Item: No item
```

```
Iced: False
Snow: 0
```

8.2.13. Strapabírás ellenőrzése

- **Leírás**

A Researcher ellenőrzi egy szomszédos mező strapabírását.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

A tényleges strapabírását csak researcher tudja ellenőrizni.

- **Bemenet**

```
load ~/kettespalya
actualCharacter r1
checkNeighbour 3.14
info it2
```

- **Elvárt kimenet**

```
File has loaded
r1's turn
Strength of it2 is 2
Number of characters: 0
Type: unstable
Strength: 2
Iglu: No Iglu
Characters: -
Neighbours: it1, it3, it4
Item: No item
Iced: False
Snow: 0
```

8.2.14. Item felvétele

- **Leírás**

Item felvétele egy jégmezőből. Befagyott itemet nem tud felvenni, felvétel után elvárt, hogy a jégtábla ne tartalmazza azt az itemet, míg a karakter igen.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Az item felvétele után a jégtábla nem tartlamazza az itemet, míg a karakter igen.

- **Bemenet**

```
load ~/kettespalya
actualTable it1
setObject rp1
setObject r1
actualCharacter r1
info r1
pickUpItem
info r1
info it1
```

- **Elvárt kimenet**

```
File has loaded
The it1 got an impulse.
rp1 was put to it1
r1 moved to it1
r1's turn

Actual IceTable: it1
Characters items: -
Life: 4
MaxLife: 4
Energy: 4
Type: Researcher
Swimsuit: False

You picked up rp1 from it1

Actual IceTable: it1
Characters items: rp1
Life: 4
MaxLife: 4
Energy: 3
Type: Researcher
Swimsuit: False

Number of characters: 1
Type: stable
Strength: -
Iglu: No Iglu
Characters: r1
Neighbours: it1, it3, it4
Item: No item
Iced: False
Snow: 0
```

8.2.15. Hóvihar Igluval

- **Leírás**

Amikor hóvihar ér egy mezőt elpusztítja az iglut, ami megvédi a benne álló(ka)t.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Hóvihar után a karakterek élete nem csökken, de a jégtáblán álló iglu eltűnik.

- **Bemenet**

```
load ~/harmaspalya
info it1
```

```
info e1
getSnowStorm it1
info it1
info e1
```

- **Elvárt kimenet**

```
File has loaded

Number of characters: 2
Type: stable
Strength: -
Iglu: Built
Characters: r1, e1
Neighbours: it2, it3
Item: No item
Iced: False
Snow: 0

Actual IceTable: it1
Characters items: -
Life: 5
MaxLife: 5
Energy: 4
Type: Eskimo
Swimsuit: False

Snow storm generated on it1

Number of characters: 2
Type: stable
Strength: -
Iglu: No iglu
Characters: r1, e1
Neighbours: it2, it3
Item: No item
Iced: True
Snow:  < 0 < Quantity >

Actual IceTable: it1
Characters items: -
Life: 5
MaxLife: 5
Energy: 4
Type: Eskimo
Swimsuit: False
```

8.2.16. Hóvihar iglu nélkül

- **Leírás**

Amikor hóvihar Iglu nélküli mezőt ér, megsebesíti a rajta álló(ka)t.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

A rajta álló összes karakter sérüljön meg.

- **Bemenet**

```
load ~/kettespalya
info r1
getSnowStorm it3
info r1
```

- **Elvárt kimenet**

```
File has loaded

Actual IceTable: it3
Characters items: -
Life: 4
MaxLife: 4
Energy: 4
Type: Researcher
Swimsuit: False
Snow storm generated on it3

Actual IceTable: it3
Characters items: -
Life: 3
MaxLife: 4
Energy: 4
Type: Researcher
Swimsuit: False
```

8.2.17. Hóvihar Hó és Jég

- **Leírás**

Amikor hóvihar ér egy mezőt, havat tesz rá és befagyasztja a rajta lévő itemet.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

A hó mennyiségének növekednie kell, az itemeknek pedig befagynia.

- **Bemenet**

```
load ~/kettespalya
info it7
getSnowStorm it7
info it7
```

- **Elvárt kimenet**

```
File has loaded

Number of characters: 0
Type: stable
Strength: -
Iglu: No iglu
Characters: -
Neighbours: it4
Item: rp1
Iced: False
Snow: 0

Snow storm generated on it7

Number of characters: 0
Type: stable
Strength: -
Iglu: No iglu
Characters: -
Neighbours: it4
Item: rp1
Iced: True
Snow:  < 0 < Quantity >
```

8.2.18. Hótakarítás

- **Leírás**

Kézzel való hótakarítás a mezőről.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Eltűnik-e egy mennyiségnyi hó a tábláról.

- **Bemenet**

```
load ~/harmaspalya
actualCharacter r2
info it2
removeSnow
info it2
```

- **Elvárt kimenet**

```
File has loaded
r2's turn

Number of characters: 1
Type: unstable
Strength: 2
Iglu: No iglu
Characters: r2
Neighbours: it1, it3
Item: -
Iced: True
Snow: 1

IceTable has been cleaned.

Number of characters: 1
Type: unstable
Strength: 2
Iglu: No iglu
Characters: r2
Neighbours: it1, it3
Item: f
Iced: True
Snow: 0
```

8.2.19. Jégtörés

- **Leírás**

Ha nincs hó a mezőn, betörjük a jegét.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

A befagyott item, elérhető lesz-e.

- **Bemenet**

```
load ~/harmaspalya
actualCharacter r2
removeSnow
info it2
removeSnow
info it2
```

- **Elvárt kimenet**

```
File has loaded
r2’s turn
IceTable has cleaned.

Number of characters: 1
Type: unstable
Strength: 2
Iglu: No iglu
Characters: r2
Neighbours: it1, it3
Item: f
Iced: True
Snow: 0

IceTable has cleaned.

Number of characters: 1
Type: unstable
Strength: 2
Iglu: No iglu
Characters: r2
Neighbours: it1, it3
Item: f
Iced: False
Snow: 0
```

8.2.20. Pályaállapot

- **Leírás**

Lekérjük a pálya jelenlegi állapotát.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

A táblák, a karakterek megfelelően vannak-e feltüntetve a parancs lefutásakor.

- **Bemenet**

```
start 0 4
info
addObject it1 Eskimo
addObject it1 Researcher
actualCharacter e1
nextPlayer
info
```

- **Elvárt kimenet**

```
Icefield created of 4 IceTables with no Characters

Number of players: 0
Number of IceTables: 4
Round number: 1

it1 got a e1
it1 got a r1
e1' turn
It is your turn: r1

Number of players: 2
Number of IceTables: 4
Round number: 2
```

8.2.21. Pálya betöltése

- **Leírás**

Egy mentett pálya betöltése.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Sikerül-e a létrehozni a várt objektumokat.

- **Bemenet**

```
load ~/kettespalya
info
info it7
info e1
```

- **Elvárt kimenet**

```
File has loaded

Number of players: 4
Number of IceTables: 7
Round number: 1

Number of characters: 0
Type: stable
Strength: -
Iglu: No iglu
Characters: -
Neighbours: it4
Item: rp1
Iced: False
Snow: 0
Actual IceTable: it6
Characters items: -
Life: 5
MaxLife: 5
Energy: 4
Type: Eskimo
Swimsuit: False
```

8.2.22. Állás mentése

- **Leírás**

A játék szüneteltetése kilépéssel

- **Ellenőrzött funkcionalitás, várható hibahelyek**
  Helyesen tölt- e be az elmentett játékállás.
- **Bemenet**

```
load ~/elsopalya
info it12
info r1
actualCharacter r1
step 0,52
info it12
info it13
info r1
save ~/elsopalya
```

- **Elvárt kimenet**

```
File has loaded

Number of characters: 1
Type: unstable
Strength: 2
Iglu: No Iglu
Characters: r1
Neighbours: it9, it10, it11, it13
Item: No item
Iced: False
Snow: 0

Actual IceTable: it12
Character items: -
Life: 4
Energy: 4
Type: Researcher
Swimsuit: False

r1’s turn
r1 moves to it13

Number of characters: 0
Type: unstable
Strength: 2
Iglu: No Iglu
Characters: r1, r2
Neighbours: it9, it10, it11, it13
```

```
Item: No item
Iced: False
Snow: 0

Number of characters: 1
Type: unstable
Strength: 2
Iglu: No Iglu
Characters: r1, r2
Neighbours: it6, it7, it10, it12
Item: No Item
Iced: False
Snow: 0

Actual IceTable: it13
Character items: -
Life: 4
Energy: 4
Type: Researcher
Swimsuit: False

File has saved successfully.
```

8.2.23. Következő játékos

- **Leírás**

Amikor egy játékosnak elfogy a 4 energiája, a soron következő cselekedhet.

- **Ellenőrzött funkcionalitás, várható hibahelyek**
  A négy energia felhasználása után a játékos valóban nem tud- e a játékos többet cselekedni és a másik játékos következik és az valóban 4 energiával kezdi cselekvés sorozatát.
- **Bemenet**

```
load ~/elsopalya
info r1
nextPlayer
info r2
```

- **Elvárt kimenet**

```
File has loaded


Actual IceTable: it12
Character items: -
Life: 4
Energy: 0
Type: Researcher
Swimsuit: False

It is your turn r2.

