6. Szkeleton beadása

25 – *bandITs*

Konzulens:
Huszerl Gábor

Csapattagok

Bencze János István	GIWUHT	gomanpc@yahoo.com	
Guzmics Gergő	VC8OQD	guzmicsgergo@gmail.com	
Kohár Zsombor	Q8EPW6	zsombor.kohar@edu.bme.hu	
Rakos Gergő Máté	I3Q7BY	gergo_rakos@yahoo.com	
Dr. Taba Szabolcs Sándor	JRGMBW	taba.szabolcs@gmail.com	

2025.03.24.

6. Szkeleton beadása

6.1 Fordítási és futtatási útmutató

6.1.1 Fájllista

Fájl neve	Méret	Keletkezés ideje	Tartalom
AridTecton.java	4 KB	2025. 03. 19. 21:27	AridTecton osztály implementációja.
FertileTecton.java	6 KB	2025. 03. 19. 21:27	FertileTecton osztály implementációja.
Insect.java	9 KB	2025. 03. 19. 14:01	Insect osztály implementációja.
InsectState.java	1 KB	2025. 03. 19. 14:01	FertileTecton osztály implementációja.
Main.java	64 KB	2025. 03. 19. 21:27	A tesztesetek és azok tesztelővel való kommunikációjának implementációja
MultiLayeredTecton.java	2 KB	2025. 03. 19. 21:27	MultiLayeredTecton osztály implementációja.
Mushroom.java	1 KB	2025. 03. 19. 14:01	Mushroom osztály implementációja.
MushroomBody.java	7 KB	2025. 03. 19. 14:01	MushroomBody osztály implementációja.
MushroomBodyGrowthEvaluator.java	4 KB	2025. 03. 19. 21:27	MushroomBodyGrowthEvaluator osztály implementációja.
Mycelium.java	6 KB	2025. 03. 19. 14:01	Mycelium osztály implementációja.
MyceliumGrowthEvaluator.java	3 KB	2025. 03. 19. 21:27	MyceliumGrowthEvaluator osztály implementációja.
OnRoundBeginSubscriber.java	1 KB	2025. 03. 19. 14:01	OnRoundBeginSubscriber osztály implementációja.
OnTurnBeginSubscriber.java	1 KB	2025. 03. 19. 14:01	OnTurnBeginSubscriber osztály implementációja.
PreventCutSpore.java	1 KB	2025. 03. 19. 14:01	PreventCutSpore osztály implementációja.
SemiFertileTecton.java	5 KB	2025. 03. 19. 21:27	SemiFertileTecton osztály implementációja.
SlownessSpore.java	1 KB	2025. 03. 19. 14:01	SlownessSpore osztály implementációja.
SpeedSpore.java	1 KB	2025. 03. 19. 14:01	SpeedSpore osztály implementációja.
Spore,java	1 KB	2025. 03. 19. 14:01	Spore osztály implementációja.
StunSpore.java	1 KB	2025. 03. 19. 14:01	StunSpore osztály implementációja.
Tecton.java	12 KB	2025. 03. 19. 14:01	Tecton osztály implementációja.

TectonVisitor.java	2 KB	2025. 03. 19. 21:27	TectonVisitor osztály implementációja.

6.1.2 Fordítás

A programot a felhasználó úgy tudja lefordítani (feltéve, hogy más java forrásfájlok nem szerepelnek azonos mappában és minden forrásfájl azonos mappában van), hogy kiadja az operációs rendszer terminálján a „javac *.java” parancsot, a forrásfájlok mappájában.

6.1.3 Futtatás

A programot a felhasználó úgy tudja futtatni, hogyha lefordítja, majd kiadja az operációs rendszer terminálján a „java Main” parancsot.

6.2 Értékelés

Tag neve	Tag neptun	Munka százalékban
Bencze János István	GIWUHT	20%
Guzmics Gergő	VC8OQD	20%
Kohár Zsombor	Q8EPW6	20%
Rakos Gergő Máté	I3Q7BY	20%
dr. Taba Szabolcs Sándor	JRGMBW	20%

6.3 Napló

Kezdet	Időtartam	Résztvevők	Leírás
2025.03.18 ., 19:00	30 perc	Bencze Guzmics Kohár Rakos Taba	Értekezlet. Döntés: - Átbeszélése a jövőbeli terveknek és hogy mi a következő feladat lényege
2025.03.19 ., 14:00	1 óra	Bencze Guzmics Kohár Rakos Taba	Értekezlet. Döntések: - Átbeszélése a konzultáción elhangzottaknak - Utolsó öt use-case teljes átdolgozásának eldöntése - Rakos és Kohár kódolási résszel fog foglalkozni - Guzmics, Taba kijavitja es ki bővíti a use-case diagrammokat és szöveget - Bencze kijavítja a saját múlt heti munkáit majd ő is kódol
2025.03.19 ., 16:00	2 óra	Kohár	Tevékenység: - Osztályok és függvények üres implementációja
2025.03.20 ., 18:00	3 óra	Bencze	Tevékenység: - Bencze múlt heti use-case-jeinek teljes átdolgozása, kibővítése és leírása
2025.03.20 ., 19:00	1 óra	Kohár	Tevékenység: - Test case IO interfész implementálása
2025.03.20 ., 20:00	1 óra	Guzmics Taba	Értekezlet. Döntések: - A szkeleton tevezés feladathoz use-case-

			ek javításának előkészítése
2025.03.20 ., 22:00	1 óra 30 perc	Guzmics	Tevékenység: - Saját use-case-ek kijegyzetelése
2025.03.21 ., 14:00	2 óra	Taba	Tevékenység: - Az 1-8. use case-ek frissítése
2025.03.21 ., 18:00	1 óra 30 perc	Bencze Guzmics Kohár Rakos Taba	Értekezlet. Döntések: - Bencze által újra dolgozott use-casek átnézése
2025.03.21 ., 20:00	1 óra 30 perc	Kohár	Tevékenység: - Gombafonál növesztéssel és gombaspóra evéssel kapcsolatos test case-ek implementálása
2025.03.21 21:00	3 óra	Rakos	Tevékenység: - Gombatest növesztéssel kapcsolatos test case-ek implementálása
2025.03.21 ., 20:30	2 óra	Guzmics	Tevékenység: - Use-case-ek egyeztetése és iterálása
2025.03.22 ., 8:30	3 óra	Guzmics	Tevékenység: - Use-case-ek iterálása
2025.03.22 ., 13:00	5 óra	Taba	Tevékenység: - Az 1-8. és a 31-35. sz. use case-ek frissítése, valamint a leadandó anyag Függelék részének előkészítése
2025.03.22 ., 16:00	2 óra	Bencze	Tevékenység: - InsectMove és InsectCut tesztjeinek lekódolása
2025.03.22 ., 18:00	2 óra	Bencze Guzmics Kohár Rakos Taba	Értekezlet. Döntések: - Átbeszélni a hiányosságokat a use-casekben

			- Formalizálni a kiírási nyelvet
2025.03.22 ., 20:30	1 óra 30 perc	Kohár	Tevékenység: - Mycelium komponensek törléséért felelős és távolság meghatározásáért felelős a függvények programozása
2025.03.22 ., 21:00	1 óra	Bencze	Tevékenység: - mushroomBody ejectSpore 4 teszt lekódolása
2025.03.22 ., 21.30	1 óra 30 perc	Rakos	Tevékenység: - Gombatest kódolásával való hibák javítása
2025.03.22 ., 22:00	1 óra	Bencze	Tevékenység: - Saját szekvencia diagramok véglegesítése, átnevezése és megszámozása
2025.03.23 ., 9:00	2 óra 30 perc	Guzmics	Tevékenység: - Use-case-ek iterálása és véglegesítése
2025.03.23 ., 10:00	1 óra 30 perc	Rakos	Tevékenység: - Tekton törés lekódolása
2025.03.23 ., 12:00	3 óra	Taba	Tevékenység: - Az 1-8. és a 31-35. sz. use case-ek véglegesítése, valamint a leadandó anzag Függelék részének előkészítése
2025.03.23 ., 15:00	1 óra 45 perc	Bencze Guzmics Kohár Rakos Taba	Értekezlet. Döntések: - Összehasonlítani a programkódot a leírt dolgokkal - Hibák és pontatlanságok átbeszélése
2025.03.23 ., 17:00	1 óra 30 perc	Rakos	Tevékenység: - Rakos által írt kód

			dokumentálása és egyes hiányos dokumentálások pótlása
2025.03.23 ., 17:00	1 óra	Guzmics	Tevékenység: - Use-case-ek formai egyeztetése
2025.03.23 ., 17:00	3 óra	Taba	Tevékenység: - A leadandó anyag Függelék részének előkészítése
2025.03.23 ., 21:00	3 óra	Taba	Tevékenység: - A leadandó anyag függelék részének véglegesítése
2025.03.23 ., 22:00	2 óra	Kohár	Tevékenység: - Végső ellenőrzés, hibajavítás

7. Prototípus koncepciója

25 – bandITs

Konzulens:
Huszerl Gábor

Csapattagok

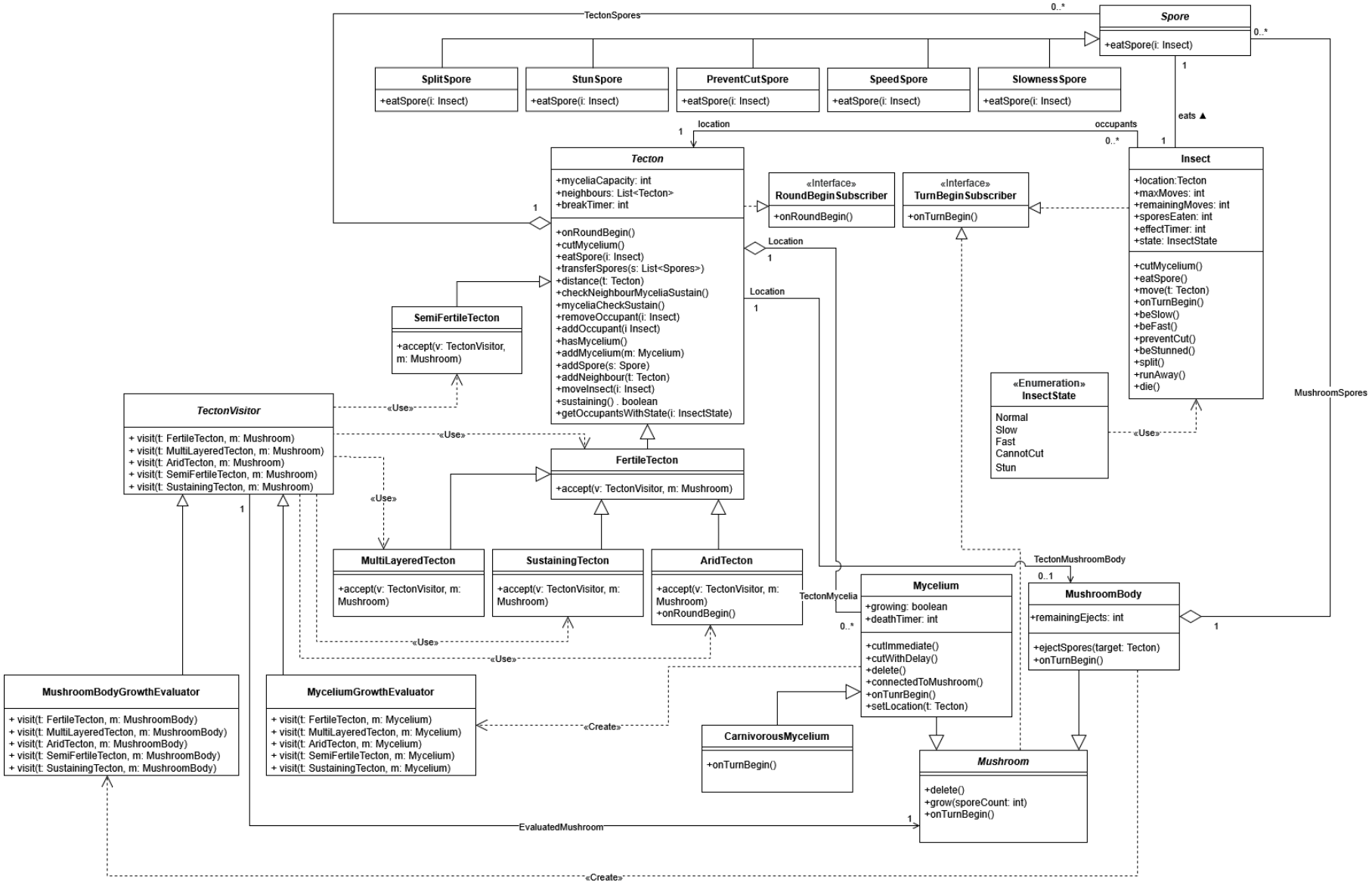
Bencze János István	GIWUHT	gomanpc@yahoo.com
Guzmics Gergő	VC8OQD	guzmicsgergo@gmail.com
Kohár Zsombor	Q8EPW6	zsombor.kohar@edu.bme.hu
Rakos Gergő Máté	I3Q7BY	gergo_rakos@yahoo.com
Dr. Taba Szabolcs Sándor	JRGMBW	taba.szabolcs@gmail.com

2025.03.31.

Prototípus koncepciója

7.0 Változás hatása a modell

7.0.1 Módosult osztálydiagram



7.0.2 Új vagy megváltozó metódusok

7.0.2.1 Mycelium – cutImmidiate()

Azonnal elvágja a gombafonalat.

7.0.2.2 Mycelium – cutWithDelay()

Egy gombafonál típustól függő idő után elvágja a gombafonalat.

7.0.2.3 Tecton - getOccupantsWithState(i: InsectState)

Azon rovaroknak a listáját hozza létre, amelyek a tektonon helyezkednek el, és adott állapotban vannak.

7.0.2.4 Tecton - sustaining()

Megadja, hogy a tekton, vagy a tektonon elhelyezkedő objektumok képesek-e egy gombafonalat életben tartani.

7.0.2.5 TectonVisitor - visit(t: SustainingTecton, m: Mushroom)

A visitor működésének kiterjesztése az új tekton fajtára.

7.0.2.6 Insect – die()

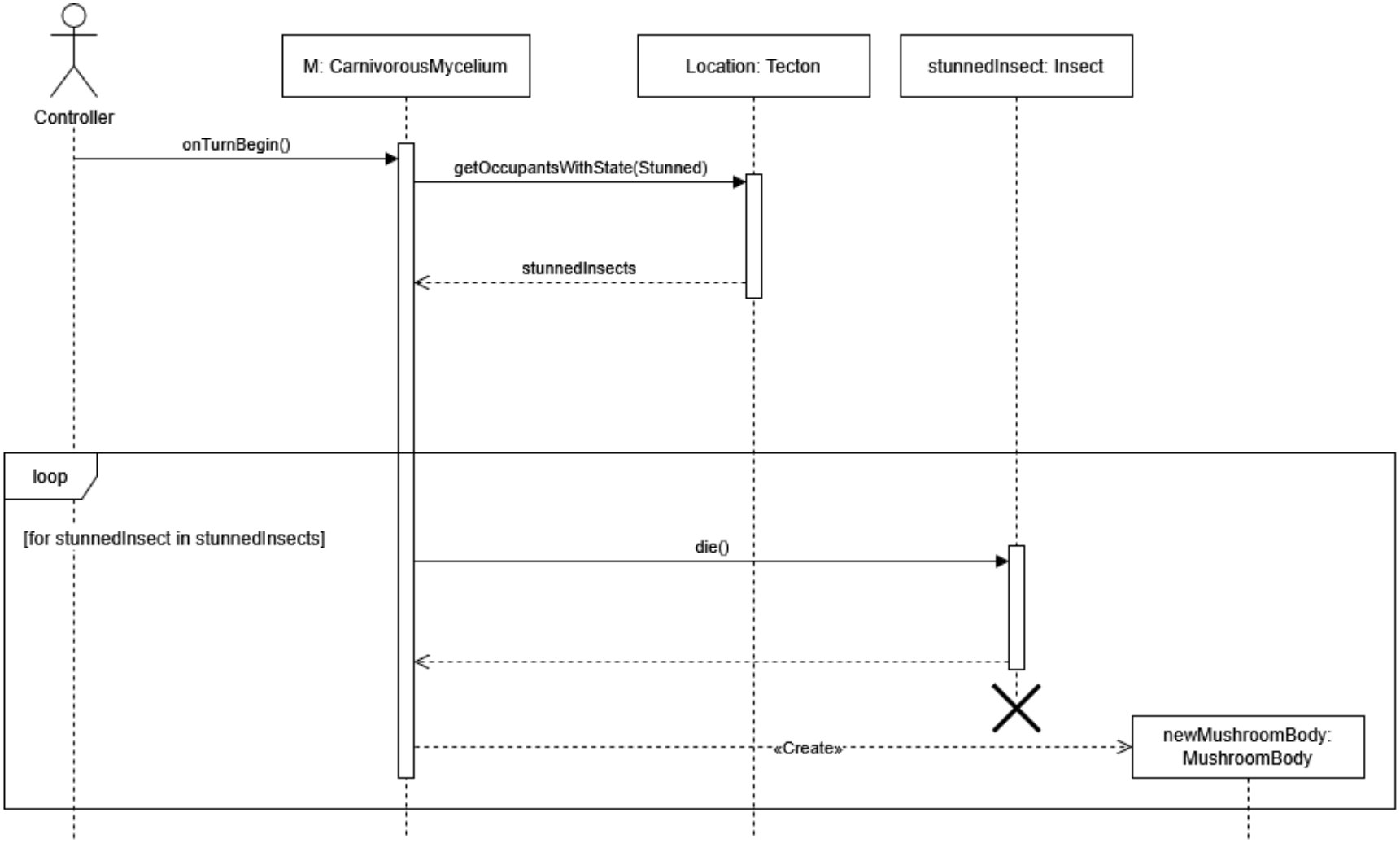
A rovar elpusztul, azaz eltűnik a játékmezőről.

7.0.2.7 Insect – split()

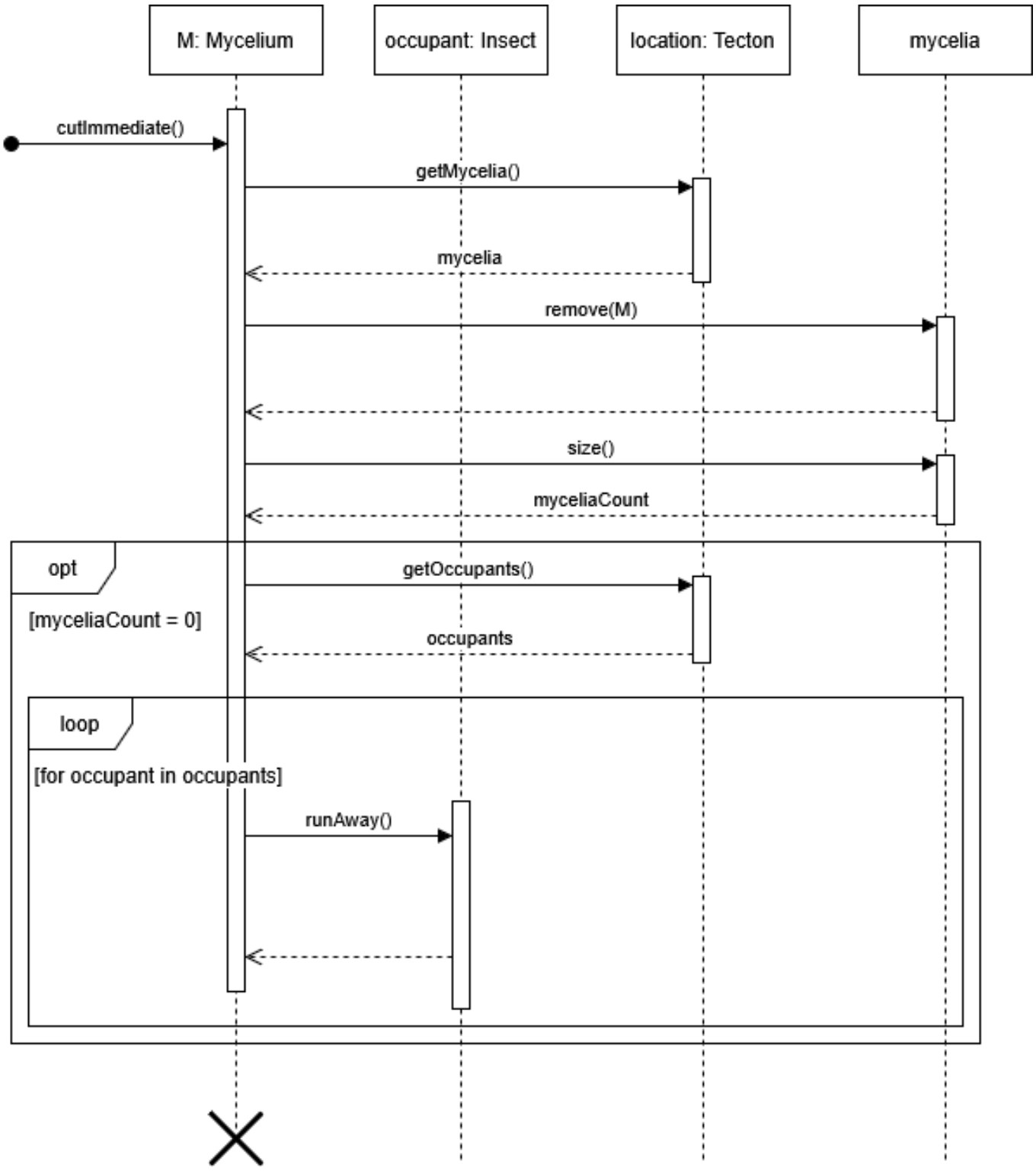
A rovar csinál egy másolatot saját magából.

7.0.3 Szekvencia-diagramok

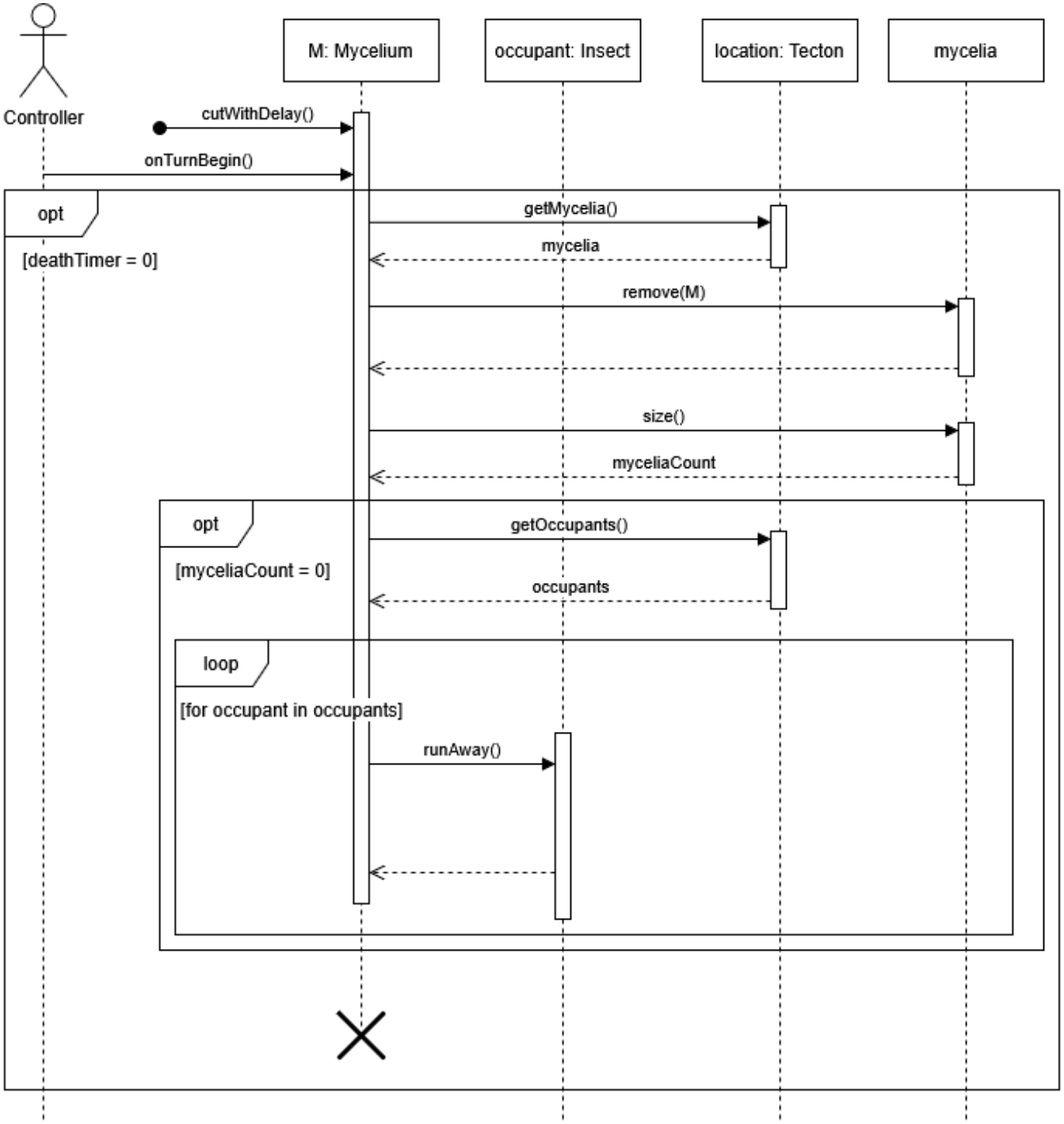
7.0.3.1 Húsevő gombafonál



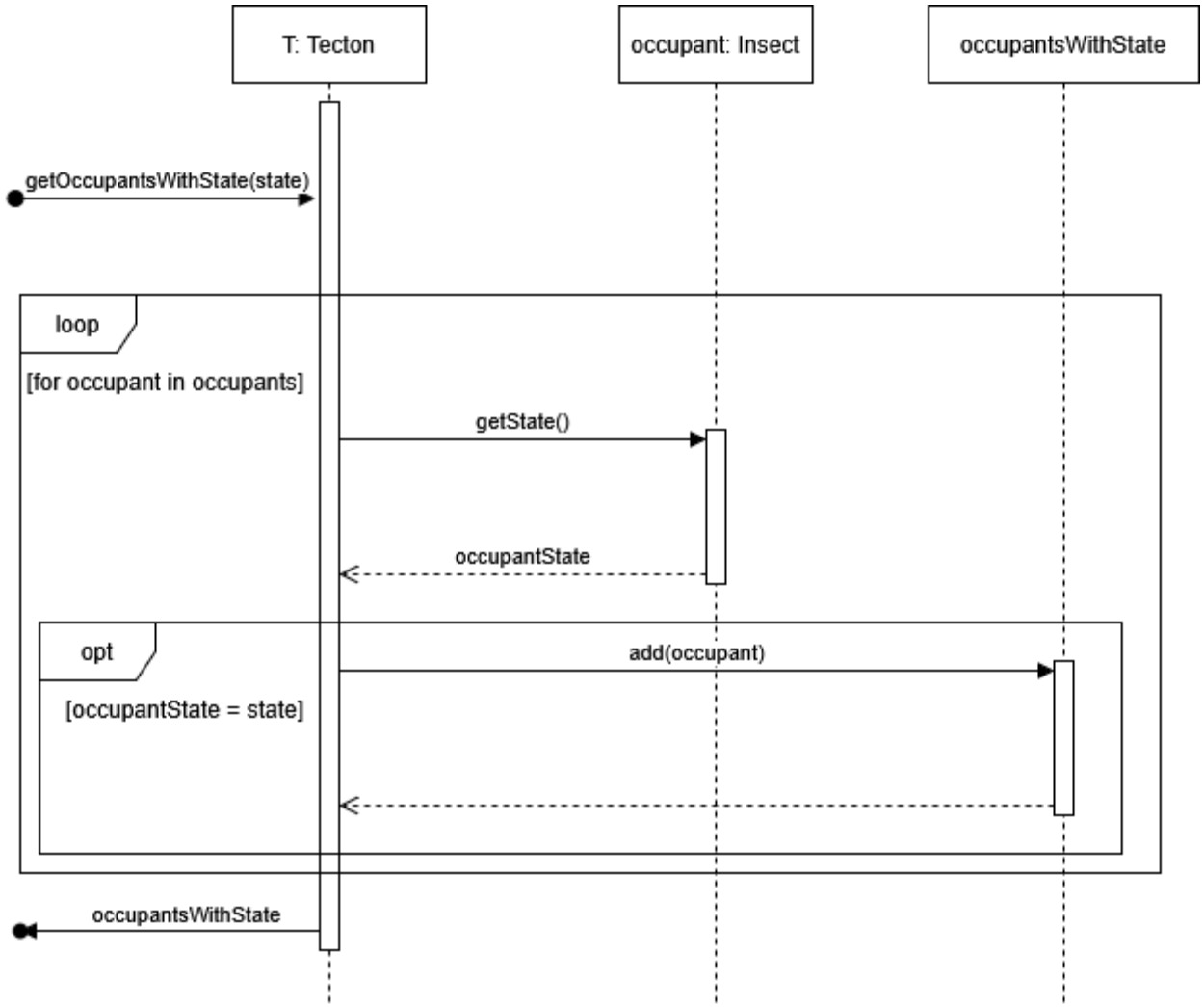
7.0.3.2 Azonnali fonálelvágás



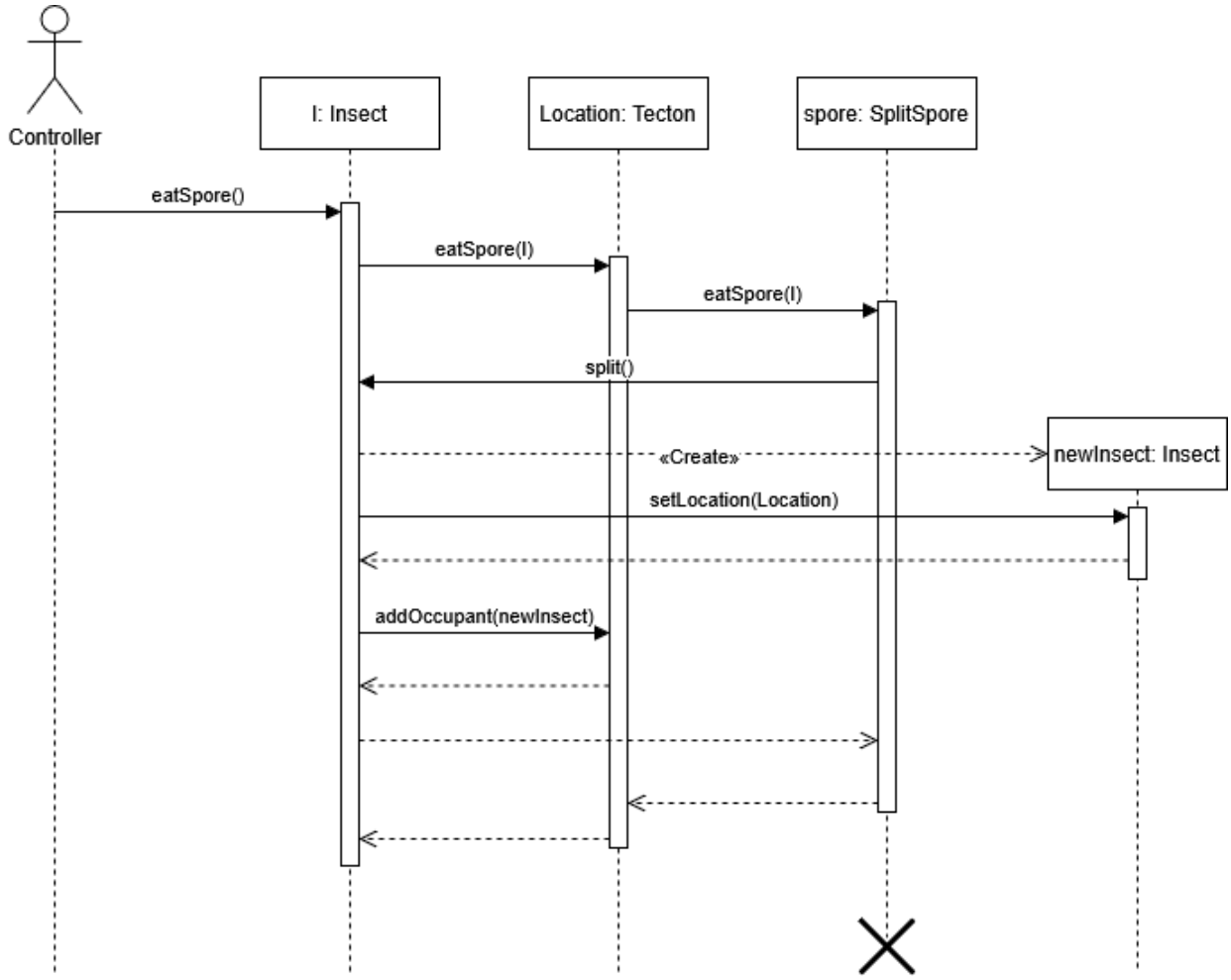
7.0.3.3 Késeltetett fonal elvágás



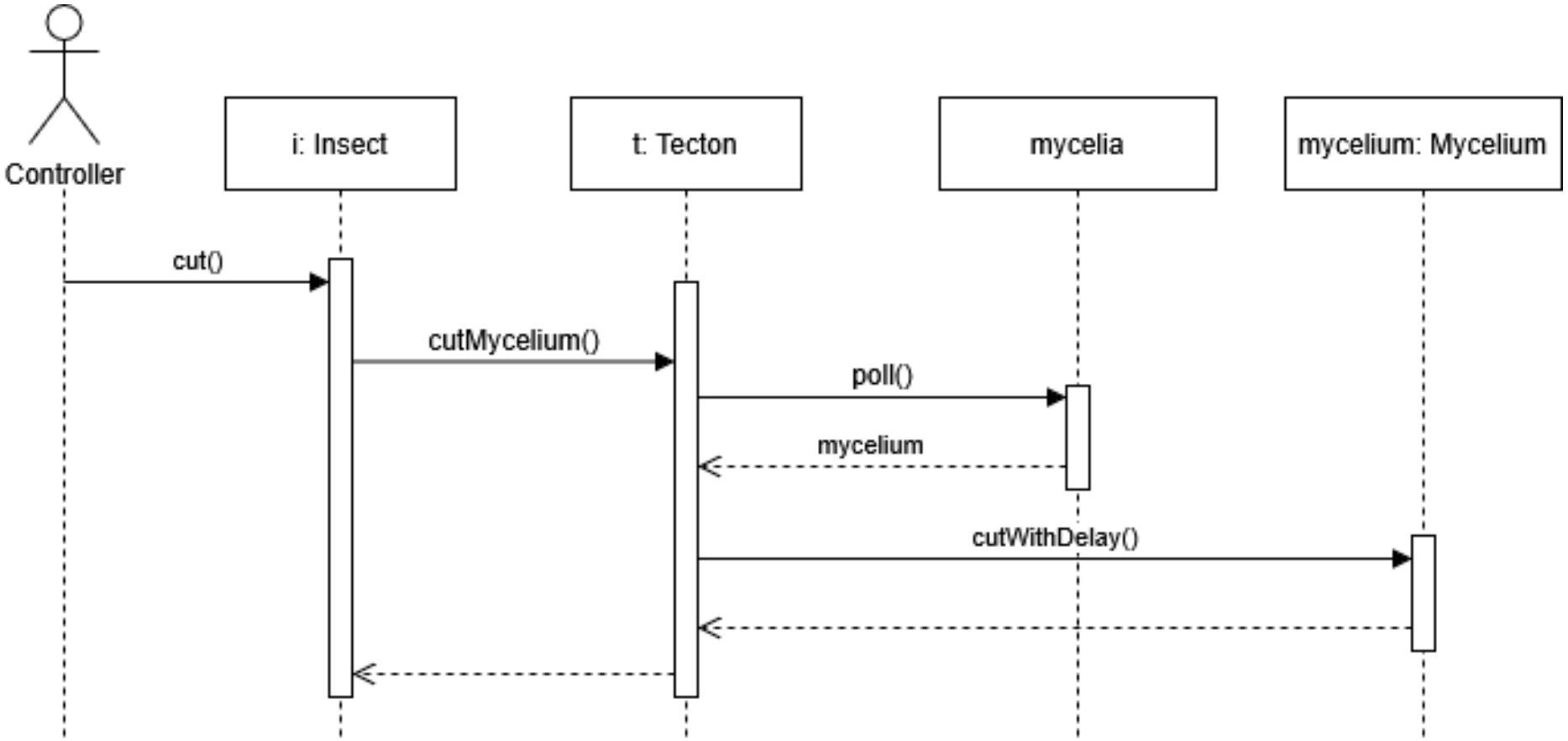
7.0.3.4 getOccupantsWithState szekvencia