Actual IceTable: it3
Character items: -
Life: 4
Energy: 4
Type: Researcher
Swimsuit: False
```

8.2.24. Objektum hozzáadása a jégtáblához

- **Leírás**

Lépéskor, vagy a pálya létrehozásakor hozzáadjuk a karaktert vagy jegesmedvét a jégtáblához, amin áll.

- **Ellenőrzött funkcionalitás, várható hibahelyek**
  Tartamazza- e a hozzáadott objektumot ténylegesen az adott jégtábla.
- **Bemenet**

```
load ~/masodikpalya
info it2
actualTable it2
addObject it2 Camp
info it2
```

- **Elvárt kimenet**

```
File has loaded.

Number of characters: 0
Type: unstable
Strength: 2
Iglu: No Iglu
Characters:No Character
Neighbours: it1, it3, it4
Item: No item
Iced: False
Snow: 0

It2 got a Camp.

Number of characters: 0
Type: unstable
Strength: 2
Iglu: No Iglu
Characters:
Neighbours: it1, it3, it4
Item: Camp
Iced: False
Snow: 0
```

8.2.25. Item hozzáadása a jégtáblához

- **Leírás**

Pályalétrehozáskor belerakjuk az Itemet egy mezőbe.

- **Ellenőrzött funkcionalitás, várható hibahelyek**
  Tartamazza- e a hozzáadott itemet ténylegesen az adott jégtábla.
- **Bemenet**

```
load ~/masodikpalya
info it2
actualTable it2
addObject it2 Shovel
info it2
```

- **Elvárt kimenet**

```
File has loaded.

Number of characters: 0
Type: unstable
Strength: 2
Iglu: No Iglu
Characters:
Neighbours: it1, it3, it4
Item: No item
Iced: False
Snow: 0

It2 got a Shovel.

Number of characters: 0
Type: unstable
Strength: 2
Iglu: No Iglu
Characters:
Neighbours: it1, it3, it4
Item: Shovel
Iced: False
Snow: 0
```

8.2.26. Instabilra lépés

- **Leírás**

Egy Character az instabil mezőre.

- **Ellenőrzött funkcionalitás, várható hibahelyek**
  Ha kevesebb ember tartózkodik a jégtáblán, mint annak a teherbírása valóban nem szakad-e be a jégtábla.
- **Bemenet**

```
load ~/elsopalya
info it13
info r2
actualCharacter r2
step 2,093
info it13
info it6
info r2
```

- **Elvárt kimenet**

```
File has loaded

Number of characters: 1
Type: stable
Strength: 2
Iglu: No Iglu
Characters: r2
Neighbours: it6, it7, it10, it12
Item: No item
Iced: False
Snow: 0

Actual IceTable: it13
Character items: -
Life: 4
Energy: 4
Type: Researcher
Swimsuit: False

r2’s turn
r2 moves to it6

Number of characters: 0
Type: stable
Strength: 2
Iglu: No Iglu
Characters:
```

```
Neighbours: it6, it7, it10, it12
Item: No item
Iced: False
Snow: 0

Number of characters: 2
Type: unstable
Strength: 2
Iglu: No Iglu
Characters: e1, r2
Neighbours: it3, it5, it7, it10, it13
Item: No Item
Iced: False
Snow: 0

Actual IceTable: it6
Character items: -
Life: 4
Energy: 3
Type: Researcher
Swimsuit: False
```

8.2.27. Instabil beszakadása

- **Leírás**

Amikor több játékos van egy mezőn, mint amennyit elbír, a mező beszakad és a játék véget ér

- **Ellenőrzött funkcionalitás, várható hibahelyek**
  Ha több játékos tartózkodik az adott jégtáblán, mint annak a teherbírása valóban beszakad-e a jégtábla és a játék véget ér- e lose-zal.
- **Bemenet**

```
load ~/elsopalya
actualCharacter r1
step 0,52
step 2,093
nextPlayer
actualCharacter r2
step 2,093
info it6
info r1
info r2
info e1
```

- **Elvárt kimenet**

```
File has loaded

r1's turn.
```

```
r1 moves to it13
r1 moves to it6
it is your turn r2
r2's turn
r2 moves to it6
Number of characters: 3
Type: stable
Strength:2
Iglu: No Iglu
Characters: r1, r2, e1
Neighbours: it3, it5, it7, it10, it13
Item: No item
Iced: False
Snow: 0

Actual IceTable: it6
Character items: -
Life: 0
Energy: 4
Type: Researcher
Swimsuit: False

Actual IceTable: it6
Character items: -
Life: 0
Energy: 3
Type: Researcher
Swimsuit: False

Actual IceTable: it6
Character items: -
Life: 0
Energy: 4
Type: Eskimo
Swimsuit: False
r1's turn
r1 moves to it2
```

8.2.28. Lyukra lépés

- **Leírás**

Ha egy játékos lyukra lép, és nem mentik ki, meghal.

- **Ellenőrzött funkcionalitás, várható hibahelyek**
  Ellenőrzi- e hogy az adott játékos visel- e búvárruhát és/vagy tartózkodik- e csapattárs a jégtábla szomszédos mezőjén, ha nem valóban véget ér- e a játék lose-zal.

- **Bemenet**

```
load ~/egyespalya
info it12
info r1
actualCharacter r1
step 3,663
info it12
info it11
info r1
```

- **Elvárt kimenet**

```
File has loaded

Number of characters: 1
Type: stable
Strength:
Iglu: No Iglu
Characters: r1
Neighbours: it9, it10, it11, it13
Item: No item
Iced: False
Snow: 0

Actual IceTable: it12
Character items: -
Life: 4
Energy: 4
Type: Researcher
Swimsuit: False

r1’s turn
r1 moves to it11

Number of characters: 0
Type: stable
Strength:
Iglu: No Iglu
Characters:
Neighbours: it9, it10, it11, it13
Item: No item
Iced: False
Snow: 0

Number of characters: 1
Type: hole
Strength: 0
```

```
Iglu: No Iglu
Characters: r1
Neighbours: it8, it9, it12
Item: No Item
Iced: False
Snow: 0

Actual IceTable: it11
Character items: -
Life: 0
Energy: 3
Type: Researcher
Swimsuit: False

Bye.
```

8.2.29. Karakter megfagyása

- **Leírás**

Ha elfogy a karakter élete/testhője meghal.

- **Ellenőrzött funkcionalitás, várható hibahelyek**
  Ha a játékos életének értéke 1 és hóviharba kerül, befejeződik- e a játék lose-zal.
- **Bemenet**

```
load ~/elsopalya
getSnowStorm it9 it10 it11 it12 it13
info r1
```

- **Elvárt kimenet**

```
File has loaded

Snow storm has generated on the icefields.

Actual IceTable: it12
Character items: -
Life: 0
Energy: 4
Type: Researcher
Swimsuit: False
```

8.2.30. Jegesmedve ölése

- **Leírás**

A jegesmedve megöli a vele egy mezőn álló Charactert.

- **Ellenőrzött funkcionalitás, várható hibahelyek**
  Ha a jegesmedve rálép az adott jégtáblára és tartózkodik rajta legalább egy Character, akkor befejeződik-e a játék lose-zal.
- **Bemenet**

```
load ~/elsopalya
actualCharacter pb1
step 5,233
step 0
info it6
info e1
```

- **Elvárt kimenet**

```
File has loaded

pb1’s turn.

pb1 moves to it5.
pb1 moves to it6.