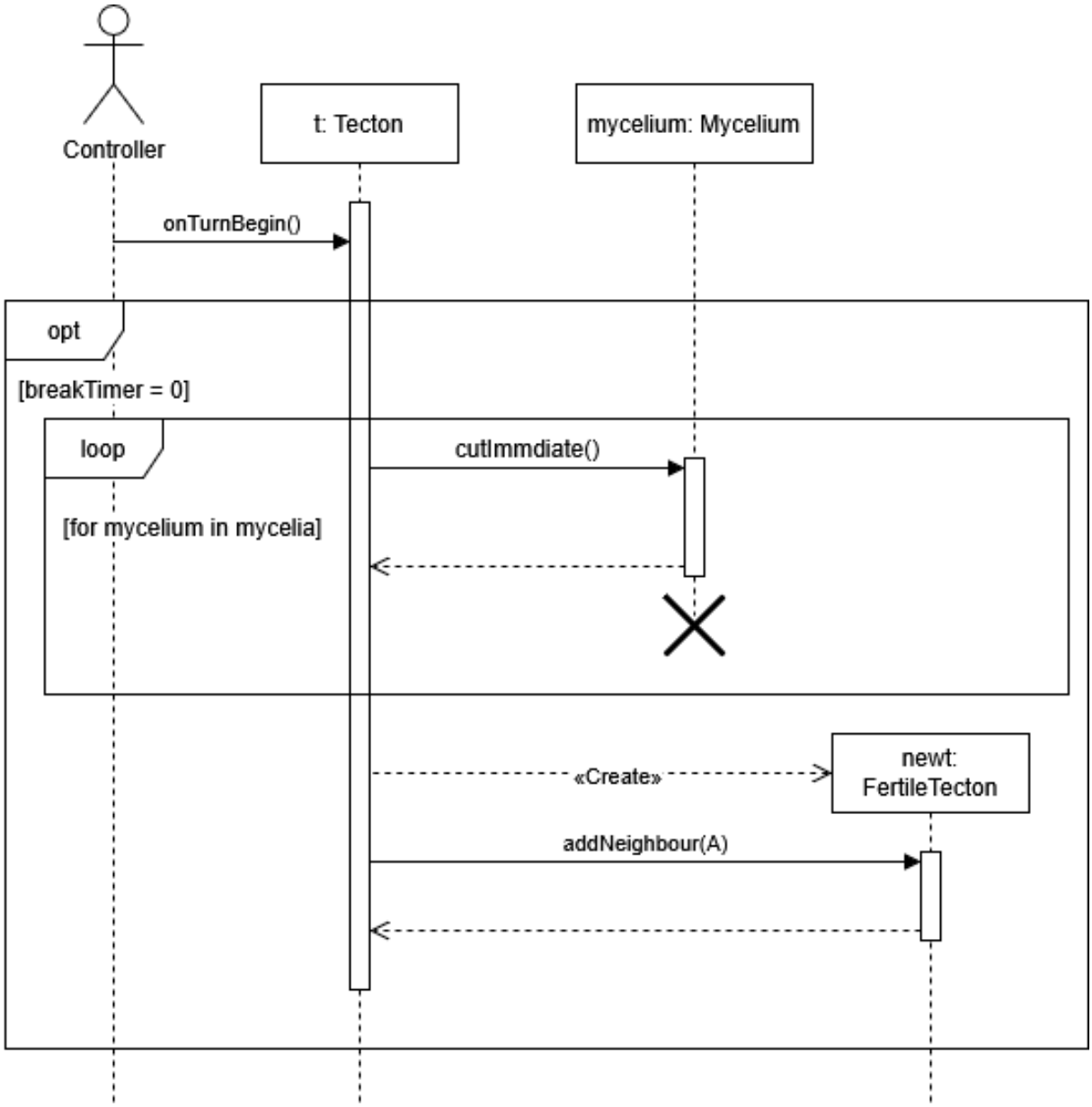
7.0.3.5 „Split” típusú spóra elfogyasztása



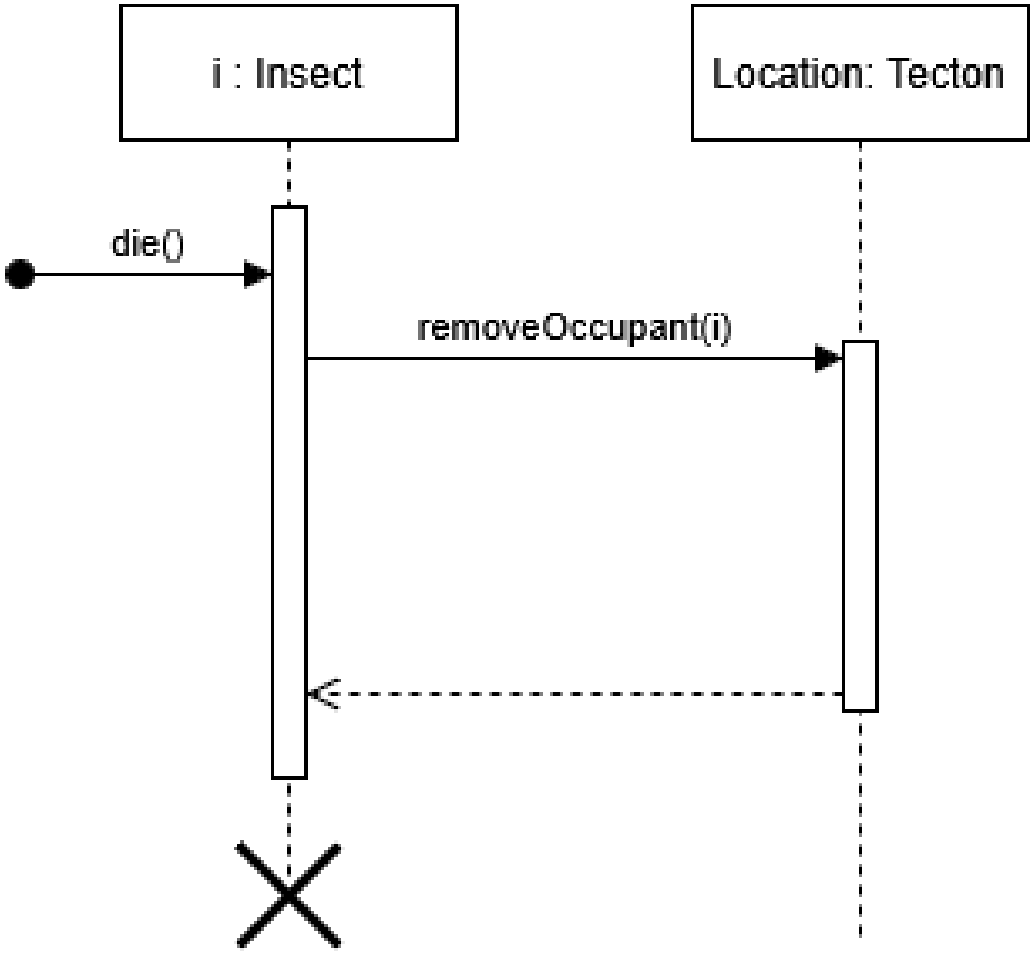
7.0.3.6 Rovar általi gombafonál elvágás



7.0.3.7 Tektontörés



7.0.3.8 Rovar elpusztulása



7.1 Prototípus interface-definíciója

7.1.1 Az interfész általános leírása

A felhasználó a program irányítását parancsokkal végzi. Ezeknek a parancsoknak vannak argumentumai, melyeket a parancs után tud a játékos beleírni. A parancsok maguktól nem adnak outputot. Ha egy parancs egy részében felhasználói választásra van szükség, akkor a program ki tud kérdezni a felhasználóhoz. A program futása során minden objektumnak van egy saját, egyedi neve, ami alapján az objektum egyértelműen azonosítható.

7.1.2 Bemeneti nyelv

Egy parancs általános szintaxisa:
<UTASÍTÁS> <PARAMÉTER> {<LISTA_ELEM_1> <LISTA_ELEM_2> <LISTA_ELEM_N>}

Tehát a parancsnak megadjuk a nevét, (utasítás), majd utána, ha szükséges, szóközzel elválasztva a parancs paramétereit megadhatjuk. Ha a parancs paramétere egy gyűjtemény lenne, akkor a gyűjtemény kezdetét és végét kapcsos zárójellel jelöljük. Az egyes elemek szóközzel vannak elválasztva.

Ha egy parancsnak felhasználói inputra van szüksége, akkor azt a program az alábbi módon jelzi:
<< 1:<OPCIÓ_1> 2:<OPCIÓ_2> N:<OPCIÓ_N>

Azaz, a „<<” karakterekkel jelöli, hogy éppen bemenetre van szükség, az opciókat pedig a sorszámukkal együtt kiírja a program. A felhasználó választani úgy tud, hogyha a választott opció sorszámot megadja.

Minden a MACRO_CASE elnevezési konvenciót használja, azaz minden betű nagybetű, és a szavak alsókötőjel (_) karakterrel vannak elválasztva.

A PARANCSOK LISTÁJA

Az 1. számú use-case kapcsán

STATE objektum

> **Leírás:** A kiválasztott objektum állapotát a megadott formátumban kiírja.
> **Opciók:** A kiírandó objektum.

A 2. számú use-case kapcsán

BREAK_TECTON tecton

> **Leírás:** Tectontörés
> **Opciók:** A tekton ami el fog törni.

A 3. számú use-case kapcsán

CREATE_TECTON TectonType {Neihgbouring Tectons}

> **Leírás:** Új Tecton létrehozása
> **Opciók:** Első paraméter eldönti milyen típusu legyen az új Tecton, A következő paraméter egy lista mely tartalmazza a Tectonokat amelyek szomszédjai lesznek az új Tectonnak

A 4. számú use-case kapcsán

SET_BREAKTIMER tecton number

> **Leírás:** Tecton BreakTimerjének beallítása egy adott értékre
> **Opciók:** A tekton, melynek az időzítőjét beállítjuk, ésa szám ami be lesz állitva mint új BreakTimer

Az 5. számú use-case kapcsán

RUN text_file

> **Leírás:** Lefuttat egy fájlban lévő parancsokat. Játék inicializálására hasznos.
> **Opciók:** A fájl amit futtatunk.

A 6. számú use-case kapcsán

END_GAME

> **Leírás:** Játék végének kezelése
> **Opciók:** Nincsenek paraméterek

A 7. számú use-case kapcsán

SET_ENDGAMETIMER number

Leírás: Az EndgameTimer beállítása
Opciók: Egyetlen paramétere egy szám mely az új maradék Round-okat jelöli

A 8. számú use-case kapcsán

END_TURN

Leírás: End turn küldése
Opciók: Nincsenek paraméterek

A 9. számú use-case kapcsán

ADD_PLAYER player_name player_type

Leírás: Játékosok hozzáadása a játékhoz
Opciók: Első paraméter a játékos neve, a második paraméter hogy Gombász vagy Rovarász lesz a játékos

A 10. számú use-case kapcsán

START_GAME

Leírás: Játék indítása
Opciók: Nincsenek paraméterek

A 11. számú use-case kapcsán

CREATE_MUSHROOMBODY tecton

Leírás: A gombatest a céltektonon létrejön.

Opciók: Céltekton (tecton)

A 12. számú use-case kapcsán

GROW_MUSHROOMBODY tecton

Leírás: A gombatest létrejön és rákerül a céltektonra.

Opciók: Céltekton (tecton)

A 13. számú use-case kapcsán

PUT_SPORE típus tecton

Leírás: Egy adott típusú spóra rákerül a céltektonra.

Opciók: Céltekton (tecton)

A 14. számú use-case kapcsán

EJECT_SPORES mushroombody tecton

Leírás: A kiválasztott gombatest valamennyi spórája rákerül a céltektonra.

Opciók: Gombatest (mushroombody); céltekton (tecton)

A 15. számú use-case kapcsán

DEACTIVATE mushroombody

Leírás: A kiválasztott gombatest elpusztul (inaktívvá válik).

Opciók: Gombatest (mushroombody)

A 16. számú use-case kapcsán

CREATE_MYCELIUM típus

Leírás: Létrehoz egy gombafonalat
Opciók: A gombafonál típusa.

A 17. számú use-case kapcsán

ADD_MYCELIUM_TO_TECTON Mycelium Tecton

Leírás: Hozzáadja a kiválasztott fonalat a kiválasztott tektonhoz
Opciók: A fonál, ami rajta lesz a tektonon és a tekton, amin lesz a fonál

A 18. számú use-case kapcsán

GROW_MYCELIUM Mushroom Tecton

Leírás: Rá-nő egy gombafonál a kiválasztott tektonra
Opciók: A gombatest vagy gombafonál, amiből növesztünk és a tekton, amin a gombafonál lesz

A 19. számú use-case kapcsán

CREATE_INSECT tecton

Leírás: A rovar létrejön és rákerül az argumentumként megadott céltektonra, ha ezen van gombafonál
Opciók: Argumentumok: A céltekton, ahova létrejönne

A 20. számú use-case kapcsán

MOVE insect tecton

Leírás: A rovar átmenne az argumentumként megadott céltektonra
Opciók: Argumentumok: A rovar, amelyik mozogna. A céltekton, ahova mozogna.

A 21. számú use-case kapcsán

EAT insect

Leírás: A rovar megeszik egy spórát a tektonján
Opciók: Argumentumok: A rovar, amelyik enne

A 22. számú use-case kapcsán

CUT insect

Leírás: A rovar elvág egy fonalat a tektonján
Opciók: Argumentumok: A rovar, amelyik vágna

7.1.3 Kimeneti nyelv

Egy objektum állapotát a STATE paranccsal lehet elérni. A STATE parancsnak és kimenetelének szintaxisa az alábbi:

```
STATE <OBJEKTUM>
<OBJEKTUM> <OBJETUM TÍPUSA>
   <TAGVÁLTOZÓ NEVE> <TAGVÁLTOZÓ TÍPUSA> = <TAGVÁLTOZÓ ÉRTÉKE>
   <LISTA NEVE> <LISTA TÍPUSA> = {
      elem1
      elem2
      elemN
   }
```

Azaz először az objektum nevét, majd típusát írja ki. Majd az adott objektum alá tartozó tagváltozók egy tabulátorral beljebb kerülnek.
Az egyszerű tagváltozókat úgy írjuk ki, hogy először a nevét, majd típusát írjuk ki, majd egy egyenlőségjel után az értékét.
A gyűjtemény típusú tagváltozókat a gyűjtemény neve, utána a gyűjtemény típusa, majd egy egyenlőségjellel elválasztva az értéke, ahol a lista elejét és végét kapcsos zárjelek jelölik, és minden elem neve fel van sorolva, úgy, hogy mindegyik elem külön sorban van.

7.2 Összes részletes use-case

1. SZÁMÚ USE CASE - ÁLLAPOT KIÍRÁSA

Use-case neve	Állapot kiírása
Rövid leírás	A felhasználó a kiválasztott objektum állapotát kiíratja.
Aktorok	User
Forgatókönyv	1. Az objektum kiválasztása 2. Az objektum állapota kiíródik.