Number of characters: 2
Type: stable
Strength:
Iglu: No Iglu
Characters: r1, pb1
Neighbours: it1, it3, it4, it5
Item: No Item
Iced: False
Snow: 0

Actual IceTable: it2
Character items: -
Life: 0
Energy: 2
Type: Eskimo
Swimsuit: False
```

8.2.31. Játék befejezése

- **Leírás**

A fegyver összeszerelése és elsütése

- **Ellenőrzött funkcionalitás, várható hibahelyek**
  A játék végeztével valóban véget ér- e a játék, méghozzá win-nel- e?
- **Bemenet**

```
load ~/elsopalya
actualCharacter r1
step 2,355
step 3,14
pickUpItem g1
step 2,0933
nextPlayer
step 3,663
step 2,093
step 1,046
pickUpItem stencil
nextPlayer
step 5,233
step 3,663
step 2,616
step 1.046
nextPlayer
pickUpItem light
step 5,233
step 0
step 1,046
nextPlayer
useItem Gun
```

- **Elvárt kimenet**

```
File has loaded.

r1’s turn
r1 moves to it9
r1 moves to it8
r1 picked up gun from it8
r1 moves to it1
It is your turn r2
r2 moves to it12
r2 moves to it9
r2 moves to it5
r2 picked up stencil from it5
```

```
It is your turn e1
e1 moves to it13
e1 moves to it12
e1 moves to it9
e1 moves to it5
It is your turn r1
r1 picked up light from it1
r1 moves to it8
r1 moves to it9
r1 moves to it5
it is your turn r2
You won the Game.
```

8.2.32. Meglepetés használata

- **Leírás**

Véletlenszerű pozitív vagy negatív hatás egy Characterre.

- **Ellenőrzött funkcionalitás, várható hibahelyek**
  A kapott véletlen esemény valóban azt a kimenetelt eredményezi- e amit a random kiválasztó szeretett volna
- **Bemenet**

```
load ~/elsopalya
setObject s1
actualCharacter r1
pickUpItem
use s1
```

- **Elvárt kimenet**

```
File has loaded.

Surprise was put on it12.
r1's turn.
You picked up Surprise from it12.
R1 used Surprise.
```

8.3. A tesztelést támogató programok tervei

Minden teszteset utasításai külön txt fájlokba lesznek kiírva soronként tagolva. A protoban kiválaszthatjuk, hogy melyik teszt fusson le. A proto soronként nézi az inputot, tagolva a benne lévő sorokat utasítás paraméter1 paraméter2 stb. sorrendben. Egy teszt lefutása után minden belső változást és eredményt kiírunk egy output txt-be, ami kezdetben üres vagy nem is létezik. A Tesztprogram külön Java programként fog működni. Ebben a programban kiválaszthatjuk, hogy a már lefutott teszteket összehasonlítsuk a teszthez statikusan megadott elvárt kimenetekkel. A program kiírja konzolra az elvárt és tényleges kimenetet minden változásra, beleértve a belső állapotváltozásokat is, és melléjük hogy a teszt lefutása sikerrel vagy sikertelenül fejeződött-e be.

8.4. Napló

Kezdet	Időtartam	Résztvevők	Leírás
2020.04.08 13:00	2 óra	Csoma Sipula Szepesi-Nagy Varga Ziaja	Előző feladatrész értékelésének átbeszélése, javítások. Feladatok felvázolása, megbeszélése és kiosztása.
2020.04.09 16:00	1 óra	Varga Ziaja	Konzultáció a tesztekről
2020.04.10 9:00	2 óra	Ziaja Szepesi-Nagy	Tesztek átbeszélése, tervezése
2020.04.11 8:00	3 óra	Ziaja	Tesztek leírásai, bemenetei elkészítése
2020.04.11 18:00	1 óra	Sipula Varga	Tesztelést támogató program terveinek átbeszélése és leírása
2020.04.12 20:00	3 óra	Csoma Sipula Szepesi-Nagy Varga Ziaja	Objektumok és metódusok tervei
2020.04.13 11:00	4 óra	Csoma Szepesi-Nagy Ziaja	A tesztek részletes leírása
2020.04.13 15:00	2 óra	Sipula Varga	Dokumentum átnézése, hibák javítása
2020.04.14 10:00	1 óra	Sipula Ziaja	Dokumentum véglegesítése

10. Prototípus beadása

10.0 Változások

10.1 Beépített tesztelő

Nem külön alkalmazás a tesztelő, hanem beépítésre került a Proto-ba. Indításnál lehet választani, hogy a protot vagy a teszt részét szeretnénk indítani.

10.2 Konzolon csak annyi szerepel amennyi fontos

A jobb átláthatóság érdekében nem írjuk ki a konzolra a minden egyes sorát a tesztnek, hanem rögtön egy output fájl-ba. A konzolra vagy annyit írunk ki, hogy a Teszt hiba nélkül lefutott, vagy ha esetleg volt hiba, akkor, hogy a teszt hibásan futott le és a hibás sor elvárt és tényleges kimenetét.

10.3 Automatikus teszt ellenőrzés

A tesztelésnél nem külön választható, hogy ellenőrizni akarjuk-e az elvárt és tényleges kimenetet, hanem automatikus ellenőrzésre kerül.

10.4 Fordítási és futtatási útmutató

10.4.1 Fájllista

Fájl neve	Méret(Byte)	Keletkezés ideje	Tartalom
BreakableShovel.java	1361	2020.04.01. 17:58	A törékeny ásó megvalósítása.
Camp.java	746	2020.04.01. 17:58	A sátor megvalósítása.
Character.java	7641	2020.03.29. 21:55	A karakterek ősosztálya, közös függvények megvalósítása.
Eskimo.java	713	2020.03.29. 21:55	A játékban szereplő egyik karaktertípust, az eszkimó és annak speciális képességének(iglu építés) megvalósítása.
Food.java	494	2020.03.29. 21:55	Az étel használatának megvalósítása.
GameWorld.java	22472	2020.03.29. 21:55	A játékkezelő megvalósítása.
Gun.java	1085	2020.03.29. 21:55	A fegyver megvalósítása, ami a jelzőpisztoly első alkatrésze.
IceTable.java	8273	2020.03.29. 21:55	Egy jégmező és annak függvényeinek megvalósítva.
Iglu.java	1631	2020.03.29. 21:55	Az iglu megvalósítása.
Item.java	395	2020.03.29. 21:55	Az eszközök interfésze.

ItemOwner.java	704	2020.03.29. 21:55	Az IceTable és Character absztrakt őse. Eszközkezelést valósít meg.
Light.java	530	2020.03.29. 21:55	A jelzőpisztoly harmadik alkatrésze.
Main.java	805	2020.03.29. 21:55	A program fő osztálya, ahol a menü került megvalósításra.
PolarBear.java	1922	2020.04.01. 17:58	A jegesmedve megvalósítása.
Reserchер.java	819	2020.03.29. 21:55	A játékban szereplő egyik karaktertípust, a sarkkutató és annak speciális képességének megvalósítása.
Rope.java	825	2020.03.29. 21:55	A kötelet megvalósítása.
Shovel.java	1224	2020.03.29. 21:55	Az ásó megvalósítása.
Stencil.java	1136	2020.03.29. 21:55	A patron megvalósítása.
Stepable.java	1633	2020.04.01. 17:58	A lépő elemek ősosztálya.
Surprise.java	927	2020.03.29. 21:55	A meglepetés megvalósítsa.
Swimsuit.java	512	2020.03.29. 21:55	Az úszóruha megvalósítása.
Test.java	31681	2020.04.20. 16:30	A tesztelést végző osztály.

Fájl neve	**Méret**	**Keletkezés ideje**	**Tartalom**
EO01.txt	189	2020.04.20. 12:40	1.teszt elvárt kimenete.
EO02.txt	715	2020.04.20. 12:41	2.teszt elvárt kimenete.
EO03.txt	259	2020.04.20. 12:41	3.teszt elvárt kimenete.
EO04.txt	523	2020.04.20. 12:42	4.teszt elvárt kimenete.
EO05.txt	885	2020.04.20. 12:42	5.teszt elvárt kimenete.
EO06.txt	985	2020.04.20. 12:42	6.teszt elvárt kimenete.
EO07.txt	277	2020.04.20. 12:43	7.teszt elvárt kimenete.
EO08.txt	866	2020.04.20. 12:43	8.teszt elvárt kimenete.
EO09.txt	647	2020.04.20. 12:44	9.teszt elvárt kimenete.
EO10.txt	167	2020.04.20. 12:45	10.teszt elvárt kimenete.
EO11.txt	413	2020.04.20. 12:45	11.teszt elvárt kimenete.
EO12.txt	477	2020.04.20. 12:45	12.teszt elvárt kimenete.
EO13.txt	50	2020.04.20. 12:46	13.teszt elvárt kimenete.
EO14.txt	568	2020.04.20. 12:47	14.teszt elvárt kimenete.
EO15.txt	572	2020.04.20. 12:47	15.teszt elvárt kimenete.
EO16.txt	252	2020.04.20. 12:47	16.teszt elvárt kimenete.
EO17.txt	354	2020.04.20. 12:48	17.teszt elvárt kimenete.
EO18.txt	379	2020.04.20. 12:49	18.teszt elvárt kimenete.