2. SZÁMÚ USE CASE - TECTONTÖRÉS

Use-case neve	Tectontörés
Rövid leírás	Az adott FertileTecton eltörik, vagyis a rajta lévő objektumok megsemmisülnek (kivéve a gombatestet) és egy új FertileTecton keletkezik, melynek egyetlen szomszédja az eredeti FertileTecton lesz. Az eredeti FertileTectonnak is beállítjuk az új FertileTectont mint szomszéd. Hasonlóan történik a törés a többi Tecton típusra is. A törés mindig FertileTectont eredményez az új Tecton típusaként.
Aktorok	User
Forgatókönyv	1. A BreakTimer eléri a 0-at 2. A Tectonrol minden eltörlődik (kivéve a gombatestet) 3.Új Tecton keletkezik 4.Az új Tecton és az eltört Tecton szomszédok lesznek

3. SZÁMÚ USE CASE - ÚJ TECTON LÉTREHOZÁSA

Use-case neve	Új Tecton létrehozása
Rövid leírás	A felhasználó paraméterként átadja az összes olyan Tectont amelyeket kívánja hogy szomszédjai legyenek az új Tectonnak. Itt legelső paraméterként azt adja át, hogy milyen típusú Tectonná szeretné csinálni az új Tectont.
Aktorok	User
Forgatókönyv	1. Új Tecton létrehozása az átadott paramétereknek megfelelően 2. Az új Tectonnak beallítódnak a szomszédjai

4. SZÁMÚ USE CASE - TECTON BREAKTIMERJÉNEK BEALLÍTÁSA EGY ADOTT ÉRTÉKRE

Use-case neve	Tecton BreakTimerjének beallítása egy adott értékre
Rövid leírás	A User kiválaszthat egy adott Tectont és annak beállítja a breakTimerjét egy adott értékre.
Aktorok	User
Forgatókönyv	1. A Tecton kiválasztása amelynek módosítanánk a BreakTimerjét 2. A kiválasztott Tecton BreakTimerjének beallítása a paraméterben átadott értékre

5. SZÁMÚ USE CASE - JÁTÉK VILÁG LEGENERÁLÁSA/INICIALIZÁLÁSA

Use-case neve	Játék világ legenerálása/inicializálása
Rövid leírás	Megteremti a kezdeti állapotú Tectonokat, Gombatesteket, Gombafonalakat és Insecteket.
Aktorok	User
Forgatókönyv	1. Amikor az applikáció elindul meghívódik a metódus hogy generáljon Tectonokat 2. Sorrendben leteremti a játékmező elemeit (Tectonok, Gombatestek, Gombafonalak, Insectek)

6. SZÁMÚ USE CASE - JÁTÉK VÉGÉNEK KEZELÉSE

Use-case neve	Játék végének kezelése
Rövid leírás	Amikor véget ért az utolsó Round, akkor eldönti a nyertest és bemutatja a végeredményeket/score-okat a user-nek
Aktorok	User
Forgatókönyv	1. Véget ér az utolsó Round 2. Kiszámolódnak a pontszámok 3. Kiíródik a végeredmény/ki mennyi pontot ért el

7. SZÁMÚ USE CASE - AZ ENDGAMETIMER BEÁLLÍTÁSA

Use-case neve	Az EndgameTimer beállítása
Rövid leírás	A User beállítja a hátralévő Round-ok számát
Aktorok	User
Forgatókönyv	1. Az EndgameTimer beállítása az új értékre

8. SZÁMÚ USE CASE - END TURN KÜLDÉSE

Use-case neve	End turn küldése
Rövid leírás	Akkor történik mikor az egyik játékos úgy gondolja, hogy mostmár befejezi a lépéseit és átadja a Turn-jét. Így elkezdődhet a következő játékos Turn-je.
Aktorok	User
Forgatókönyv	1. A User kiküldi a parancsot, hogy vége van a Turn-jének 2. Végrehajtódnak a Turn-ök közötti események 3. Elkezdődik a sorban következő User/Játékos Turn-je

9. SZÁMÚ USE CASE - JÁTÉKOSOK HOZZÁADÁSA A JÁTÉKHOZ

Use-case neve	Játékosok hozzáadása a játékhoz
Rövid leírás	Miután ki lett választva, hogy hányan szeretnének játszani. A rendszer kiválasztja, hogy ki mi legyen (Gombász vagy Rovarász)
Aktorok	User
Forgatókönyv	1. Valaki megnyitja a játékot 2. Kiválasztódik hány játékos szeretne játszani 3. A rendszer kiossza a játékosokat szerep szerint (Gombász vagy Rovarász)

10. SZÁMÚ USE CASE - JÁTÉK INDÍTÁSA

Use-case neve	Játék indítása
Rövid leírás	Miután ki lettek választva a játékosok és a játék világ legenerálódott, a rendszer sorrendbe rakja a játékosokat majd az első játékosnak a sorból átadja az uralmat.
Aktorok	User
Forgatókönyv	1. Meghívódik a játék elindítása 2. A rendszer sorrendbe rakja a játékosokat 3. A rendszer átadja az uralmat a sorban első játékosnak

11. SZÁMÚ USE CASE – GOMBATEST LÉTREHOZÁSA

Use case neve	Gombatest létrehozása
Rövid leírás	Létrejön egy gombatest a kiválasztott tektonon (céltekton).
Aktorok	Gombász
Forgatókönyv	1. Gombatest – a szükséges feltételek fennállása esetén – a céltektonon létrejön.

12. SZÁMÚ USE CASE – GOMBATEST NÖVESZTÉSE

Use case neve	Gombatest növesztése
Rövid leírás	A gombász olyan játékutasítást ad, hogy jöjjön létre egy gombatest egy általa kiválasztott tektonon (céltekton).
Aktorok	Gombász
Forgatókönyv	1. Gombatest létrehozása a céltektonon. 2. Feltételek vizsgálata. 3. A létrehozott gombatest a céltektonon véglegesen elhelyezésre kerül.

13. SZÁMÚ USE CASE – SPÓRA ELHELYEZÉSE EGY TEKTONRA

Use case neve	Spóra elhelyezése egy tektonra
Rövid leírás	Egy tektonon (céltekton) spóra kerül elhelyezésre.
Aktorok	Gombász
Forgatókönyv	1. Céltekton kiválasztása. 2. Spóra típusának meghatározása. 3. Céltekton nyilvántartásba veszi a spórát.

14. SZÁMÚ USE CASE – GOMBATEST SPÓRAKILÖVÉSE

Use case neve	Gombatest spórakilövése
Rövid leírás	A gombász olyan játékutasítást ad, hogy az általa kiválasztott gombatest lője ki az összes spóráját egy, a gombász által kiválasztott tektonra (céltekton).
Aktorok	Gombász
Forgatókönyv	1. Gombatest kiválasztása. 2. Céltekton kiválasztása. 3. A gombatest a céltektonra kilövi a spórákat.

15. SZÁMÚ USE CASE – GOMBATEST ELPUSZTULÁSA (INAKTÍVVÁ VÁLÁS)

Use case neve	Gombatest elpusztulása (inaktívvá válás)
Rövid leírás	A gombász a gombatest elpusztulására (inaktívvá válás) vonatkozó játékutasítást ad.
Aktorok	Gombász
Forgatókönyv	1. Gombatest kiválasztása. 2. A gombatest elpusztul (inaktívvá válik).

16. SZÁMÚ USE CASE – GOMBAFONÁL LÉTREHOZÁSA

Use-case neve	Gombafonál létrehozása
Rövid leírás	A játékos létrehoz egy gombafonalat.
Aktorok	Gombász
Forgatókönyv	1. Létrejön egy gombafonál

17. SZÁMÚ USE CASE – GOMBAFONÁL HOZZÁADÁSA TEKTONHOZ

Use-case neve	Gombafonál hozzáadása tektonhoz
Rövid leírás	Hozzáadja a kiválasztott fonalat a kiválasztott tektonhoz
Aktorok	Gombász
Forgatókönyv	1. A játékos kiválaszt egy gombafonalat 2. A játékos kiválaszt egy tektont 3. A kiválasztott fonal rákerül a kiválasztott tektonra

18. SZÁMÚ USE CASE – GOMBAFONÁL NÖVESZTÉSE

Use-case neve	Gombafonál növesztése
Rövid leírás	Gombafonál növesztése a kiválasztott tektonra.
Aktorok	Gombász
Forgatókönyv	1. A játékos kiválaszt egy gombatestet vagy gombafonalat, amiből növeszteni fog 2. A gombafonál megvizsgáltatja a kiválasztott tektonnal, hogy képes-e nőni rá 3. A fonál véglegesen létrejön a tektonon

19. SZÁMÚ USE CASE – ROVAR LÉTREHOZÁSA ÉS LETEVÉSE

Use-case neve	Rovar létrehozása és letevése
Rövid leírás	Rovar létrehozódik, és rákerül egy tektonra
Aktorok	Rovarász
Forgatókönyv	1. Ha olyan tektonra hoznánk létre ahol tud lenni (van az adott tektonon Mycelium): 2. Rovar létrehozása 3. Rovar helyének beállítása 4. Location-tektonra rovar rátevése

20. SZÁMÚ USE CASE – ROVAR MOZGATÁSA

Use-case neve	Rovar mozgatása
Rövid leírás	Rovar mozgása egyik tektonról másikra
Aktorok	Rovarász
Forgatókönyv	1. Rovar megnézi, hogy tud-e a céltektonra menni Ha sikerül 2. Régi tektonjáról rovar leszedése 3. Rovar tektonjának átállítása a régiről a céltektonra 4. Rovar hozzáadása a céltektonra 5. Csökken egyel a tevékenységeinek száma a körben

21. SZÁMÚ USE CASE – ROVAR ÁLTALI SPÓRAEVÉS

Use-case neve	Rovar általi spóraevés
Rövid leírás	A rovar megeszik egy spórát a tektonjáról
Aktorok	Rovarász
Forgatókönyv	1. Megnézi, hogy van-e a rovar tektonján spóra Ha van spóra a tektonon 2. A rovar megeszik egy spórát a tektonjáról 3. A tektonról eltűnik az a spóra 4. A spóra beállítja a típusának megfelelő hatást a rovaron 5. Csökken egyel a tevékenységeinek száma a körben

22. SZÁMÚ USE CASE – ROVAR ÁLTALI GOMBAFONÁL ELVÁGÁS

Use-case neve	Rovar általi gombafonál elvágás
Rövid leírás	A rovar elvág egy fonalat a tektonján
Aktorok	Rovarász
Forgatókönyv	1. A rovar elvág egy fonalat a tektonján 2. Ha az utolsó fonalat vágta el, elmenekül 3. Csökken egyel a tevékenységeinek száma a körben

7.3 Tesztelési terv

Teszt-eset neve	Új Tecton sikeres legyártása
Rövid leírás	A rendszer sikeresen legyárt egy általa kiválasztott típusú Tectont. A felhasználó egy időben megmondja azt is, hogy ennek az új Tectonnak kik lesznek a szomszédjai.
Teszt célja	Megnézni, hogy a rendszer sikeresen tud bármilyen típusú Tecton-t gyártani

Teszt-eset neve	Tectontörés
Rövid leírás	A BreakTimert beállítjuk 0-ra egy adott Tectonon, majd megvizsgáljuk, hogy az új kör után lett-e neki új szomszédja és hogy a gombatesten kívül minden más megsemmisült róla.
Teszt célja	Megnézi, hogy a rendszer sikeresen tud-e Tectontörést elvégezni és hogy ilyenkor tényleg minden letörlődik az adott Tectonról (kivéve a gombatest).

Teszt-eset neve	Világ legenerálása
Rövid leírás	A rendszer felépít egy új játék világot. Benne Tectonokkal, rajtuk Gombatestekkel és Gombafonalakkal és Insectekkel.
Teszt célja	Hogy minden egyes játék világ eleme sikeresen legyártható és legyártódik

Teszt-eset neve	Játék végének kezelése
Rövid leírás	Miután lejárt az utolsó Round is, meghatározni mind a Gombászok közül egy nyertest, mind a Rovarászok közül egy nyertest majd kiírni a pontszámukat.
Teszt célja	Megnézni, hogy jól számolódik-e ki a pontszám és hogy helyesen íródnak ki a végeredmények

Teszteset neve	Gombatest sikeres növesztése FertileTectonra (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton) gombafonál által
Rövid leírás	Gombafonál sikeresen növeszt gombatestet olyan FertileTectonon, amely nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton.
Teszt célja	Annak ellenőrzése, hogy gombafonál létre tud-e hozni gombatestet olyan FertileTectonon (céltekton), amely nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton, és amelyen van legalább 3 db spóra és még nincs gombatest. A teszt eredményeként az új gombatest megjelenik a céltektonon. (Gombatest FertileTectonon történő létrehozásának feltétele, hogy a céltektonon legyen 3 db spóra, valamint, hogy ne legyen rajta gombatest.) (Megjegyzés: SustainingTecton, MultiLayeredTecton és AridTecton esetén a teszt hasonlóképpen működik, mutatis mutandis.)

Teszteset neve	Gombatest spórahiány miatti sikertelen növesztése FertileTectonra (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton) gombafonál által
Rövid leírás	Gombafonál sikertelenül kísérel meg gombatestet létrehozni olyan FertileTectonon, amely nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton, és amelyen nem található elegendő spóra.
Teszt célja	Annak ellenőrzése, hogy egy gombafonál létre tud-e hozni gombatestet olyan FertileTectonon (céltekton), amely nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton, és amelyen még nincs gombatest és nincs legalább 3 db spóra. A teszt eredményeként új gombatest nem jelenik meg a játéktéren. (Gombatest FertileTectonon történő létrehozásának feltétele, hogy a céltektonon legyen 3 db spóra, valamint, hogy ne legyen rajta gombatest.) (Megjegyzés: SustainingTecton, MultiLayeredTecton és AridTecton esetén a teszt hasonlóképpen működik, mutatis mutandis.)

Teszteset neve	Gombatest sikertelen növesztése gombafonál által olyan FertileTectonra (nem SustainingTecton, és nem AridTecton), amelyen már van gombatest
Rövid leírás	Gombafonál sikertelenül kísérel meg gombatestet létrehozni olyan FertileTectonon, amely nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton, és amelyen már van gombatest.
Teszt célja	Annak ellenőrzése, hogy egy gombafonál létre tud-e hozni gombatestet olyan FertileTectonon (céltekton), amely nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton, és amelyen van gombatest és legalább 3 db spóra. A teszt eredményeként új gombatest nem jelenik meg a játéktéren. (Gombatest FertileTectonon történő létrehozásának feltétele, hogy a céltektonon legyen 3 db spóra, valamint, hogy ne legyen rajta gombatest.) (Megjegyzés: SustainingTecton, MultiLayeredTecton és AridTecton esetén a teszt hasonlóképpen működik, mutatis mutandis.)

Teszteset neve	Gombatest sikertelen növesztése gombafonál által SemiFertileTectonra
Rövid leírás	Gombafonál sikertelenül kísérel gombatestet létrehozni SemiFertileTectonon, amelyen van legalább 3db spóra (és nincs rajta gombatest)
Teszt célja	Annak ellenőrzése, hogy egy gombafonál létre tud-e hozni gombatestet SemiFertileTectonon, amelyen van legalább 3db spóra (és nincs rajta gombatest). A teszt eredményeként új gombatest nem jelenik meg a játéktéren. (A SemiFertileTecton definíciója szerint az ilyen tektonon nem jöhet létre gombatest.)

Teszteset neve	Gombatest sikeres spórakilövése a gombatest elhelyezkedése szerinti tektonnal szomszédos FertileTectonra (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton)
Rövid leírás	Gombatest sikeresen kilövi a spóráit a gombatest elhelyezkedése szerinti tektonnal szomszédos FertileTectonra (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton)
Teszt célja	Annak ellenőrzése, hogy egy gombatest kit tudja-e lőni a spóráit a gombatest elhelyezkedése szerinti tektonnal szomszédos FertileTectonra (céltekton), amely nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton. A teszt eredményeként a gombatest spóráinak száma 0-ra csökken, és a kilőtt spórákat a továbbiakban a céltekton tartja nyilván. (Gombatest az utolsó, azaz a harmadik kilövése előtt csak valamely szomszédjára lőhet spórát. Fejlettnek az utolsó kilövése során minősül, amely abban nyilvánul meg, hogy ekkor a szomszédja szomszédjára is tud lőni.) (Megjegyzés: SustainingTecton, MultiLayeredTecton, AridTecton és SemiFertileTecton mint céltektonok esetén a teszt hasonlóképpen működik, mutatis mutandis.)

Teszteset neve	Gombatest sikeres spórakilövése olyan FertileTectonra (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton), amely a gombatest elhelyezkedése szerinti tektonnal szomszédos tekton szomszédja
Rövid leírás	Gombatest sikeres, összesen a harmadik (utolsó) spórakilövése olyan, egyébként FertileTectonnak (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton) minősülő tektonra, amely a gombatest elhelyezkedése szerinti tektonnal szomszédos tekton szomszédja.
Teszt célja	Annak ellenőrzése, hogy a gombatest végre tudja-e hajtani a harmadik (összességében az utolsó) spórakilövését olyan FertileTectonra (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton; céltekton), amely a gombatest elhelyezkedése szerinti tektonnal szomszédos tekton szomszédja. A teszt eredményeként a gombatest spóráinak száma 0-ra csökken, és a kilőtt spórákat a továbbiakban a céltekton tartja nyilván. (Gombatest az utolsó, azaz a harmadik kilövése előtt csak valamely szomszédjára lőhet spórát. Fejlettnek az utolsó kilövése során minősül, amely abban nyilvánul meg, hogy ekkor a szomszédja szomszédjára is tud lőni.) (Megjegyzés: SustainingTecton, MultiLayeredTecton, AridTecton és SemiFertileTecton mint céltektonok esetén a teszt hasonlóképpen működik, mutatis mutandis.)

Teszteset neve	Gombatest sikertelen spórakilövése olyan FertileTectonra (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton), amely a gombatest elhelyezkedése szerinti tektonnal szomszédos tekton szomszédja
Rövid leírás	Gombatest sikertelenül kísérel meg spórakilövést olyan FertileTectonra (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton), amely a gombatest elhelyezkedése szerinti tektonnal szomszédos tekton szomszédja, mert nem ez lesz a gombatest harmadik (összességében az utolsó) spórakilövése.
Teszt célja	Annak ellenőrzése, hogy a gombatest végre tudja-e hajtani a nem az utolsó (azaz nem a harmadik) spórakilövését olyan FertileTectonra (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton; céltekton), amely a gombatest elhelyezkedése szerinti tektonnal szomszédos tekton szomszédja. A teszt eredményeként a gombatest spóráinak száma változatlan marad, és a céltekton által nyilvántartott spórákban sem következik be változás. (Gombatest az utolsó, azaz a harmadik kilövése előtt csak valamely szomszédjára lőhet spórát. Fejlettnek az utolsó kilövése során minősül, amely abban nyilvánul meg, hogy ekkor a szomszédja szomszédjára is tud lőni.) (Megjegyzés: SustainingTecton, MultiLayeredTecton, AridTecton és SemiFertileTecton mint céltektonok esetén a teszt hasonlóképpen működik, mutatis mutandis.)

Teszteset neve	Gombatest sikertelen spórakilövése olyan FertileTectonra (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton), amely a gombatest elhelyezkedése szerinti tekton harmadik szomszédja
Rövid leírás	Gombatest sikertelenül kísérel meg spórakilövést olyan FertileTectonra (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton, céltekton), amely a gombatest elhelyezkedése szerinti tekton harmadik szomszédja. [Azaz létezik A, B, C és D FertileTecton, amelyek a következőképpen szomszédosak (a szomszédosságot a – jelöli): A – B – C – D. (A tektonok egyéb módon nem szomszédosak egymással.) A gombatest A FertileTectonon található.]
Teszt célja	Annak ellenőrzése, hogy a gombatest végre tud-e hajtani spórakilövést olyan FertileTectonra (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton; céltekton), amely a gombatest elhelyezkedése szerinti tekton harmadik szomszédja – a művelet a gombatest érettségétől függetlenül nem lehetséges. (Gombatest az utolsó, azaz a harmadik kilövése előtt csak valamely szomszédjára lőhet spórát. Fejlettnek az utolsó kilövése során minősül, amely abban nyilvánul meg, hogy ekkor a szomszédja szomszédjára is tud lőni.) A teszt eredményeként a gombatest spóráinak száma változatlan marad, és a céltekton által nyilvántartott spórákban sem következik be változás. (Megjegyzés: SustainingTecton, MultiLayeredTecton, AridTecton és SemiFertileTecton mint céltektonok esetén a teszt hasonlóképpen működik, mutatis mutandis.)

Teszteset neve	Elpusztult (inaktív) gombatest sikertelen spórakilövése a gombatest elhelyezkedése szerinti tektonnal szomszédos FertileTectonra (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton)
Rövid leírás	Elpusztult (inaktív) gombatest sikertelenül próbálja meg kilőni a spóráit a gombatest elhelyezkedése szerinti tektonnal szomszédos FertileTectonra (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton; céltekton)
Teszt célja	Annak ellenőrzése, hogy egy elpusztult (inaktív) gombatest ki tudja-e lőni a spóráit a gombatest elhelyezkedése szerinti tektonnal szomszédos FertileTectonra, amely nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton. A teszt eredményeként semmilyen változás nem következik be. Inaktív gombatest semmilyen cselekvésre nem képes. (Ebbe az állapotba közvetlenül a harmadik spórakilövése után kerül a gombatest.) (Megjegyzés: SustainingTecton, MultiLayeredTecton, AridTecton és SemiFertileTecton mint céltektonok esetén a teszt hasonlóképpen működik, mutatis mutandis.)

Teszteset neve	StunSpore sikeres elhelyezése FertileTectonon (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton)
Rövid leírás	StunSpore sikeresen elhelyezésre kerül egy FertileTectonon (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton; céltekton)
Teszt célja	Annak ellenőrzése, hogy a StunSpore sikeresen elhelyezésre került-e a FertileTectonon (céltekton; nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton), amelynek eredményeként a spóra újólag megjelenik a céltekton nyilvántartásában. (Megjegyzés: SustainingTecton, MultiLayeredTecton, AridTecton és SemiFertileTecton mint céltektonok, valamint PreventCutSpore, SpeedSpore és SlownessSpore esetén a teszt hasonlóképpen működik, mutatis mutandis.)

Teszt-eset neve	Gombafonál sikeres (lassú) növesztése gombatestből FertileTectonra (nem MultiLayeredTecton és nem AridTecton)
Rövid leírás	A gombafonál rá-nő a tesztelő által kiválasztott FertileTectonra, mert az még nincs „tele” fonállal (rajta lévő lehetséges fonalak száma és rajta lévő fonalak száma egyenlő) és közvetlen szomszédja a másik FertileTectonnak, amin van a növesztést kezdeményező gombatest. A kiválasztott FertileTectonon nincs spóra, ezért a gombafonál lassan (2 kör allat) fog nőni. (Megjegyzés: a növesztés hasonlóan működik, ha a tekton, ahova növesztünk MultiLayeredTecton, AridTecton SustainingTecton vagy SemiFertileTecton. Akkor is hasonló a teszteset, ha gombafonálból növesztünk.)
Teszt célja	Megvizsgáljuk a gombafonál osztálynak a növését, a növéshez tartozó feltételeket és azt, hogy a tektonon ténylegesen rajta lesz-e a fonál. A kiválasztott FertileTectonon megjelenik egy új gombafonál 2 kör után.

Teszt-eset neve	Gombafonál sikertelen növesztése gombatestből, olyan FertileTectonra (nem MultiLayeredTecton és nem AridTecton), ahol már van gombafonál
Rövid leírás	A gombafonál nem nő rá a tesztelő által kiválasztott FertileTectonra, mert az kiválasztott FertileTecton már „tele” van fonállal (kapacitása és rajta lévő fonalak száma egyenlő). (Megjegyzés: a növesztés hasonlóan működik, ha a tekton, ahova növesztünk MultiLayeredTecton, AridTecton SustainingTecton vagy SemiFertileTecton. Akkor is hasonló a teszteset, ha gombafonálból növesztünk.)
Teszt célja	Megvizsgáljuk a gombafonál növését és a növéshez tartozó feltételeket. Nem lesz változás a játéktéren a parancs kiadása előtti állapothoz képest.

Teszt-eset neve	Gombafonál sikertelen növesztése olyan FertileTectonra (nem MultiLayeredTecton és nem AridTecton), ami a növést kezdeményező gombatest tektonjával nem közvetlenül szomszédos.
Rövid leírás	A gombafonál nem nő rá a tesztelő által kiválasztott FertileTectonra, mert az nem szomszédja a másik tektonnak, amin van a növesztést kezdeményező gombatest. (Megjegyzés: a növesztés hasonlóan működik, ha a tekton, ahova növesztünk MultiLayeredTecton, AridTecton SustainingTecton vagy SemiFertileTecton. Akkor is hasonló a teszteset, ha gombafonálból növesztünk.)
Teszt célja	Megvizsgáljuk a gombafonál növését és a növéshez tartozó feltételeket. Nem lesz változás a játéktéren a parancs kiadása előtti állapothoz képest.

Teszt-eset neve	Gombafonál sikeres gyors növesztése FertileTectonra (nem MultiLayeredTecton és nem AridTecton)
Rövid leírás	A gombafonál rá nő a tesztelő által kiválasztott FertileTectonra mert az még nincs „tele” fonállal (kapacitása és rajta lévő fonalak száma egyenlő) és közvetlen szomszédja a másik FertileTectonnak, amin van a növesztést kezdeményező gombatest. A kiválasztott FertileTectonon van 1 spóra, ezért a gombafonál gyorsabban (1 kör alatt) fog nőni. (Megjegyzés: a növesztés hasonlóan működik, ha a tekton, ahova növesztünk MultiLayeredTecton, AridTecton SustainingTecton vagy SemiFertileTecton. Akkor is hasonló a teszteset, ha gombafonálból növesztünk. A céltektonon lehetne több mint egy spóra is, ez nem változtatna a működésen.)
Teszt célja	Megvizsgáljuk a gombafonál osztálynak a növését, a növéshez tartozó feltételeket és azt, hogy a tektonon ténylegesen rajta lesz-e a fonál. A kiválasztott FertileTectonon megjelenik egy új gombafonál egy kör után.

Teszt-eset neve	Húsevő fonál általi rovarevés és gombatest növesztés
Rövid leírás	Új kör kezdetekor a FertileTectonon (nem MultiLayeredTecton és nem AridTecton) lévő húsevő fonál megeszi a rajta lévő bénult állapotban lévő rovarokat és gombatestet növeszt. (Jelen esetben a vizsgált FertileTectonon még nincs gombatest.)
Teszt célja	Megvizsgáljuk a CarnivorousMycelium osztály működését, ha teljesülnek az ahhoz szükséges feltételek. Valamint azt, hogy megtörténik-e az ebből következő gombatest növesztés és rovarok halála. A FertileTectonon meghalnak a rovarok és nő egy új gombatest.

Teszt-eset neve	Gombafonál elhalása AridTectonon
Rövid leírás	Új kör kezdetekor az AridTectonon lévő fonál elpusztul, mert már 5 köre van ott.
Teszt célja	Megvizsgáljuk az AridTecton többi tektontól különböző működését. A vizsgált AridTectonon lévő fonál elpusztul.

Teszt-eset neve	Rovar mozgása
Rövid leírás	Rovar mozog egy tektonról a másikra
Teszt célja	Megvizsgáljuk, hogy ténylegesen megváltozott-e a kettő tekton és a rovar állapota

Teszt-eset neve	Rovar sikertelen mozgása nem-szomszédos tektonra
Rövid leírás	Rovar mozogna egy tektonról a másikra, de nem tud, mert a location-tektonja nem szomszédos a céltektonnal
Teszt célja	Minden alapállapotban marad-e

Teszt-eset neve	Rovar sikertelen mozgása olyan tektonra, amelyen nincs gombafonál
Rövid leírás	Rovar mozogna egy tektonról a másikra, de nem tud, mert nincs a céltektonon gombafonál
Teszt célja	Megvizsgáljuk, hogy minden alapállapotban marad-e

Teszt-eset neve	Rovar általi spóraevés következtében kettészakadás
Rövid leírás	Spóraevés hatására a rovar kettészakad
Teszt célja	Megvizsgáljuk, hogy létrejött-e a másik rovar, a megfelelő tulajdonságokkal

Teszt-eset neve	Rovar általi spóraevés következtében Slow állapotba kerülés
Rövid leírás	A rovar spóraevés hatására Slow állapotba kerül
Teszt célja	Megvizsgáljuk, hogy ténylegesen belekerült-e az adott állapotba, és képességei ezek szerint megváltoztak-e

Teszt-eset neve	Rovar általi spóraevés következtében Fast állapotba kerülés
Rövid leírás	A rovar spóraevés hatására Fast állapotba kerül
Teszt célja	Megvizsgáljuk, hogy ténylegesen belekerült-e az adott állapotba, és képességei ezek szerint megváltoztak-e

Teszt-eset neve	Rovar általi spóraevés következtében PreventCunt állapotba kerülés
Rövid leírás	A rovar spóraevés hatására PreventCut állapotba kerül
Teszt célja	Megvizsgáljuk, hogy ténylegesen belekerült-e az adott állapotba, és képességei ezek szerint megváltoztak-e

Teszt-eset neve	Rovar általi spóraevés következtében Stunned állapotba kerülés
Rövid leírás	A rovar spóraevés hatására Stunned állapotba kerül
Teszt célja	Megvizsgáljuk, hogy ténylegesen belekerült-e az adott állapotba, és képességei ezek szerint megváltoztak-e

Teszt-eset neve	Rovar általi sikertelen spóraevés
Rövid leírás	A rovar spórát próbálna enni, de nincs spóra a tektonon
Teszt célja	Megvizsgáljuk, hogy minden alapállapotban marad-e

Teszt-eset neve	Rovar általi gombafonál elvágás
Rövid leírás	Rovar elvág egy gombafonalat a tektonján
Teszt célja	Megvizsgáljuk, hogy a rovar tektonján a fonalak állapota tükrözi-e, hogy az egyik elvágódott, nem maradtak-e nem fenntartott fonalak akárhol, illetve, hogy a rovarok megfelelő tektonokra menekültek-e

Teszt-eset neve	Rovar létrehozása és letevése
Rövid leírás	A rovar létrejön és letevődik a céltektonra
Teszt célja	Megvizsgáljuk, hogy ténylegesen létrejött-e a rovar és rajta van-e a tektonon

7.4 Tesztelést támogató segéd- és fordítóprogramok specifikálása

A teszteléshez JUnit 5[1] test framework-öt fogunk használni.

[1] https://junit.org/junit5/

7.5 Napló

Kezdet	Időtartam	Résztvevők	Leírás
2025.03.26 ., 20:00	2 óra	Bencze Guzmics Kohár Rakos Taba	Értekezlet. Döntések: - Követelmények átnézése - Feladatok kiosztása - Kohár a 4 új feature implementálása - Bencze Insect-el kapcsolatos use-casek és test-casek - Guzmics Gombafonállal kapcsolatos use-casek és test-casek - Taba Gombatestekkel kapcsolatos use-casek és test-casek - Rakos Tectonokkal és Játéklogikával kapcsolatos use-casek és test-casek
2025.03.27 ., 12:00	2 óra	Kohár	Tevékenység: - Az új funkciók implementálása az osztály diagramba - Az új funkciók szekvencia diagramjainak elkészítése
2025.03.27 ., 14:00	2 óra	Rakos	Tevékenység: - A vonatkozó use-casek és test-casek kidolgozása (itt még csak a Tektonnal kapcsolatosak)
2025.03.27 ., 17:00	1 óra	Guzmics	Tevékenység: - A vonatkozó use-casek és test-casek előkészítése
2025.03.27 ., 17:00	45 perc	Bencze	Tevékenység: - A vonatkozó use-casek és test-casek kezdetleges kidolgozása
2025.03.27 ., 18:00	1 óra	Taba	Tevékenység: - A vonatkozó use-casek és test-casek előkészítése
2025.03.27 ., 19:00	2 óra 30 perc	Bencze Guzmics Kohár Rakos Taba	Értekezlet. Döntések: - Az eddigi kitalált use-casek és test-casek átnézése - Kohár által kibővített osztály diagram átnézése

			- Újonnan kidolgozott szekvencia diagramok átnézése
2025.03.27 ., 22:00	1 óra	Kohár	Tevékenység: - Osztály diagram és szekvencia diagram javítása
2025.03.28 ., 14:00	2 óra	Taba	Tevékenység: - A vonatkozó use-casek és test-casek kidolgozása
2025.03.28 ., 14:00	1 óra 30 perc	Rakos	Tevékenység: - A vonatkozó use-casek és test-casek további kidolgozása - A játék menetével-logikájával kapcsolatos use és test-casek kidolgozása
2025.03.28 ., 15:30	1 óra	Benzce	Tevékenység: - A vonatkozó use-casek és test-casek további kidolgozása
2025.03.28 ., 16:00	1 óra	Guzmics	Tevékenység: - A vonatkozó use-casek és test-casek további kidolgozása
2025.03.28 ., 17:00	2 óra 30 perc	Bencze Guzmics Kohár Rakos Taba	Értekezlet. Döntések: - A játék kezdetének, menetének és végének átbeszélése - Kidolgozott use-casek és test-casek átbeszélése
2025.03.28 ., 20:00	1 óra	Kohár	Tevékenység: - Kimeneti nyelv definiálása - Osztálydiagramok exportálása
2025.03.28 ., 20:00	30 perc	Bencze	Tevékenység: - A vonatkozó use-casekben és test-casekben hibák javítása
2025.03.28 ., 21:00	1 óra	Guzmics	Tevékenység: A vonatkozó use-casek és test-casek további kidolgozása
2025.03.29 ., 13:00	3 óra 30 perc	Taba	Tevékenység: - A vonatkozó use-casek és test-casek továbbfejlesztése - Objektumkatalógus módosítása
2025.03.29 ., 14:00	1 óra	Rakos	Tevékenység: - A vonatkozó use-casek és test-casek javítása,

			helyesírási hibák javítása
2025.03.29 ., 16:00	1 óra	Guzmics	Tevékenység: A vonatkozó use-casek és test-casek további kidolgozása
2025.03.29 ., 17:00	1 óra	Bencze Guzmics Kohár Rakos Taba	Értekezlet. Döntések: - Javasolt nyelvi elemekkel kiegészíteni a use-caseket - Pontosítani és összehasonlítani a use-caseket és test-caseket
2025.03.29 ., 19:00	30 perc	Kohár	Tevékenység: - Osztálykatalógus átnézése
2025.03.30 ., 2:00	30 perc	Bencze	Tevékenység: - A vonatkozó use-casek és test-casek további kidolgozása - Bemenetek leírása
2025.03.30 ., 7:00	3 óra	Guzmics	Tevékenység: - A vonatkozó use-casek és test-casek további kidolgozása - Bemenetek leírása
2025.03.30 ., 8:00	30 perc	Rakos	Tevékenység: - A vonatkozó use-casek és test-casek véglegesítése - Taba által kidolgozott use-casek és test-casek átnézése
2025.03.30 ., 10:00	1 óra 30 perc	Taba	Tevékenység: - A vonatkozó bemeneti nyelv use case-enkénti kidolgozása - Guzmics munkájának részletes áttekintése
2025.03.30 ., 14:30	1 óra	Guzmics	Tevékenység: - Bencze munkájának részletes átnézése
2025.03.30 ., 16:00	1 óra	Bencze Guzmics Kohár Rakos Taba	Értekezlet. Döntések: - Végleges átbeszélése a megírt use-caseknek és test-caseknek - Hibás vagy hiányos use-casek és test-casek észrevétele
2025.03.30 ., 18:00	1 óra	Taba	Tevékenység: - A vonatkozó use-casek és test-casek véglegesítése - Objektumkatalógus véglegesítése

2025.03.30 ., 20:00	5 perc	Kohár	Tevékenység: - Teszt támogató segédprogram definiálása
2025.03.30 ., 20:00	30 perc	Bencze	Tevékenység: - A vonatkozó use-casek és test-casek véglegesítése
2025.03.30 ., 22:00	1 óra	Rakos	Tevékenység: - A naplózás összesítése és részletes megírása

8. Részletes tervek

25 – bandITs

Konzulens:
Huszerl Gábor

Csapattagok

Bencze János István	GIWUHT	gomanpc@yahoo.com
Guzmics Gergő	VC8OQD	guzmicsgergo@gmail.com
Kohár Zsombor	Q8EPW6	zsombor.kohar@edu.bme.hu
Rakos Gergő Máté	I3Q7BY	gergo_rakos@yahoo.com
dr. Taba Szabolcs Sándor	JRGMBW	taba.szabolcs@gmail.com

2025. 04. 14.