EO19.txt	378	2020.04.20. 12:49	19.teszt elvárt kimenete.
EO20.txt	310	2020.04.20. 12:49	20.teszt elvárt kimenete.
EO21.txt	887	2020.04.20. 12:50	21.teszt elvárt kimenete.
EO22.txt	42	2020.04.20. 12:50	22.teszt elvárt kimenete.

EO23.txt	83	2020.04.20. 12:51	23.teszt elvárt kimenete.
EO24.txt	377	2020.04.20. 12:51	24.teszt elvárt kimenete.
EO25.txt	372	2020.04.20. 12:51	25.teszt elvárt kimenete.
EO26.txt	490	2020.04.20. 12:51	26.teszt elvárt kimenete.
EO27.txt	355	2020.04.20. 12:52	27.teszt elvárt kimenete.
EO28.txt	482	2020.04.20. 12:52	28.teszt elvárt kimenete.
EO29.txt	265	2020.04.20. 12:53	29.teszt elvárt kimenete.
EO30.txt	402	2020.04.20. 12:53	30.teszt elvárt kimenete.
EO31.txt	108	2020.04.20. 12:54	31.teszt elvárt kimenete.
EO32.txt	166	2020.04.20. 12:55	32.teszt elvárt kimenete.

Fájl neve	**Méret**	**Keletkezés ideje**	**Tartalom**
elsopalya.txt	1057	2020.04.20. 10:31	Az első pálya leíró bemenete.
harmadikpalya.txt	271	2020.04.20. 10:31	A második pálya leíró bemenete.
TEST01.txt	351	2020.04.20. 10:32	1.teszt bemenete
TEST02.txt	94	2020.04.20. 10:33	2.teszt bemenete
TEST03.txt	88	2020.04.20. 10:34	3.teszt bemenete
TEST04.txt	151	2020.04.20. 10:35	4.teszt bemenete
TEST05.txt	231	2020.04.20. 10:36	5.teszt bemenete
TEST06.txt	162	2020.04.20. 10:37	6.teszt bemenete
TEST07.txt	92	2020.04.20. 10:38	7.teszt bemenete
TEST08.txt	171	2020.04.20. 10:39	8.teszt bemenete
TEST09.txt	150	2020.04.20. 10:42	9.teszt bemenete
TEST10.txt	149	2020.04.20. 10:43	10.teszt bemenete
TEST11.txt	69	2020.04.20. 10:45	11.teszt bemenete
TEST12.txt	112	2020.04.20. 10:46	12.teszt bemenete
TEST13.txt	56	2020.04.20. 10:47	13.teszt bemenete
TEST14.txt	88	2020.04.20. 10:48	14.teszt bemenete
TEST15.txt	96	2020.04.20. 10:49	15.teszt bemenete
TEST16.txt	76	2020.04.20. 10:50	16.teszt bemenete
TEST17.txt	60	2020.04.20. 10:52	17.teszt bemenete
TEST18.txt	87	2020.04.20. 10:53	18.teszt bemenete
TEST19.txt	89	2020.04.20. 10:54	19.teszt bemenete
TEST20.txt	131	2020.04.20. 10:55	20.teszt bemenete
TEST21.txt	143	2020.04.20. 10:56	21.teszt bemenete
TEST22.txt	26	2020.04.20. 10:57	22.teszt bemenete
TEST23.txt	94	2020.04.20. 10:58	23.teszt bemenete
TEST24.txt	62	2020.04.20. 10:59	24.teszt bemenete
TEST25.txt	62	2020.04.20. 11:01	25.teszt bemenete
TEST26.txt	75	2020.04.20. 11:02	26.teszt bemenete
TEST27.txt	120	2020.04.20. 11:03	27.teszt bemenete
TEST28.txt	111	2020.04.20. 11:05	28.teszt bemenete

TEST29.txt	121	2020.04.20. 11:06	29.teszt bemenete
TEST30.txt	67	2020.04.20. 11:07	30.teszt bemenete
TEST31.txt	133	2020.04.20. 11:09	31.teszt bemenete
TEST32.txt	83	2020.04.20. 11:11	32.teszt bemenete

10.4.2 Fordítás és Futtatás

A zip fájl kicsomagolása után nyissuk meg az Eclipse Java 2018-12 alkalmazást ami a kari cloudon megtalálható. Itt válasszuk a bal felső sarokban a File menüpontot, azon belül az Import lehetőséget. Itt a General alatt a "Projects from Folder or Archive"-ot kell választanunk. Megadjuk az Antarctica mappát és a Finish gombra kattintva hozzáadásra is kerül a projekt.
Megnyitjuk a projektet a Package Explorer-ben dupla kattintással és a fenti menüből kiválasztjuk a Run Main lehetőséget. Ekkor el fog indulni a program.
A program elindulása után a 0-ás bemenetre elindul a tesztelő része az alkalmazásnak. Ott 0-ás bemenetre futtathatjuk az összes tesztet, vagy 1 és 32 között választhatunk 1 tesztet futtatásra. Egy teszt lefutása után egyből ellenőrzi, hogy megegyezik-e a kimenet az elvárt kimenettel. Ha megegyezik, akkor jelzi a tesztelőnek, hogy sikeresen futott a teszt, ha nem, akkor kiírja console outputra a hibás sort az elvárt és a tényleges kimenettel. Inputról csak a futtatandó tesztesetet lehet beolvasni, bemeneteket külön nem lehet megadni.

10.5 *Tesztek jegyzőkönyvei*

10.5.1 Pálya

Tesztelő neve	Varga
Teszt időpontja	2020.04.26. 21:12

10.5.2 Lépés

Tesztelő neve	Varga
Teszt időpontja	2020.04.26. 21:13

10.5.3 Item használata

Tesztelő neve	Varga
Teszt időpontja	2020.04.26. 21:14

10.5.4 Food használata

Tesztelő neve	Varga
Teszt időpontja	2020.04.26. 21:16

10.5.5 Camp használata

Tesztelő neve	Varga
Teszt időpontja	2020.04.26. 21:20

10.5.6 Rope használata

Tesztelő neve	Varga
Teszt időpontja	2020.04.26. 21:21

10.5.7 Swimsuit használata

Tesztelő neve	Varga
Teszt időpontja	2020.04.26. 21:21

10.5.8 Shovel használata

Tesztelő neve	Varga
Teszt időpontja	2020.04.26. 21:22

10.5.9 BreakableShovel használata

Tesztelő neve	Varga
Teszt időpontja	2020.04.26. 21:24

10.5.10 Gun használata

Tesztelő neve	Varga
Teszt időpontja	2020.04.26. 21:26

10.5.11 Iglu építése

Tesztelő neve	Varga
Teszt időpontja	2020.04.26. 21:26

10.5.12 Iglu teljes elpusztítása

Tesztelő neve	Varga
Teszt időpontja	2020.04.26. 21:28

10.5.13 Strapabírás ellenőrzése

Tesztelő neve	Varga
Teszt időpontja	2020.04.26. 21:30

10.5.14 Item felvétele

Tesztelő neve	Varga
Teszt időpontja	2020.04.26. 21:30

10.5.15 Hóvihar Igluval

Tesztelő neve	Varga
Teszt időpontja	2020.04.26. 21:31

10.5.16 Hóvihar iglu nélkül

Tesztelő neve	Varga
Teszt időpontja	2020.04.26. 21:33

10.5.17 Hóvihar Hó és Jég

Tesztelő neve	Varga
Teszt időpontja	2020.04.26. 21:32

10.5.18 Hótakarítás

Tesztelő neve	Varga
Teszt időpontja	2020.04.26. 21:17

10.5.19 Jégtörés

Tesztelő neve	Varga
Teszt időpontja	2020.04.26. 21:15

10.5.20 Pályaállapot

Tesztelő neve	Varga
Teszt időpontja	2020.04.26. 21:09