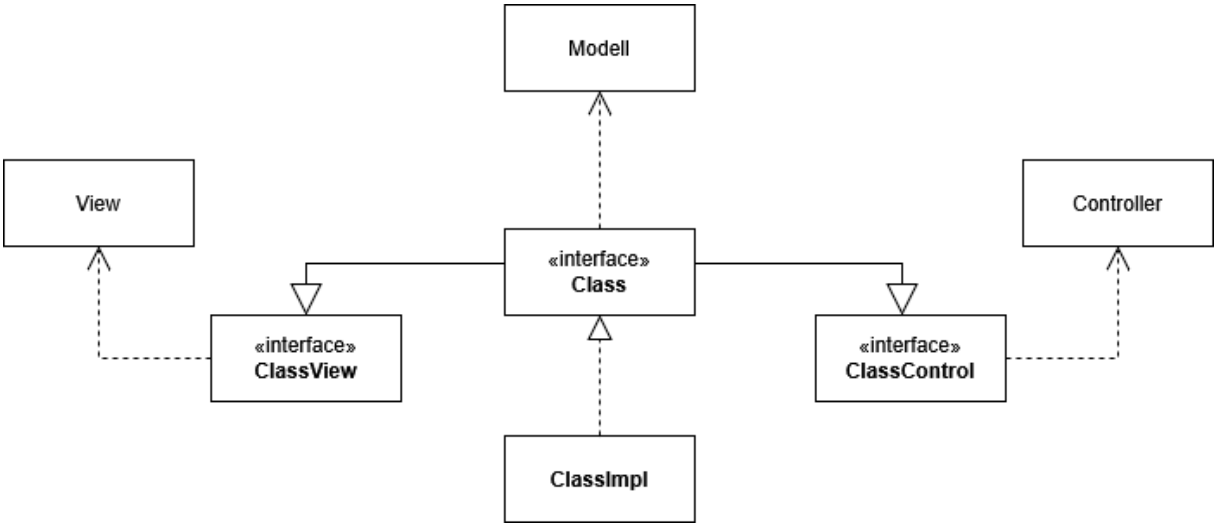
8. Részletes tervek

[A dokumentum célja, hogy pontosan specifikálja az implementálandó osztályokat, beleértve a privát attribútumokat és metódusokat, ezek definícióját is.
A dokumentum második fele részletesen be kell mutassa a korábban definiált be- és kimeneti nyelv szintakszisát felhasználva, hogy mely tesztekkel lesz a prototípus ellenőrizve.]

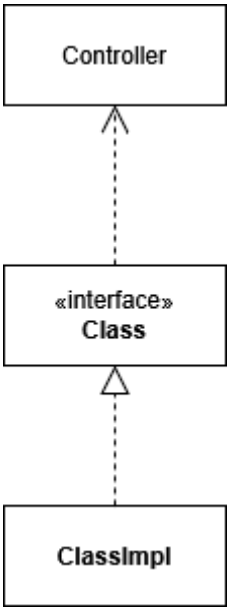
Osztály anatómia

Mivel az osztályoknak a felépítése, első látásra nem feltétlen átlátható, és az analízis modellben lévő interfész felépítéssel nem egyezik meg, ezért jött létre ez a magyarázó rész:

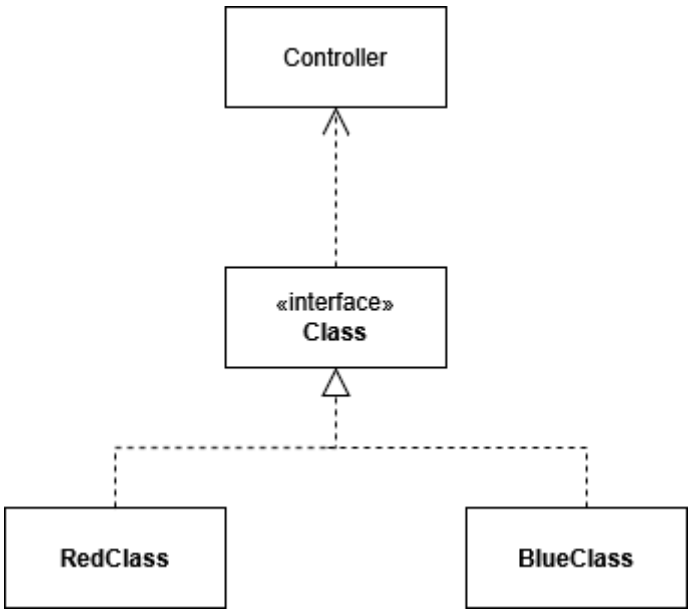
Legnagobb változtatásokon a modell osztályai mentek keresztül. Egy osztály minden esetben legalább 3 interfészt valósít meg. Egy, amit a kontroller lát, egy, amit a view és egy, amit a modell többi része is lát. Mivel a modell többi része mindig látja azt, ami a view vagy a kontroller ezért a modell interfésze örököl a view és a kontroller interfészétől. Az elnevezési konvenció az alábbi: A modell interfésze kapja az osztály alapvető nevét, a kontroller interfésze az osztály neve mellé a „Control” utónevet kapja, a view interfésze a az osztály neve mellé a „View” utónevet kapja és az implementáció pedig az osztály neve mellé az „Impl” utónevet kapja. Az alábbi leírás UML diagrammként:



Mivel a kontroller osztályainál az ilyen szintű szeparáció értelmetlen, mivel csak egymástól függenek, ezért azok egy sokkal egyszerűbb szeparációt követnek, ami alapvetően ugyan azt a elnevezési konvenciót követik. A leírás UML diagrammként:



Lehetséges, hogy egy külsőleg azonos, de belső működésben más osztályok ugyan azt az interfészt valósítják meg. Ebben az esetben az elnevezésnél az osztály nevéhez egy megkülönböztető jelzőt kap előnévként az osztály neve mellé. A leírás UML diagrammként:



8.1 Osztályok és metódusok tervei.

8.1.1 TectonImpl

- **Felelősség**

  Az absztrakt Tecton osztály implementációja. Kezeli a gombatestek és gombafonalak fenntartását, rajta lévő objektumok tárolását. Szomszédos tektonok tárolását.

- **Interfészek**

  Tecton, RoundBeginSubscriber

- **Attribútumok**
  - **-breakTimer**: int
    Az időzítő amely ha elér 0-ra a tekton eltörik
  - ***-neighbours: List<Tecton>***
    A tektonnal szomszédos tektonok listája
  - ***-myceliumCapacity: int***
    Maximum ennyi gombafonál lehet az adott tektonon
  - ***-spores: Queue<Spore>***
    Az adott tektonon lévő spórák listája
  - ***-mushroomBody: MushroomBody***
    Itt van eltárolva ha az adott tektonon van-e gombatest
  - ***-mycelia: Queue<Mycelium>***
    Az adott tektonon lévő gombafonalak listája
  - ***-occupants: List<Insect>***
    Az adott tektonon lévő bogarak listája
  - ***-notSustained: Set<Tecton>***
    A tektonok összessége, aminek fonalai nincsenek összeköttetésben gombatestel, ezért el fognap pusztulni

- **Metódusok**
  - ***+getBreakTimer(): int***
    A tektontörés getterje
  - ***+setBreakTimer(breakTimer: int): void***
    A tektontörés setterje. A kapott paraméter az új körök száma a törésig
  - ***+getNeighbours(): List<Tecton>***
    A tekton szomszédlistájának getterje
  - ***+setNeighbours(neighbours: List<Tecton>): void***
    A tekton szomszédlistájának setterje. A kapott paraméter az új szomszédok listája
  - ***+getMyceliaCapacity(): int***
    A maximális gombafonál szám
  - ***+setMyceliaCapacity(myceliaCapacity: int): void***
    A maximális gombafonál szám setterje
  - ***+getSpores(): Queue<Spore>***
    A tektonon lévő spórák getterje
  - ***+setSpores(spores: Queue<Spore>): void***
    A tektonon lévő spórák setterje

- ***+getMushroomBody(): MushroomBody***
  A tektonon lévő gombatest getterje
- ***+setMushroomBody(mushroomBody: MushroomBody): void***
  A tektonon lévő gombatest setterje
- ***+getMycelia(): Queue<Mycelium>***
  A tektonon lévő gombafonalak getterje
- ***+setMycelia(mycelia: Queue<Mycelium>): void***
  A tektonon lévő gombafonalak setterje
- ***+distance(tecton: Tecton): int***
  A függvény megadja, hogy milyen messze van egy cél tekton a jelenlegi tektontól (ezt a metódust a spórák kilövés miatt kell használni, hogy az adott gombatest tudja, hogy melyik tektonra szabad, vagy nem, spórát lőjön)

```
procedure distance(target_tecton):

// Inicializáció
 distances ← CREATE new Map (to store Tecton → Integer distance)
 queue ← CREATE new Queue (to store Tectons to visit)

// BFS elindítása az adott Tectonrol
 SET distances[this] ← 0  // Távolság a kezdeti Tectontol önmagáig
 ENQUEUE this INTO queue       // a queue-ba belerakni a kezdeti Tectont

 //BFS
 WHILE queue IS NOT EMPTY DO
   // Lekérni a következő Tectont
     current_tecton ← DEQUEUE from queue
     // Lekérni a távolságát
     current_distance ← GET distances[current_tecton]

    // Megnézni, hogy elértük-e a cél Tectont
     IF current_tecton IS target_tecton THEN
        RETURN current_distance
     END IF

     // Meglátogatni a szomszédos Tectonokat
     FOR EACH neighbour IN current_tecton.neighbours DO
    // Ha meg nem volt ez a Tecton meglatogatva
        IF distances DOES NOT CONTAIN neighbour THEN
       // Beállítani mint látogatott, elmenteni távolságát es a queue-ba rakni
            SET distances[neighbour] ← current_distance + 1
            ENQUEUE neighbour INTO queue
        END IF
     // Különben nem csinálunk semmit
     END FOR EACH
 END WHILE

// Ha nem találtuk meg a cél Tectont
 RETURN -1
END
```

- ***-neighboursWithMycelia(): List<Tecton>***
  Azok a szomszédok összege, amelyen van gombafonál vagy gombatest
- ***-myceliaCheckSustain(): void***
  A függvény megnézi, hogy a tekton és velük gombafonállal összekötött tektonok még összekötésben állnak-e gombatestel

```
procedure myceliaCheckSustain:
```

```
// Inicializáció
connected ← CREATE new Set<Tecton>
queue ← CREATE new Queue<Tecton>
visited ← CREATE new Set<Tecton>
isSustaining = false

ADD this To visited
ENQUEUE this ONTO queue

 //BFS
 WHILE queue IS NOT EMPTY DO
   // Lekerni a kovetkezo Tectont
     current_tecton ← DEQUEUE from queue
   ADD current_tecton TO connected
      IF current_tecton.sustaining THEN BEGIN
       isSustaining = true
      END

      FOR EACH neighbour IN neighboursWithMycelia(current_tecton) DO BEGIN
       IF ADD neighbour TO visited returns True THEN BEGIN
             ENQUEUE neighbour ONTO queue
       END
      END FOR EACH
 END WHILE
 IF isSustaining IS False THEN BEGIN
      ADD ALL elements FROM connected TO notSustained
 END
END
```

- ***+checkNeighbourMyceliaSustain(): void***
  A függvény megnézi, hogy a szomszédos tektonok és velük gombafonállal összekötött tektonok még összeköttetésbe állnak-e gombatestel

```
procedure checkNiegbhourMyceliaSustain:

 // Inicializáció
 CLEAR notSustained

 FOR EACH neighbour IN this.neighbours DO BEGIN
      CALL neighbour.myceliaCheckSustain
 END

 FOR EACH tecton IN notSustained DO BEGIN
      FOR EACH m IN tecton.getMycelia DO BEGIN
            REMOVE m FROM tecton.mycelia
            CALL m.delete
      END FOR EACH
 END FOR EACH
END
```

- ***+getOccupants(): List<Insect>***
  A tektonon lévő rovarok listájának getterje
- ***+setOccupants(occupants: List<Insect>): void***
  A tektonon lévő rovarok listájának setterje
- ***+addOccupant(insect: Insect): void***
  Hozzáad egy rovart a tektonhoz
- ***+removeOccupant(insect: Insect): void***
  Levesz egy rovart a tektonról
- ***+hasMycelium(): boolean***

true-t ad vissza, ha van-e legalább 1 gombafonál a tektonon ami nem növekszik, különben false

- ***+addMycelium(mycelium: Mycelium): void***
  Hozzáad egy gombafonalat a tektonhoz
- ***+addSpore(spore: Spore): void***
  Hozzáad egy spórát a tektonhoz
- ***+transferSpores(newSpores: List<Spore>): void***
  Hozzáad egyszerre több spórát a tektonhoz
- ***+addNeighbour(tecton: Tecton): void***
  Egy új szomszédot ad a tektonnak
- **accept(myceliumGrowthEvaluator: MyceliumGrowthEvaluator, mycelium: Mycelium): void**
  Eldönti, hogy az adott gombafonál nőhet-e ezen a tektonon
- **accept(mushroomBodyGrowthEvaluator: MushroomBodyGrowthEvaluator, mushroomBody: MushroomBody): void**
  Eldönti, hogy az adott gombatest nőhet-e ezen a tektonon
- ***+sustaining(): boolean***
  A sustaining tektonnál true, a többinél false, kivéve ha van rajtuk gombatest, akkor true
- ***killOccupants(): void***
  Megpróbál minden rajta lévő rovart eltávolítani
- **+eatSpore(insect: SporeEater):** Ha van spóra a tectonon, meghívja az első spórának az eatSpore() metódusát, a megkapott insect-el, mint argumentum
- **+cutMycelium():** Elvágódik az első spóra a tectonon
-

```
procedure cutMycelium:

      DEQUEUE mycelium FROM mycelia
      mycelium.cutWithDelay

END
```

- **+moveInsect(insect: InsectMover, insectLocation: Tecton):** Ha tud az insect a tectonra (amin meg volt hívva a metódus) mozogni, akkor megcsinálja ezt a műveletet

```
procedure moveInsect(insect, insectLocation):

 distance = insectLocation.distance(this)

      IF (distance IS EQUAL TO 1) AND (this.hasMycelium IS TRUE) THEN BEGIN
            insectLocation.removeOccupant(insect)
            this.addOccupant(insect)
            insect.setLocation(this)
            insect.setRemainingMoves(insect.getRemainingMoves() - 1)
      END
END
```

8.1.2 FertileTectonImpl

- **Felelősség**

A FertileTecton típusú tektonok növesztésének feltételét szabályozza. A többi felelősségét az absztrakt Tecton osztálytól örökli.

- **Ősosztályok**

TectonImpl → FertileTectonImpl

- **Interfészek**

FertileTecton, RoundBeginSubscriber

- **Metódusok**
  - ***+FertileTecton():***
    konstruktor, a gombafonál kapacitást beállítja 1-re és a BreakTimer-jét is beállítja
  - ***+accept(myceliumGrowthEvaluator: MyceliumGrowthEvaluator, mycelium: Mycelium): void***
    eldönti hogy a kapott gombafonál nőhet-e az adott tektonon

```
procedure accept(myceliumGrowthEvaluator : MyceliumGrowthEvaluator, mycelium :
Mycelium)

 currentMyceliaCount ← GET this.getMycelia.size
 capacity ← GET this.getMyceliaCapacity

 IF currentMyceliaCount >= capacity THEN BEGIN
      CALL mycelium.delete
      RETURN
 END
 ADD mycelium TO this.getMycelia
 sporeCount ← GET this.getSpores.size
 CALL mycelium.grow(sporeCount)
END
```

  - ***+accept(mushroomBodyGrowthEvaluator: MushroomBodyGrowthEvaluator , mushroomBody: MushroomBody): void***
    eldönti hogy a kapott gombatest nőhet-e az adott tektonon

```
procedure accept(mushroomBodyGrowthEvaluator : MushroomBodyGrowthEvaluator,
mushroomBody : MushroomBody)

 sporeCount ← GET this.getSpores.size
 hasExistingMushroomBody ← (GET this.getMushroomBody IS NOT NULL)
 hasMycelia ← (GET this.getMycelia.isEmpty IS FALSE)

 IF (sporeCount < 3) OR (hasExistingMushroomBody IS TRUE) OR (hasMycelia IS
      FALSE)  THEN BEGIN
            CALL muhsroomBody.delete
            RETURN
 END

 CALL this.setMushroomBody(mushroomBody)
 CALL mushroomBody.grow(sporeCount)
```

```
END
```

- *+**onRoundBegin(): void***
  Itt történik a tektontörés és annak következményei

```
procedure onRoundBegin:

 currentBreakTimer ← GET this.getBreakTimer
SET this.breakTimer ← currentBreakTimer − 1

currentBreakTimer ← GET this.getBreakTimer
IF currentBreakTimer <= 0 THEN BEGIN
      WHILE GET this.getMycelia.isEmpty IS FALSE DO BEGIN
            mycelium ← DEQUEUE FROM this.getMycelia
            IF mycelium IS NOT NULL THEN BEGIN
                  CALL mycelium.cutImmediate
            END
      END WHILE

      newFertileTecton ← CREATE new FertileTecton
      CALL newFertileTecton.addNeighbour(this)
      CALL this.addNeighbour(newFertileTecton)

 END

END
```

- ***sustaining(): boolean***
  Visszaadja, ha az adott tekton képes-e fenntartani gombafonalakat. FertileTectonnal ha van rajta gombatest akkor true-val tér vissza különben false

8.1.3 SemiFertileTectonImpl

- **Felelősség**

A SemiFertileTecton típusú tektonok növesztésének feltételét szabályozza. A többi felelősségét az absztrakt Tecton osztálytól örökli.