10.5.21 Pálya betöltése

Tesztelő neve	Varga
Teszt időpontja	2020.04.26. 21:10

10.5.22 Állás mentése

Tesztelő neve	Varga
Teszt időpontja	2020.04.26. 21:36

10.5.23 Következő játékos

Tesztelő neve	Varga
Teszt időpontja	2020.04.26. 21:38

10.5.24 Objektum hozzáadása a jégtáblához

Tesztelő neve	Varga
Teszt időpontja	2020.04.26. 21:34

10.5.25 Item hozzáadása a jégtáblához

Tesztelő neve	Varga
Teszt időpontja	2020.04.26. 21:40

10.5.26 Instabilra lépés

Tesztelő neve	Varga
Teszt időpontja	2020.04.26. 21:39

10.5.27 Instabil beszakadása

Tesztelő neve	Varga
Teszt időpontja	2020.04.26. 21:42

10.5.28 Lyukra lépés

Tesztelő neve	Varga
Teszt időpontja	2020.04.26. 21:44

10.5.29 Karakter megfagyása

Tesztelő neve	Varga
Teszt időpontja	2020.04.26. 21:40

10.5.30 Jegesmedve ölése

Tesztelő neve	Varga
Teszt időpontja	2020.04.26. 21:41

10.5.31 Játék befejezése

Tesztelő neve	Varga
Teszt időpontja	2020.04.26. 21:41

10.5.32 Meglepetés tesztelése

Tesztelő neve	Varga
Teszt időpontja	2020.04.26. 21:45

10.6 Értékelés

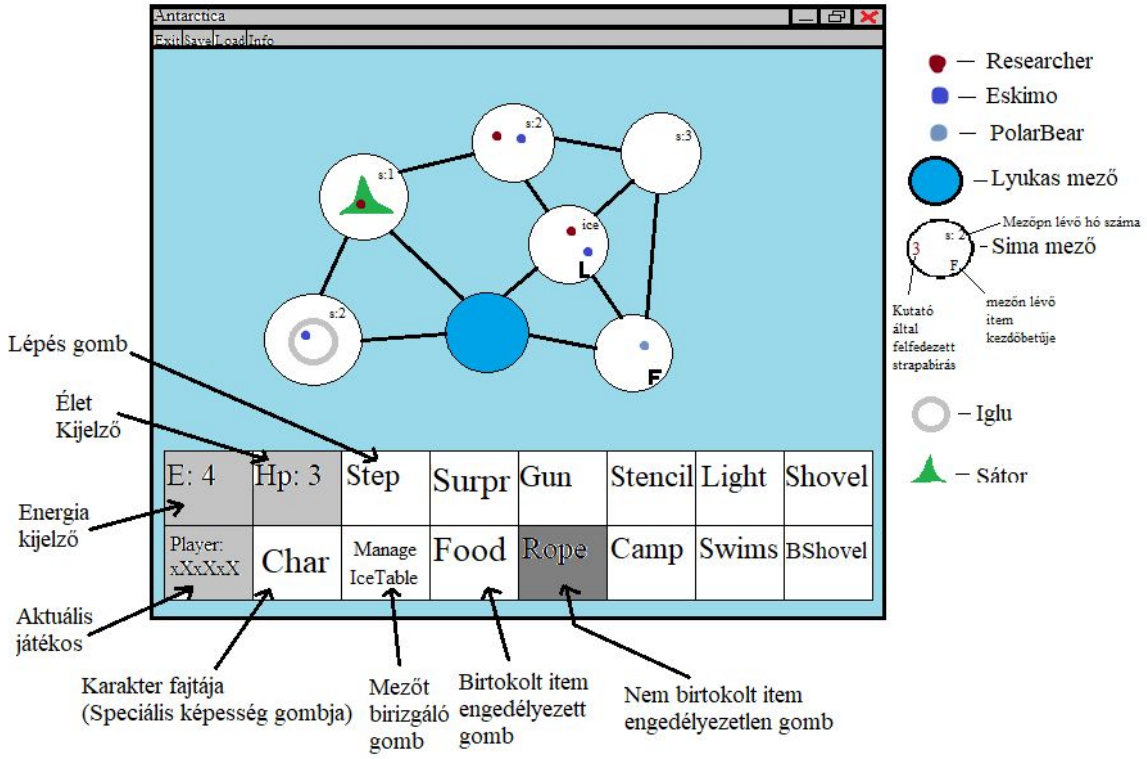
Tag neve	Tag neptun	Munka százalékban
Csoma Zoltán	G0IOFT	20%
Sipula László Márk	A1D4QD	20%
Szepesi-Nagy István	K45SFS	20%
Varga Zsombor	AKCJOP	20%
Ziaja Bálint	HICYBO	20%

10.7 Napló

Kezdet	Időtartam	Résztvevők	Leírás
2020.04.13. 14:00	2,5 óra	Csoma Szepesi-Nagy Sipula Varga Ziaja	Feladatok kiosztása, fontosabb dolgok átbeszélése
2020.04.16 10:00	7 óra	Csoma Szepesi-Nagy Varga	Proto tervezése, szkeleton bővítése és főbb funkciük megvalósítása
2020.04.17 10:00	6 óra	Sipula Varga Ziaja	Proto befejezése és tesztelő rész megírása
2020.04.20 13:00	3 óra	Csoma Szepesi-Nagy Sipula Varga Ziaja	Kommentek elhelyezése, program átnézése
2020.04.26. 21:00	2 óra	Sipula Varga	Dokumentáció elkészítése. Dokumentáció és program véglegesítése.

11. Grafikus felület specifikációja

11.1 A grafikus interfész



A fenti képen látható a játék grafikus felületének terve.
Felül a menüsorban a különböző menüpontokat (Exit - kilépés, Save - játék mentése, Load - játék betöltése és az Info - információ a játékról) találhatjuk meg.
Ez alatt található maga a játékfelület. A játék háttere kék, ami a jégmezőt körülölelő tengert reprezentálja. A felső részen gráfszerűen elrendezve található maga a pálya, vagyis a jégmezőt alkotó jégtáblák. Minden jégtáblát egy-egy kör és a táblák szomszédságát a körök között behúzott vonal jelöl. A táblákon megtalálható különböző szereplőket a különböző pontok jelölik. Egy sarkkutatót egy bordó, egy eszkimót egy kék és egy jegesmedvét egy zöldes-kék pont jelöl. Az iglut egy szürke kör, míg a sátrat egy zöld háromszög.
Ha egy sarkkutató már felmérte egy tábla teherbírását, azt a tábla bal oldalán jelenítjük meg, a hó vastagságát a jobb felső sarokban. Amennyiben már nincs hó rajta, de be van fagyva, akkor ez átváltozik ice feliratra.
Ha egy táblán található valamilyen tárgy, annak a kezdő vagy első két betűjével jelöljük a jobb alsó részén a táblának.
A stabil és instabil táblák nincsenek megkülönböztetve, ugyanúgy fehérek, a lyukas jégtábla viszont kék.
A képernyő alsó felében található a kezelőfelület. Itt az első sor első két rubrikájában jelenik meg az aktuális játékos energiája és élet. A második sor első rublikájában az aktuális karakter neve és mellette a karakterének típusa, mellyel annak a különleges képességét tudjuk kiváltani. Ezektől jobbra a második sorban található a Manage IceTable gomb, amivel a karakter jégtábláját tudjuk

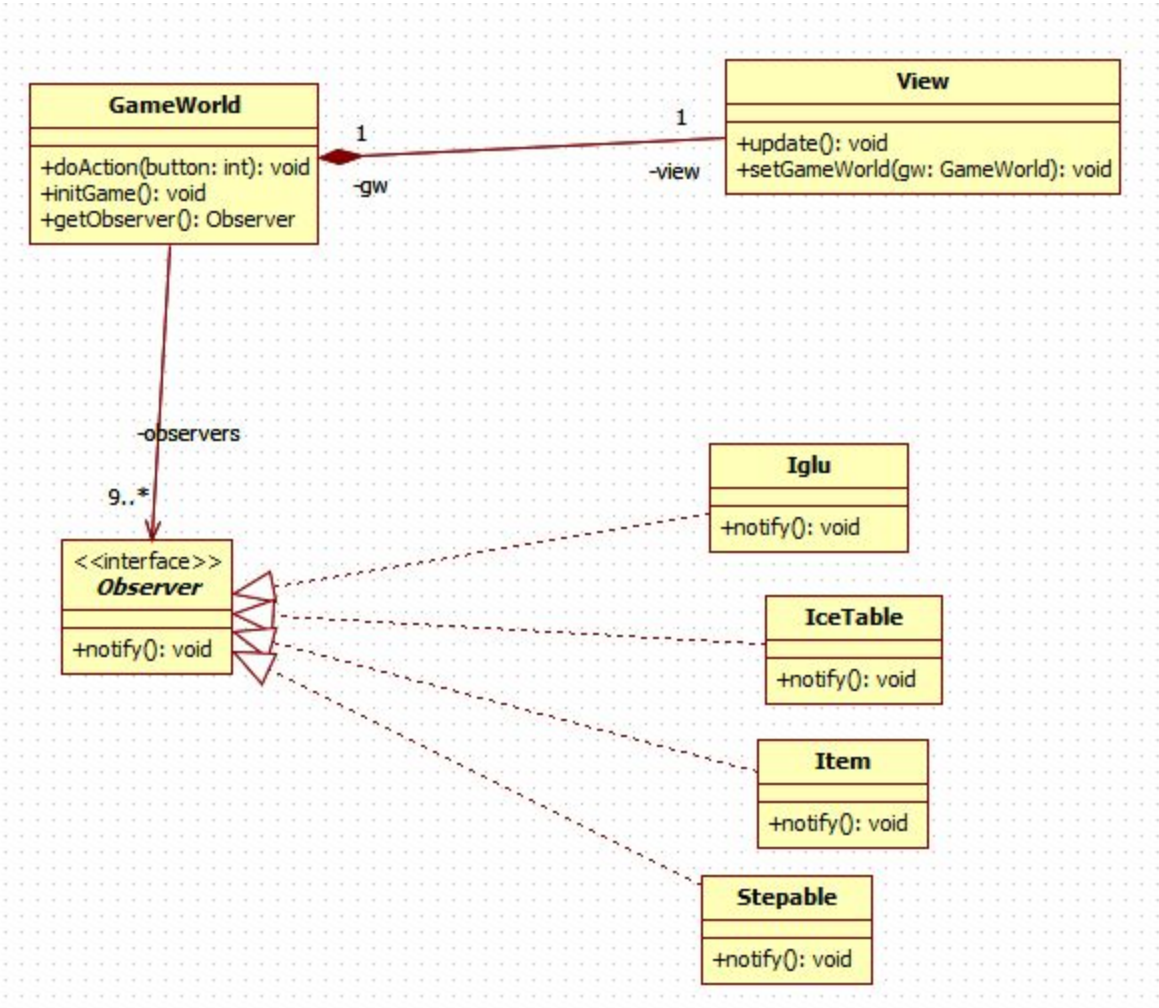
kezelni, vagyis kézzel havat eltávolítani, vagy ha nincs rajta hó és be van fagyva, akkor a jeget feltörni.
Azok a gombok, amik eddig nem kerültek felsorolásra, mind a játékban megtalálható eszközöket jelölik. Amennyiben egy játékosnál már megtalálható valamelyik, úgy inaktívról aktívra változik a gomb és már használható is.

11.2 A grafikus rendszer architektúrája

11.2.1 A felület működési elve

A grafikus felület tervezésekor törekedtünk az **MVC** elveinek a követésére. Így a továbbiakban igyekszünk azokat betartani.
Az eddigi osztályokat próbáltuk a lehető legkevésbé módosítani, hogy a későbbi bővíthetőséget lehetővé tegyük.
Létrehoztunk egy **View** osztályt, ami a grafikus megjelenítésért felel. Az **Observer** egy interfész, melyet a pályán megtalálható különböző elemek valósítanak meg. A View és Observer osztályokon keresztül történik az input beérkezéséről való értesítés a játék felé, és visszafelé is az értesítés hatására változott objektumok jelzése, hogy ők változtak, újra ki kell őket rajzolni.
Így az MVC mellett **pull** alapelvet használunk, amennyiben egy objektum állapota változott, az jelez, hogy ő megváltozott.

11.2.2 A felület osztály-struktúrája



11.3 A grafikus objektumok felsorolása

11.3.1 View

- **Felelősség**

A programban található dolgok grafikus megjelenítéséért felel.

- **Ősosztályok**
  - -
- **Interfészek**
  - -
- **Attribútumok**
  - **gw: GameWorld:** A játékvilág, amit megjelenítünk a View segítségével
- **Metódusok**
  - **+void update():** Frissíti a grafikus megjelenítést egy változás hatására
  - **+void setGameWorld(GameWorld gw):** GameWorld setter, beállítja a megjelenítendő GameWorld-öt.

11.3.2 Observer

- **Felelősség**

A játékban lévő objektumok egy interfésze, ezen keresztül tárolja őket a GameWorld és ezen keresztül jelez nekik input hatására.

- **Ősosztályok**
  - -
- **Interfészek**
  - -
- **Attribútumok**
  - -
- **Metódusok**
  - **+void notify():** A jelzés után, ha az objektumon történik változás, akkor frissíti az állapotát a View-nak.

11.3.3 GameWorld (csak ami új)

- **Felelősség**

Az eddig meglévő felelőssége mellett úgy is viselkedik, mint egy Controller.

- **Attribútumok**
  - **view: View**: A megjelenítésért felelős objektum a játéktérben.
  - **observers: Observer[ ]**: A megjelenítésért felelős objektum a játéktérben.

- **Metódusok**
  - **+void doAction(int button):** Gombnyomás hatására létrejövő cselekmény. Ezzel irányíthatjuk a karakterünket.
  - **+void initGame():** A játék inicializálása kibővítve: View létrehozása és első beállítása.
  - **+Observer getObserver():** Az observerek objektumokra getter függvény, ezzel elérhetővé válik kívülről az összes megjelenítendő objektum.

11.3.4 IceTable (csak ami új)

- **Interfészek**

  Observer

- **Metódusok**
  - **+void notify():** A saját gameWorld attribútuma segítségével a view attribútum megszerzése után, meghívja az update() függvényt, amivel frissül a grafikus megjelenítés.

11.3.5 Item (csak ami új)

- **Interfészek**

  Observer

- **Metódusok**
  - **+void notify():** A használatkor paraméterként kapott karakter jégtáblája segítségével megszerzi a GameWorld objektumot, a view attribútum megszerzése után, meghívja az update() függvényt, amivel frissül a grafikus megjelenítés.

11.3.6 Stepable (csak ami új)

- **Interfészek**

  Observer

- **Metódusok**
  - **+void notify():** A jégtáblája segítségével megszerzi a GameWorld objektumot, a view attribútum megszerzése után, meghívja az update() függvényt, amivel frissül a grafikus megjelenítés.

11.3.7 Iglu (csak ami új)

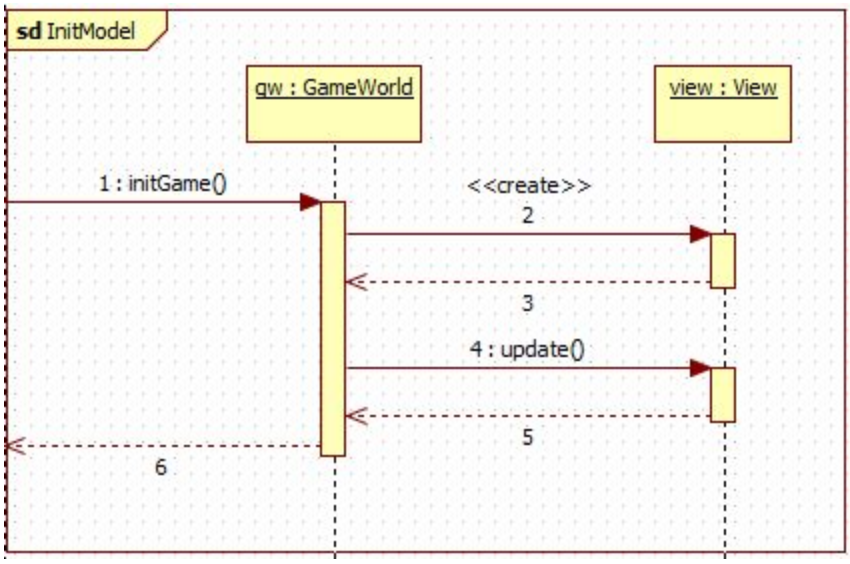
- **Interfészek**

  Observer

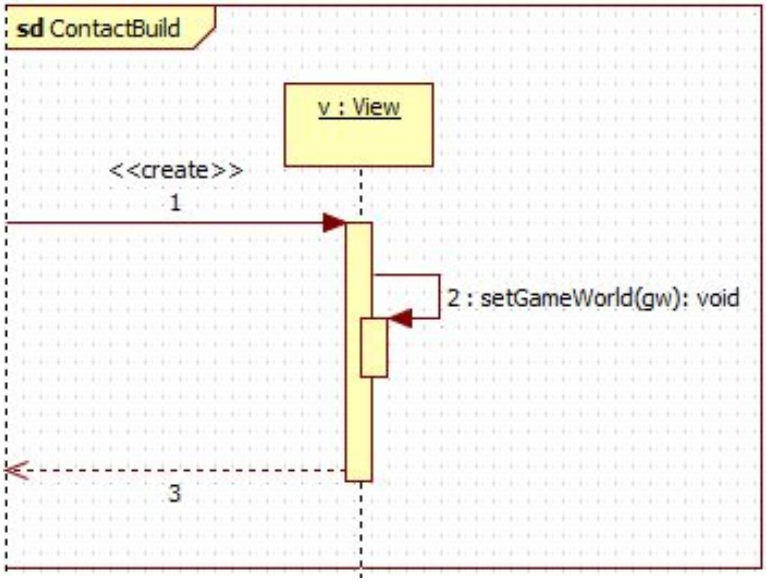
- **Metódusok**
  - **+void notify():** A jégtáblája segítségével megszerzi a GameWorld objektumot, a view attribútum megszerzése után, meghívja az update() függvényt, amivel frissül a grafikus megjelenítés.

11.4 Kapcsolat az alkalmazói rendszerrel

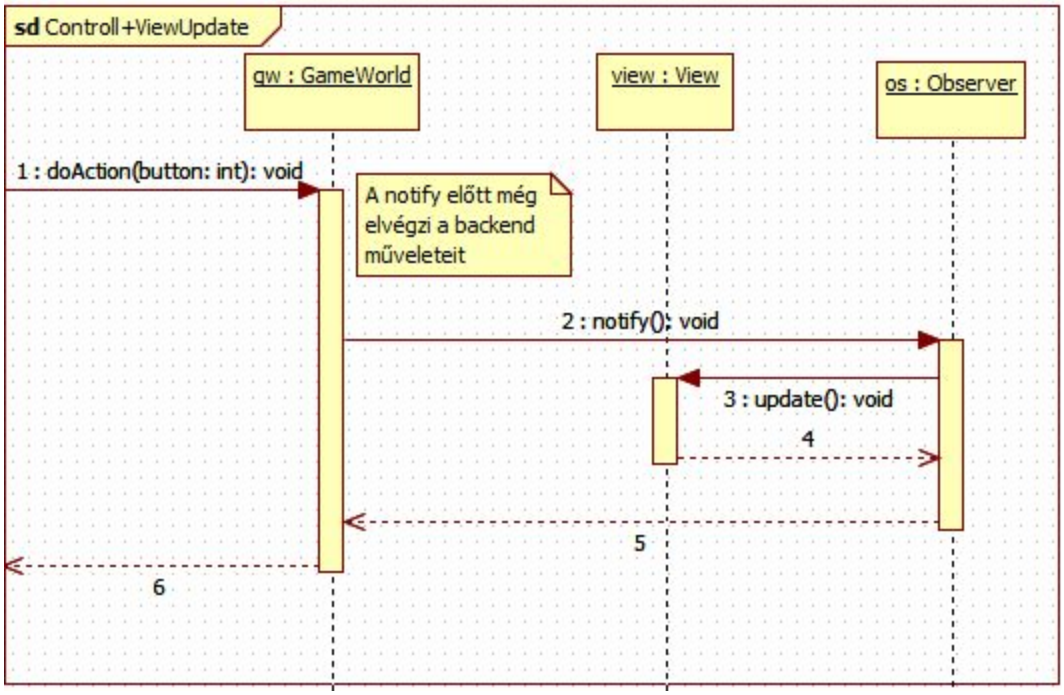
11.4.1 InitModel (modell felépítése)

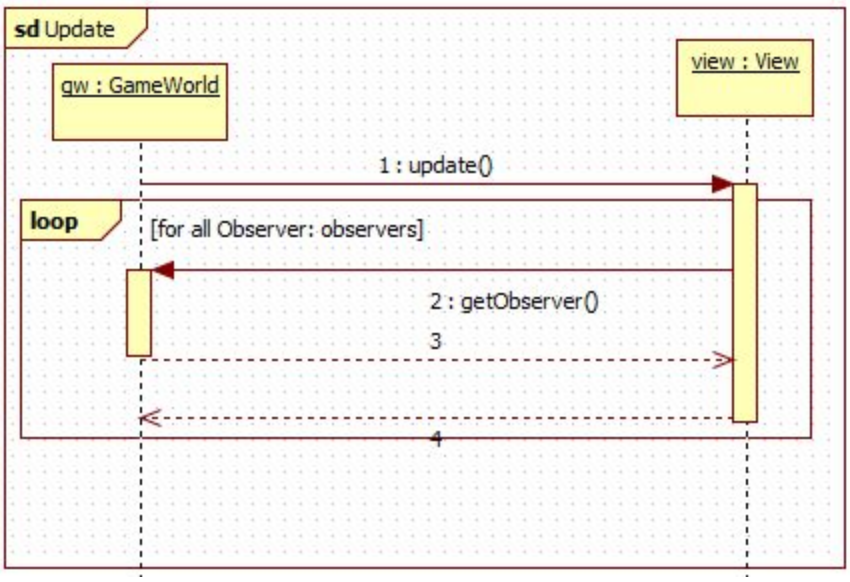


11.4.2 A Model és a Nézet közötti kapcsolat beállítása



11.4.3 A Játék vezérlése és a hozzá kapcsolódó megjelenítés





11.5 Napló

Kezdet	Időtartam	Résztvevők	Leírás
2020.05.01. 17:00	2 óra	Csoma Sipula Szepesi-Nagy Varga Ziaja	Értekezlet, alapok átbeszélése, tervezés elkezdése.
2020.05.02. 18:00	3 óra	Csoma Sipula Szepesi-Nagy Varga Ziaja	Konzulenstől kapott válaszok megbeszélése, tervezés folytatása.
2020.05.03. 15:30	3 óra	Csoma Sipula Szepesi-Nagy Varga Ziaja	Szekvenciadiagramok szerkesztése, dokumentum véglegsítése, javítások/módosítások megbeszélése.

13. Grafikus változat beadása

13.0 Változások az előző beadáshoz és a proto-hoz képest

- Observer interfész nem került bevezetésre
- A protoban is látott módon a parancsok hasonlóan kerülnek átadásra a gombok lenyomása szerint, így a javaFX és a program között nem keletkezett deadlock.
- Lépésnél jégtábla indexe kerül megadásra, nem irány radiánban

13.1.1 Fájllista

Fájl neve	Méret (Byte)	Keletkezés ideje	Tartalom
backpack.png	118913	2020. május 17.	Táska megnyitásánál megjelenő ikon.
bcsv_11.jar	8779597	2020. május 17.	A programot lefordítva tartalmazó jar fájl.
BreakableShovel.java	1361	2020. május 17.	A törékeny ásó megvalósítása.
Camp.java	746	2020. május 17.	A sátor megvalósítása.
Character.java	8472	2020. május 17.	A karakterek ősosztálya, közös függvények megvalósítása.
Eskimo.java	713	2020. május 17.	A játékban szereplő egyik karaktertípust, az eszkimó és annak speciális képességének (iglu építés) megvalósítása.
eszkimo.jpg	366509	2020. május 17.	Egy kép, ami az adatok beadásakor jelenik meg.
Food.java	494	2020. május 17.	Az étel használatának megvalósítása.
gamestate.txt	6606	2020. május 17.	Kimentett játékállapot
GameWorld.java	26189	2020. május 17.	A játékkezelő megvalósítása.
Gun.java	1085	2020. május 17.	A fegyver megvalósítása, ami a jelzőpisztoly első alkatrésze.
IceTable.java	8529	2020. május 17.	Egy jégmező és annak függvényeinek megvalósítva.
igloo.png	205857	2020. május 17.	Egy kép, ami az ablak ikonjaként jelenik meg.
Iglu.java	1631	2020. május 17.	Az iglu megvalósítása.
Item.java	395	2020. május 17.	Az eszközök interfésze.
ItemOwner.java	704	2020. május 17.	Az IceTable és Character absztrakt őse. Eszközkezelést valósít meg.
leiras.txt	1971	2020. május 17.	Játékleírás, Info fül tartalma.
Light.java	530	2020. május 17.	A jelzőpisztoly harmadik alkatrésze.

Main.java	3577	2020. május 17.	A program fő osztálya, ahol a menü került megvalósításra.
matrix01	988	2020. május 17.	Első pálya szomszédsági mátrixa
matrix02	988	2020. május 17.	Második pálya szomszédsági mátrixa
PolarBear.java	2464	2020. május 17.	A jegesmedve megvalósítása.
Researcher.java	816	2020. május 17.	A játékban szereplő egyik karaktertípust, a sarkkutató és annak speciális képességének megvalósítása.
Rope.java	823	2020. május 17.	A kötelet megvalósítása.
RunAntarctica.bat	245	2020. május 17.	A program jar fájljának futtatásáért felelős bat fájl.
sarkkutatok.jpg	101796	2020. május 17.	Egy két, ami az adatok bekérésénél kerül megjleneítésre.
Shovel.java	1224	2020. május 17.	Az ásó megvalósítása.