- **Ősosztályok**

TectonImpl → SemiFertileTectonImpl

- **Interfészek**

SemiFertileTecton, RoundBeginSubscriber

- **Metódusok**
  - ***+SemiFertileTecton():***
    konstruktor, a gombafonál kapacitást beállítja 1-re és a BreakTimer-jét is beállítja
  - ***+accept(myceliumGrowthEvaluator: MyceliumGrowthEvaluator, mycelium: Mycelium): void***
    eldönti hogy a kapott gombafonál nőhet-e az adott tektonon

```
procedure accept(myceliumGrowthEvaluator : MyceliumGrowthEvaluator, mycelium :
Mycelium)
```

```
 currentMyceliaCount ← GET this.getMycelia.size
 capacity ← GET this.getMyceliaCapacity

 IF currentMyceliaCount >= capacity THEN BEGIN
      CALL mycelium.delete
      RETURN
 END
 ADD mycelium TO this.getMycelia
 sporeCount ← GET this.getSpores.size
 CALL mycelium.grow(sporeCount)

END
```

- ***+accept(mushroomBodyGrowthEvaluator: MushroomBodyGrowthEvaluator, mushroomBody: MushroomBody): void***
  eldönti hogy a kapott gombatest nőhet-e az adott tektonon. Itt szimplán kitörli a kapott gombatestet, mivel SemiFertile tectonon sosem nőhet gombatest
- ***+onRoundBegin(): void***
  Itt történik a tektontörés és annak következményei

```
procedure onRoundBegin:

 currentBreakTimer ← GET this.getBreakTimer
 SET this.breakTimer ← currentBreakTimer − 1

 currentBreakTimer ← GET this.getBreakTimer
 IF currentBreakTimer <= 0 THEN BEGIN
      WHILE GET this.getMycelia.isEmpty IS FALSE DO BEGIN
            mycelium ← DEQUEUE FROM this.getMycelia
            IF mycelium IS NOT NULL THEN BEGIN
                  CALL mycelium.cutImmediate
            END
      END WHILE

      newFertileTecton ← CREATE new FertileTecton
      CALL newFertileTecton.addNeighbour(this)
      CALL this.addNeighbour(newFertileTecton)

 END

END
```

- ***sustaining(): boolean***
  Visszaadja, ha az adott tekton képes-e fenntartani gombafonalakat. SemiFertileTectonnal mindig false-al tér vissza

8.1.4 MultiLayeredTectonImpl

- **Felelősség**

A MultiLayeredTecton típusu tektonok növesztésének feltételét szabályozza. A többi felelősségét az absztrakt Tecton osztálytól örökli.

- **Ősosztályok**

TectonImpl → FertileTectonImpl → MultiLayeredTectonImpl

- **Interfészek**

MultiLayeredTecton, RoundBeginSubscriber

- **Metódusok**
  - ***+MultiLayeredTecton():***
    konstruktor, a gombafonál kapacitást beállítja 3-ra
  - ***+accept(myceliumGrowthEvaluator: MyceliumGrowthEvaluator, mycelium: Mycelium): void***
    eldönti hogy a kapott gombafonál nőhet-e az adott tektonon

```
procedure accept(myceliumGrowthEvaluator : MyceliumGrowthEvaluator, mycelium :
Mycelium)

 currentMyceliaCount ← GET this.getMycelia.size
 capacity ← GET this.getMyceliaCapacity

 IF currentMyceliaCount >= capacity THEN BEGIN
      CALL mycelium.delete
      RETURN
 END
 ADD mycelium TO this.getMycelia
 sporeCount ← GET this.getSpores.size
 CALL mycelium.grow(sporeCount)
END
```

  - ***+accept(mushroomBodyGrowthEvaluator: MushroomBodyGrowthEvaluator, mushroomBody: MushroomBody): void***
    eldönti hogy a kapott gombatest nőhet-e az adott tektonon

```
procedure accept(mushroomBodyGrowthEvaluator : MushroomBodyGrowthEvaluator,
mushroomBody : MushroomBody)

 sporeCount ← GET this.getSpores.size
 hasExistingMushroomBody ← (GET this.getMushroomBody IS NOT NULL)
 hasMycelia ← (GET this.getMycelia.isEmpty IS FALSE)

 IF (sporeCount < 3) OR (hasExistingMushroomBody IS TRUE) OR (hasMycelia IS
      FALSE)  THEN BEGIN
            CALL muhsroomBody.delete
            RETURN
 END

 CALL this.setMushroomBody(mushroomBody)
 CALL mushroomBody.grow(sporeCount)
END
```

  - ***+onRoundBegin(): void***
    Itt történik a tektontörés és annak következményei

```
procedure onRoundBegin:

 currentBreakTimer ← GET this.getBreakTimer
 SET this.breakTimer ← currentBreakTimer − 1

 currentBreakTimer ← GET this.getBreakTimer
 IF currentBreakTimer <= 0 THEN BEGIN
      WHILE GET this.getMycelia.isEmpty IS FALSE DO BEGIN
            mycelium ← DEQUEUE FROM this.getMycelia
            IF mycelium IS NOT NULL THEN BEGIN
                  CALL mycelium.cutImmediate
```

```
            END
        END WHILE

        newFertileTecton ← CREATE new FertileTecton
        CALL newFertileTecton.addNeighbour(this)
        CALL this.addNeighbour(newFertileTecton)

    END

END
```

- ***sustaining(): boolean***
  Visszaadja, ha az adott tekton képes-e fenntartani gombafonalakat. MultiLayeredTectonnal ha van rajta gombatest akkor true-val tér vissza különben false

8.1.5 AridTectonImpl

- **Felelősség**

Az AridTecton típusú tektonok növesztésének feltételét szabályozza. A többi felelősségét az absztrakt Tecton osztálytól örökli.

- **Ősosztályok**

TectonImpl → FertileTectonImpl → AridTectonImpl

- **Interfészek**

AridTecton, RoundBeginSubscriber, TurnBeginSubscriber

- **Attribútumok**
  - **-absorbCountdown**: **int**
    Azt mutatja, hogy hány kör múlva szívja fel a gombafonalat a tekton

- **Metódusok**
  - ***+AridTecton():***
    konstruktor, a gombafonál kapacitást beállítja 1-re
  - ***+accept(myceliumGrowthEvaluator: MyceliumGrowthEvaluator, mycelium: Mycelium): void***
    eldönti hogy a kapott gombafonál nőhet-e az adott tektonon

```
procedure accept(myceliumGrowthEvaluator : MyceliumGrowthEvaluator, mycelium :
Mycelium)

 currentMyceliaCount ← GET this.getMycelia.size
 capacity ← GET this.getMyceliaCapacity

 IF currentMyceliaCount >= capacity THEN BEGIN
      CALL mycelium.delete
      RETURN
 END
 ADD mycelium TO this.getMycelia
 sporeCount ← GET this.getSpores.size
 CALL mycelium.grow(sporeCount)
 absorbCountdown ← 5
```

```
END
```

- ***+accept(mushroomBodyGrowthEvaluator: MushroomBodyGrowthEvaluator, mushroomBody: MushroomBody): void***
  eldönti hogy a kapott gombatest nőhet-e az adott tektonon

```
procedure accept(mushroomBodyGrowthEvaluator : MushroomBodyGrowthEvaluator,
mushroomBody : MushroomBody)

 sporeCount ← GET this.getSpores.size
 hasExistingMushroomBody ← (GET this.getMushroomBody IS NOT NULL)
 hasMycelia ← (GET this.getMycelia.isEmpty IS FALSE)

 IF (sporeCount < 3) OR (hasExistingMushroomBody IS TRUE) OR (hasMycelia IS
      FALSE)  THEN BEGIN
            CALL muhsroomBody.delete
            RETURN
 END

 CALL this.setMushroomBody(mushroomBody)
 CALL mushroomBody.grow(sporeCount)
END
```

- ***+onRoundBegin(): void***
  Itt történik a tektontörés és annak következményei

```
procedure onRoundBegin:

 currentBreakTimer ← GET this.getBreakTimer
 SET this.breakTimer ← currentBreakTimer − 1

 currentBreakTimer ← GET this.getBreakTimer
 IF currentBreakTimer <= 0 THEN BEGIN
      WHILE GET this.getMycelia.isEmpty IS FALSE DO BEGIN
            mycelium ← DEQUEUE FROM this.getMycelia
            IF mycelium IS NOT NULL THEN BEGIN
                  CALL mycelium.cutImmediate
            END
      END WHILE


      newFertileTecton ← CREATE new FertileTecton
      CALL newFertileTecton.addNeighbour(this)
      CALL this.addNeighbour(newFertileTecton)

 END

END
```

- ***+onTurnBegin(): void***
  Itt történik a tektonon lévő fonál elszáradása és így elpusztulása, ha az absorbCountdown eléri a 0-át

```
procedure onTurnBegin:

 IF absorbCountdown > 0 THEN BEGIN
      absorbCountdown ← absorbCountdown − 1
      IF absorbCountdown <= 0 THEN BEGIN
            mycelium DEQUEUE FROM this.getMycelia
            IF mycelium IS NOT NULL THEN BEGIN
                  CALL mycelium.delete
            END
      END
 END
END
```

- ***sustaining(): boolean***
  Visszaadja, ha az adott tekton képes-e fenntartani gombafonalakat. AridTectonnal ha van rajta gombatest akkor true-val tér vissza különben false

8.1.6 SustainingTectonImpl

- **Felelősség**

A SustainingTecton típusú tektonok növesztésének feltételét szabályozza. A többi felelősségét az absztrakt Tecton osztálytól örökli.

- **Ősosztályok**

TectonImpl → FertileTectonImpl → SustainingTectonImpl

- **Interfészek**

SustainingTecton, RoundBeginSubscriber

- **Metódusok**
  - ***+SustainingTecton():***
    konstruktor, a gombafonál kapacitást beállitja 1-re és a BreakTimer-jét is beállitja
  - ***+accept(myceliumGrowthEvaluator: MyceliumGrowthEvaluator, mycelium: Mycelium): void***
    eldönti hogy a kapott gombafonál nőhet-e az adott tektonon

```
procedure accept(myceliumGrowthEvaluator : MyceliumGrowthEvaluator, mycelium :
Mycelium)

 currentMyceliaCount ← GET this.getMycelia.size
 capacity ← GET this.getMyceliaCapacity

 IF currentMyceliaCount >= capacity THEN BEGIN
      CALL mycelium.delete
      RETURN
 END
 ADD mycelium TO this.getMycelia
 sporeCount ← GET this.getSpores.size
 CALL mycelium.grow(sporeCount)
END
```

  - ***+accept(mushroomBodyGrowthEvaluator: MushroomBodyGrowthEvaluator, mushroomBody: MushroomBody): void***
    eldönti hogy a kapott gombatest nőhet-e az adott tektonon

```
procedure accept(mushroomBodyGrowthEvaluator : MushroomBodyGrowthEvaluator,
mushroomBody : MushroomBody)

 sporeCount ← GET this.getSpores.size
 hasExistingMushroomBody ← (GET this.getMushroomBody IS NOT NULL)
 hasMycelia ← (GET this.getMycelia.isEmpty IS FALSE)

 IF (sporeCount < 3) OR (hasExistingMushroomBody IS TRUE) OR (hasMycelia IS
      FALSE)  THEN BEGIN
            CALL muhsroomBody.delete
            RETURN
```

```
 END

 CALL this.setMushroomBody(mushroomBody)
 CALL mushroomBody.grow(sporeCount)
END
```

- *+**onRoundBegin(): void***
  Itt történik a tektontörés és annak következményei

```
procedure onRoundBegin:

 currentBreakTimer ← GET this.getBreakTimer
 SET this.breakTimer ← currentBreakTimer − 1

 currentBreakTimer ← GET this.getBreakTimer
 IF currentBreakTimer <= 0 THEN BEGIN
      WHILE GET this.getMycelia.isEmpty IS FALSE DO BEGIN
            mycelium ← DEQUEUE FROM this.getMycelia
            IF mycelium IS NOT NULL THEN BEGIN
                  CALL mycelium.cutImmediate
            END
      END WHILE

      newFertileTecton ← CREATE new FertileTecton
      CALL newFertileTecton.addNeighbour(this)
      CALL this.addNeighbour(newFertileTecton)

 END

END
```

- ***sustaining(): boolean***
  Visszaadja, ha az adott tekton képes-e fenntartani gombafonalakat.
  SustainingTectonnal mindig true-val tér vissza

8.1.7 MushroomBodyImpl

- **Felelősség**

  A gombatestekért felelős osztály. A gombatest a spórák termeléséért és kilövéséért felelős. 3 spórakilövés után inaktívvá válik, amely abban nyilvánul meg, hogy a remaingEjects változó értéke 0 lesz. A gombatest az utolsó kilövése előtt csak valamely szomszédjára lőhet spórát. Fejlettnek az utolsó kilövése során minősül, amely abban nyilvánul meg, hogy ekkor a szomszédja szomszédjára is tud lőni.

- **Interfészek**

  Mushroom, MushroomBody, TurnBeginSubscriber

- **Attribútumok**

  - **-remainingEjects: int**

    A megmaradt spórakilövések számát tároló változó. Alapértelmezett értéke 3.

- **-location: Tecton**

  A gombatest elhelyezkedése szerinti tektont tároló változó.

- **-mushroomSpores: List<Spore>**

  A gombatest spóráit tartalmazó lista. A lista alapértelmezetten üres.

- **Metódusok**

  - **+MushroomBody(location: FertileTecton, name: String)**

    Konstruktor, amely beállítja a létrehozandó gombatest nevét és azt a tektont (céltekton), amelyen az elhelyezésre kerül. Ez a konstruktor használandó FertileTecton, továbbá a FertileTecton valamennyi leszármazottja, azaz AridTecton, MultiLayeredTecton és SustainingTecton esetén. A metódus pszeudokódja:

```
procedure MushroomBody(location: FertileTecton, name: String)
        // elmenti a céltektont
        SET this.location ← location

        // létrehoz egy MushroomBodyGrowthEvaluator példányt
        evaluator ← CREATE MushroomBodyGrowthEvaluator(this)

        // a feltételek fennállásának kiértékelése céljából meghívja a
        // a visit metódust
        CALL evaluator.visit(location, this)
END
```

  - **+MushroomBody(location: SemiFertileTecton, name: String)**

    Konstruktor, amely beállítja a létrehozandó gombatest nevét és azt a tektont (céltekton), amelyen az elhelyezésre kerül. Ez a konstruktor használandó SemiFertileTecton esetén. A metódus pszeudokódja:

```
procedure MushroomBody(location: SemiFertileTecton, name: String)
        // elmenti a céltektont
        SET this.location ← location

        // létrehoz egy MushroomBodyGrowthEvaluator példányt
        evaluator ← CREATE MushroomBodyGrowthEvaluator(this)

        // a feltételek fennállásának kiértékelése céljából meghívja a
        // a visit metódust
        CALL evaluator.visit(location, this)
END
```

  - **+MushroomBody()**

    Paraméter nélküli (default) konstruktor.

- **+delete(): void**

  A növekedési feltételek hiánya esetében kerül meghívásra az előzetesen létrehozott gombatest törlése céljából.

- **+grow(sporeCount: int): void**

  A gombatest növekedési folyamatát lezáró metódus, amelyet a Mushroom interfész miatt szükséges a gombatestnél ilyen formában megvalósítani. A paramétert a céltektontól kapja. A tekton abban az esetben hívja meg ezt a metódust (és nem a delete()-et), ha a gombatest növesztési feltételeire vonatkozó vizsgálat pozitív eredményt hozott. Ezért ez a metódus a gombatest esetében nem, csak a gombafonálnál bír jelentőséggel.

- **+onTurnBegin(): void**

  A gombatest minden új körének kezdetekor – beleértve a játék első körét is – a gombatestben egy új spóra termelődik. A spóra típusa véletlenszerűen kerül kiválasztásra. A metódus pszeudokódja:

```
procedure onTurnBegin()
        // Egy spóratípust véletlenszerűen kiválasztásra kerül
        random ← RANDOM NUMBER BETWEEN 1 AND 5

        IF random == 1 THEN
                newSpore ← CREATE SpitSpore()
        ELSE IF random == 2 THEN
                newSpore ← CREATE StunSpore()
        ELSE IF random == 3 THEN
                newSpore ← CREATE PreventCutSpore()
        ELSE IF random == 4 THEN
                newSpore ← CREATE SpeedSpore()
        ELSE
                newSpore ← CREATE SlownessSpore()
        END IF

        // Hozzáadja az új spórát a gombatest spóralistájához
        CALL this.addSpore(newSpore)
END
```

- **+getRemainingEjects(): int**

  Visszaadja a gombatest megmaradt spórakilövéseinek számát.