Stencil.java	1136	2020. május 17.	A patron megvalósítása.
step.png	5930	2020. május 17.	A lépésnél megjelenő ikon.
Stepable.java	1630	2020. május 17.	A lépő elemek ősosztálya.
Surprise.java	1645	2020. május 17.	A meglepetés megvalósítsa.
Swimsuit.java	512	2020. május 17.	Az úszóruha megvalósítása.
View.java	38520	2020. május 17.	A nézetet (grafikus felületet megvalósító osztály.)

13.1.2 Fordítás és telepítés

Fontos: A következő fordítási és futtatási útmutató egy olyan útmutatót tartalmaz, amivel a programot fordítani és futtatni lehetne, viszont a cloud-os virtuális gépen ez nem mindig működik teljesen jól a javaFX miatt, így kérünk, hogy a kicsomagolás után a mellékelt ***RunAntarctica.bat*** fájl segítségével futtasd a játékot, ami egy olyan jar fájlt futtat, melyben minden Java 11 vagy fölötti rendszeren kompatibilis a javaFX-es futtatás.

Letöltés és kicsomagolás után az src mappában nyitott parancssor segítségével tudjuk fordítani a programot.
Beállítjuk először a javaFX elérési útját a *set FX_PATH="%cd%\javafx-sdk-14.0.1\lib"* parancs segítségével.
Mivel a fenti listában szereplő java fájlok különböző package-ben helyezkednek el, ezért ahhoz, hogy ezeket egyszerre tudjuk fordítani össze kell gyűjteni őket egy fájlba, amit a *dir /s /B *java > javas.txt* a parancs segítségével tudunk megtenni.
Miután ezek megvannak, ki kell adni a *javac --module-path %FX_PATH% --add-modules javafx.controls @javas.txt* parancsot, hogy a java fájlok fordításra kerüljenek.

Ezután a *java --module-path %FX_PATH% --add-modules javafx.controls game.Main* segítségével tudjuk elindítani.

13.1.3 Futtatás

Futtassuk a programot a ***RunAntarctica.bat*** fájl segítségével, ahol meg fognak jelenni a játékosok számát majd nevét bekérő ablakok. Itt minimum három játékost meg kell adnunk, mint az korábban a specifikációban meglett adva.
Javaslat: Adjunk meg fejenként 2-2 vagy 3-3 játékost, hogy ne kelljen annyi nevet beadni a következő lépésben.
Miután ezeket megadtuk, meg is fog jelenni a pálya és kezdődhet is a játék.

13.2 Értékelés

Tag neve	Tag neptun	Munka százalékban
Csoma Zoltán	G0IOFT	20
Sipula László	A1D4QD	20
Szepesi-Nagy István	K45SFS	20
Varga Zsombor	AKCJOP	20
Ziaja Bálint	HICYBO	20

13.3 Napló

Kezdet	Időtartam	Résztvevők	Leírás
2020.05.06. 14:00	3 óra	Csoma Szepesi-Nagy Sipula Varga Ziaja	Értekezlet. Fontosabb újítások megbeszélése.
2020.05.06. 20:00	2 óra	Szepesi-Nagy	Grafikus felület programozásának megkezdése, osztályok bevezetése.
2020.05.07. 12:00	1 óra	Szepesi-Nagy Sipula	javaFX telepítése a programhoz, alapok bevezetése a programba.
2020.05.08. 14:00	2 óra	Szepesi-Nagy	Fő ablak megszerkesztése, indítási utáni bekérés elkészítése.
2020.05.09. 15:00	3 óra	Varga	Grafikus felület és program közötti szálkezelés és kommunikáció kijavítása. Gombok alapjainak az elkészítése.
2020.05.10. 13:00	2 óra	Szepesi-Nagy Ziaja	Grafikus felület bővítése.
2020.05.11. 16:00	2 óra	Szepesi-Nagy Varga	Debuggolás, meglévő kód módosítása
2020.05.13 18:00	2 óra	Sipula Szepesi-Nagy Varga	Program átnézése, kisebb változtatások
2020.05.15 12:00	3 óra	Sipula Szepesi-Nagy	Cloud-os tesztelés, új részek és régi változásainak kommentelése.

2020.05.16 14:00	4 óra	Sipula Varga	Dokumentációk elkészítése. Összes doksi összefűzése

14. Összefoglalás

14.1 A projektre fordított összes munkaidő

Tag neve	Munkaidő (óra)
Csoma Zoltán	71
Sipula László	100
Szepesi-Nagy István	88
Varga Zsombor	116
Ziaja Bálint	82
Összesen	457

14.2 A feltöltött programok forrássorainak száma

Fázis	Kódsorok száma
Szkeleton	1176
Prototípus	2738
Grafikus változat	3247
Összesen	7161

14.3 • Projekt összegzés

14.3.1 Mit tanultak a projektből konkrétan és általában?

Általában: Jó tervezés megkönnyíti a kódolást.
Konkrétan: Elég nehéz csapatban dolgozni, főleg, ha egyes csapattagok kezét fogni kell munka közben vagy részletes átnézést igényel az általuk végzett munka.

14.3.2 Mi volt a legnehezebb és a legkönnyebb?

Legnehezebb: Összeegyeztetni, hogy kinek mikor jó(főleg ha valakinek soha se jó).
Legkönnyebb: Ha más munkájára nem számítva megírod azt a részt is inkább.

14.3.3 Összhangban állt-e az idő és a pontszám az elvégzendő feladatokkal?

A nagy mérföldköveken (Szkeleton, Proto és Grafikus) belül nagyjából igen, viszont az adott részek súlyozása egyáltalán nem. A tervezéssel több munka van, mint mondjuk a protoval.

14.3.4 Ha nem, akkor hol okozott ez nehézséget?

A jelenlegi félévben kialakult különleges helyzetre való tekintettel olyan “jelentéktelen” feladatrész vett el időt egy hosszabb feladatrésztől.

14.3.5 Milyen változtatási javaslatuk van?

A fejenként kapott összesített óraszám nem teljesen tükrözi a a tényleges belefektetett időt, mert nem mindig figyeli az ember, hogy mennyi is telt el igazán és emlékezetből nehéz pontosan megadni. Egyes embereknél ez bőven több lehetett(volt). Így a 3 kredit nem igazán tükrözi a belefektetett energiát és időt.

A konzultációs lehetőségeket időben át lehetne alakítani, hogy ténylegesen konzultációk legyenek, hogy még az adott rész beadása előtt lehessen kérdezni és hibákat feltárni a konzulenssel élőben.

14.3.6 Milyen feladatot ajánlanának a projektre?

Nincs ilyen.

14.3.7 Egyéb kritika és javaslat

Az alapozó tárgy (Szoftvertechnológiák) itt használandó részeinek részletesebb megismerése és gyakorlása, mint pl. Kommunikációs diagramok.