- **+setRemainingEjects(remainingEjects: int): void**

  Beállítja a gombatest megmaradt spórakilövéseinek számát.

- **+getSpores(): List<Spore>**

  Visszaadja a gombatest spóráit tartalmazó listát.

- **+addSpore(newSpore: Spore): void**

  Hozzáad egy új spórát a gombatest spóráinak listájához.

- **+ejectSpores(target: Tecton): void**

  A gombatest spóráinak kilövéséért felelős metódus. A metódus pszeudokódja:

```
procedure ejectSpores(target: Tecton)
    // Ha már volt 3 spórakilövése, a gombatest inaktív, nem tud aktivitást
    // kifejteni, így spórát sem lőhet ki (nincs is már neki)
    IF remainingEjects == 0 THEN
        RETURN // A gombatest inaktív, nem tud aktivitást kifejteni!
    END IF

    // Ha ez az utolsó, azaz a 3. spórakilövése, a gombatest fejlett állapotú,
    // így a céltekton lehet szomszéd vagy a szomszéd szomszédja
    IF remainingEjects == 1 THEN
        reachable ← EMPTY SET

        FOR EACH primary IN this.neighbours DO
            ADD primary TO reachable
            FOR EACH secondary IN primary.neighbours DO
                ADD secondary TO reachable
            END FOR
        END FOR

        IF target IS IN reachable THEN
            IF this.mushroomSpores IS NOT EMPTY
                ejectSpores(target)
                remainingEjects ← remainingEjects – 1
            ELSE
                RETURN // A gombatestnek nincsen kilőhető
                       // spórája!
            END IF
        ELSE
            RETURN // A céltekton túl messze van!
        END IF

    ELSE
    // A gombatest még nem fejlett, ezért csak közvetlen szomszédjára lőhet
    // spórát
        IF target IS IN this.neighbours THEN
            IF this.mushroomSpores IS NOT EMPTY
                ejectSpores(target)
                remainingEjects ← remainingEjects – 1
            ELSE
                RETURN // A gombatestnek nincsen kilőhető
                       // spórája!
            END IF
        ELSE
            RETURN // A céltekton túl messze van!
        END IF
    END IF
END
```

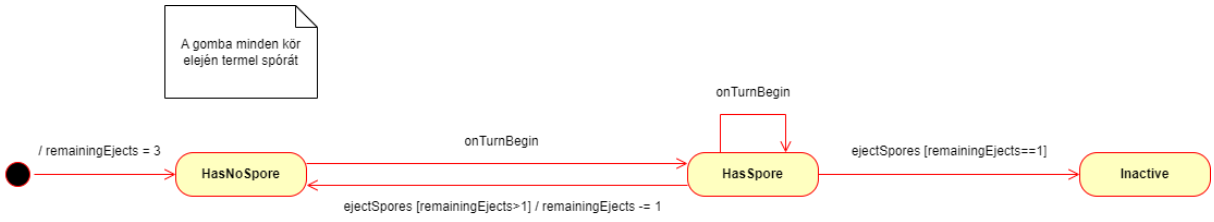
- **+getLocation(): Tecton**

  Visszaadja a gombatest elhelyezkedése szerinti tektont.

- **+setLocation(location: Tecton): void**

  Beállítja a gombatest elhelyezkedése szerinti tektont.

- **Állapot Diagramm**



8.1.8 MyceliumImpl

- **Felelősség**

A játékban a gombafonalakat reprezentáló osztály. Felelős a gombafonal növési folyamatban a növés gyorságáért, a gombafonalak elvágása esetén pedig a fonál kitörléséért és részben a többi fonál életben maradásának ellenőrzéséért is.

- **Interfészek**

Mushroom, Mycelium, TurnBeginSubscriber

- **Attribútumok**

- -growing: boolean
  Azt ábrázolja, hogy a gombafonál éppen növekedés alatt van-e. Alapértelmezett értéke hamis. Ha a tekton úgy dönt, hogy nőhet rajta a fonál akkor igaz lesz.
- -location: Tecton
  A tekton, ahol a gombafonál elhelyezkedik.
- -growTimer: int
  Az idő, ami alatt a gombafonál megnő. Alapértelmezett értéke 0. Növéskor, ha a céltektonon van spóra, akkor ez az érték 1, ha nincs akkor 2 lesz.
- -deathTimer:int
  Az idő, ami mulva elszakad a fonál. Alapértelmezett érték -1

- **Metódusok**

- +Mycelium(location: FertileTecton, name: String)
  Konstruktor, amely beállítja a létrehozandó gombafonál nevét és azt a tektont (céltekton), amelyen az elhelyezésre kerül. Ez a konstruktor használandó FertileTectonon, továbbá a FertileTecton valamennyi leszármazottjánál, azaz AridTecton, MultiLayeredTecton és SustainingTecton esetén.

```
procedure Mycelium(location: FertileTecton, name: String)
    // A fonál helyét beállítja a kapott helynek
    SET this.location ← location

    // létrejön egy új MyceliumGrowthEvaluator aminek paraméterként beadjuk a Myceliumot
    myceliumGrowthEvaluator ← CREATE MyceliumGrowthEvaluator(this)

    //A fonálnövesztés feltételeinek ellenőrzésére meghívjuk a myceliumGrowthEvaluator visit
    //függvényét, ami majd a location Tektonnal kommunikál
    CALL myceliumGrowthEvaluator.visit(location)
END
```

- +Mycelium(location: SemiFertileTecton, name: String)

Konstruktor, amely beállítja a létrehozandó gombafonál nevét és azt a tektont (céltekton), amelyen az elhelyezésre kerül. Ez a konstruktor használandó SemiFertileTectonon.

```
procedure Mycelium(location: SemiFertileTecton, name: String)
    // A fonál helyét beállítja a kapott helynek
    SET this.location ← location
```

// létrejön egy új MyceliumGrowthEvaluator aminek paraméterként beadjuk a Myceliumot
myceliumGrowthEvaluator ← CREATE MyceliumGrowthEvaluator(this)

//A fonálnövesztés feltételeinek ellenőrzésére meghívjuk a myceliumGrowthEvaluator visit //függvényét, ami majd a location Tektonnal kommunikál
CALL myceliumGrowthEvaluator.visit(location)
END

- +delete(): void
  Kitörli a gombafonalat
- +grow(sporeCount: int): void
  Ezt a jelzést a tekton fogja meghívni a gombafonálra, ha nőhet. A céltektonon lévő spóraszámtól függően módosítja a growtimert.
- +onTurnBegin(): void
  A growtimer visszaszámlálását végzi (minden körben eggyel csökken) és ha az lejárt, már nem lesz növésben a gombafonál.

  A deathTimer visszaszámlálását is végzi (minden körben eggyelc csökken), és ha az lejárt, meghivja a cutImmediate metódust
- +isGrowing(): boolean
  A growing attribútum getterje
- setGrowing(growing: boolean): void
  A growing attribútum setterje
- +cutImmediate(): void
  A gombafonál elvágódik, ezzel szól a többi gombafonálnak, hogy nézzék meg, hogy hozzá vannak-e csatlakoztatva gombatesthez

  Pszeudokód

```
Procedure cutImmediate:
	this.delete()
	location.checkNeighbourMyceliaSustain()
	if(location.getMycelia IS EMTPY) begin
		List<Insect> temp
		FOR EACH i IN location.getOccupants begin
			ADD i TO temp
		end
		FOR EACH i IN temp begin
			i.runAway()
		end
	end
end procedure
```

- +cutWithDelay(): void

  Beállítja a deathTimer-t 2-re

- +getLocation(): Tecton
  A location attribútum getterje
- +setLocation(location: Tecton): void
  A location attribútum getterje

8.1.9 CarnivorousMycelium

- **Felelősség**

Az alapvető gombafonál funkciókon kívül speciális feltételek között a rovarak evését és fonál növesztését megvalósító osztály.

- **Interfészek**

Mushroom, Mycelium, TurnBeginSubscriber

- **Attribútumok**

- -growing: boolean
  Azt ábrázolja, hogy a gombafonál éppen növekedés alatt van-e. Alapértelmezett értéke hamis. Ha a tekton úgy dönt, hogy nőhet rajta a fonál akkor igaz lesz.
- -location: Tecton
  A tekton, ahol a gombafonál elhelyezkedik.
- -growTimer: int
  Az idő, ami alatt a gombafonál megnő. Alapértelmezett értéke 0. Növéskor, ha a céltektonon van spóra, akkor ez az érték 1, ha nincs akkor 2 lesz.
- -deathTimer:int
  Az idő, ami mulva elszakad a fonál. Alapértelmezett érték -1

- **Metódusok**

- +Mycelium(location: FertileTecton, name: String)
  Konstruktor, amely beállítja a létrehozandó gombafonál nevét és azt a tektont (céltekton), amelyen az elhelyezésre kerül. Ez a konstruktor használandó FertileTectonon, továbbá a FertileTecton valamennyi leszármazottjánál, azaz AridTecton, MultiLayeredTecton és SustainingTecton esetén.

```
procedure CarnivorousMycelium(location: FertileTecton, name: String)
    // A fonál helyét beállítja a kapott helynek
    SET this.location ← location

    // létrejön egy új MyceliumGrowthEvaluator aminek paraméterként beadjuk a Myceliumot
    myceliumGrowthEvaluator ← CREATE MyceliumGrowthEvaluator(this)

    //A fonálnövesztés feltételeinek ellenőrzésére meghívjuk a myceliumGrowthEvaluator visit
    //függvényét, ami majd a location Tektonnal kommunikál
    CALL myceliumGrowthEvaluator.visit(location)
END
```

- +CarnivorousMycelium(location: SemiFertileTecton, name: String)

Konstruktor, amely beállítja a létrehozandó gombafonál nevét és azt a tektont (céltekton), amelyen az elhelyezésre kerül. Ez a konstruktor használandó SemiFertileTectonon.

```
procedure Mycelium(location: SemiFertileTecton, name: String)
    // A fonál helyét beállítja a kapott helynek
    SET this.location ← location
```

// létrejön egy új MyceliumGrowthEvaluator aminek paraméterként beadjuk a Myceliumot
myceliumGrowthEvaluator ← CREATE MyceliumGrowthEvaluator(this)

//A fonálnövesztés feltételeinek ellenőrzésére meghívjuk a myceliumGrowthEvaluator visit
//függvényét, ami majd a location Tektonnal kommunikál
CALL myceliumGrowthEvaluator.visit(location)
END

- +delete(): void
  Kitörli a gombafonalat
- +grow(sporeCount: int): void
  Ezt a jelzést a tekton fogja meghívni a gombafonálra, ha nőhet. A céltektonon lévő spóraszámtól függően módosítja a growtimert.
- +onTurnBegin(): void
  A growtimer visszaszámlálását végzi (minden körben eggyel csökken) és ha az lejárt, már nem lesz növésben a gombafonál. Ha a tektonján levő rovarok Stunned állapotban vannak megöli a rovarokat és egy gombát növesztését kezdeményezi.

  A deathTimer visszaszámlálását is végzi (minden körben eggyelc csökken), és ha az lejárt, meghivja a cutImmediate metódust
- +isGrowing(): boolean
  A growing attribútum getterje
- setGrowing(growing: boolean): void
  A growing attribútum setterje
- +cutImmediate(): void
  A gombafonál elvágódik, ezzel szól a többi gombafonálnak, hogy nézzék meg, hogy hozzá vannak-e csatlakoztatva gombatesthez

  Pszeudokód

```
Procedure cutImmediate:
	this.delete()
	location.checkNeighbourMyceliaSustain()
	if(location.getMycelia IS EMTPY) begin
		List<Insect> temp
		FOR EACH i IN location.getOccupants begin
			ADD i TO temp
		end
		FOR EACH i IN temp begin
			i.runAway()
		end
	end
end procedure
```

- +cutWithDelay(): void

  Beállítja a deathTimer-t 2-re

- +getLocation(): Tecton
  A location attribútum getterje
- +setLocation(location: Tecton): void
  A location attribútum getterje

8.1.10 MyceliumGrowthEvaluator

- **Felelősség**

Egy segédosztály, ami annak az eldöntésében segít, hogy egy adott tektonra nőhet-e az őt létrehozó gombafonál.

- **Interfészek**

TectonVisitor

- **Attribútumok**

  - **-creator: Mycelium**

    Ezt az objektumot létrehozó gombafonál.

- **Metódusok**

- + MyceliumGrowthEvaluator(mushroom: Mycelium)
  Létrehozáskor meg kell adni, hogy melyik gombafonál hozta őt létre.
- +visit(tecton: FertileTecton): void
  Megkér egy "Fertile" típusú tektont, hogy "fogadja be" az objektumot létrehozó gombafonalat. A tekton ezt a kérést az accept függvényében elbírálja és visszajelez a fonálnak.
- +visit(tecton: MultiLayeredTecton): void
  Megkér egy "MultiLayered" típusú tektont, hogy "fogadja be" az objektumot létrehozó gombafonalat. A tekton ezt a kérést az accept függvényében elbírálja és visszajelez a fonálnak.
- +visit(tecton: AridTectton): void
  Megkér egy "Arid" típusú tektont, hogy "fogadja be" az objektumot létrehozó gombafonalat. A tekton ezt a kérést az accept függvényében elbírálja és visszajelez a fonálnak.
- +visit(tecton: SemiFertileTecton): void
  Megkér egy "SemiFertile" típusú tektont, hogy "fogadja be" az objektumot létrehozó gombafonalat. A tekton ezt a kérést az accept függvényében elbírálja és visszajelez a fonálnak.
- +visit(tecton: SustainingTecton): void
  Megkér egy "Sustaining" típusú tektont, hogy "fogadja be" az objektumot létrehozó gombafonalat. A tekton ezt a kérést az accept függvényében elbírálja és visszajelez a fonálnak.
- +getCreator(): Mycelium
  A creator attribútum getterje

8.1.11 MushroomBodyGrowthEvaluator

- **Felelősség**

Egy segédosztály, ami annak az eldöntésében segít, hogy egy adott tektonra nőhet-e az őt létrehozó gombatest.

- **Interfészek**

TectonVisitor

- **Attribútumok**

  - **-creator: MushroomBody**
    Ezt az objektumot létrehozó gombafonál.

- **Metódusok**

- + MushrooomBodyGrowthEvaluator(mushroom: MushroomBody)
  Létrehozáskor meg kell adni, hogy melyik gombatest hozta őt létre.
- +visit(tecton: FertileTecton): void
  Megkér egy "Fertile" típusú tektont, hogy "fogadja be" az objektumot létrehozó gombatestet. A tekton ezt a kérést az accept függvényében elbírálja és visszajelez a gombatestnek.
- +visit(tecton: MultiLayeredTecton): void
  Megkér egy "MultiLayered" típusú tektont, hogy "fogadja be" az objektumot létrehozó gombatestet. A tekton ezt a kérést az accept függvényében elbírálja és visszajelez a gombatestnek.
- +visit(tecton: AridTectton): void
  Megkér egy "Arid" típusú tektont, hogy "fogadja be" az objektumot létrehozó gombatestet. A tekton ezt a kérést az accept függvényében elbírálja és visszajelez a gombatestnek.
- +visit(tecton: SemiFertileTecton): void
  Megkér egy " SemiFertile" típusú tektont, hogy "fogadja be" az objektumot létrehozó gombatestet. A tekton ezt a kérést az accept függvényében elbírálja és visszajelez a gombatestnek.
- +visit(tecton: SustainingTecton): void
  Megkér egy " Sustaining" típusú tektont, hogy "fogadja be" az objektumot létrehozó gombatestet. A tekton ezt a kérést az accept függvényében elbírálja és visszajelez a gombatestnek.
- +getCreator(): MushroomBody
  A creator attribútum getterje

8.1.12 InsectImpl

- **Felelősség**

Egy rovarral kapcsolatos adatok keze

- **Interfészek**

Insect, InsectMover, SporeEater, TurnBeginSubscriber

- **Attribútumok**
  - **-location: Tecton**
    A tekton, amin a rovar van
  - **-maxMoves: int**
    Egy körön belüli maximális lépéseinek száma
  - **-remainingMoves: int**
    Az aktuális körben maradt lépések száma
  - **-sporesEaten: int**
    Megevett spórák száma
  - **-effectTimer: int**
    Ha van spórából származó állapot a rovaron, ez a visszaszámláló, hogy mikor jár le ennek az ideje
  - **-state: InsectState**
    Az aktuális állapota a rovarnak
  - **-splitNum: int**
    Az szakadások száma

- **Metódusok**
  - +**Insect(t: Tecton):** Insect konstruktora. Az insect a t tectonra fog létrejönni. Alapértelmezett értékek:
    -maxMoves: 2
    -remainingMoves: maxMoves
    -sporesEaten: 0
    -effectTimer: 0
    -state: Normal
  - **+Tecton getLocation():** visszaadja a location-t
  - **+setLocation(Tecton t):** beállítja a location-t
  - **+int getMaxMoves():** visszaadja a MaxMoves-t
  - **+setMaxMoves(i: int):** beállítja a maxMoves-t
  - **+int getRemainingMoves():** visszaadja a remainingMoves-t
  - **+setRemainingMoves(i: int):** beállítja a remainingMoves-t
  - **+int getSporesEaten():** visszaadja a sporesEaten-t
  - **+setSporesEaten(i: int):** beállítja a sporesEaten-t
  - **+int getEffectTimer():** visszaadja az effectTimer-t
  - **+setEffectTimer(i: int):** beállítja az effectTimer-t
  - **+InsectState getState():** visszaadja a state-t
  - **+setState(newState: InsectState):** beállítja a state-t
  - **+int getSplitNum():** visszaadja a splitNum-ot
  - **+setSplitNum(i: int):** beállítja a splitNum-ot

- **+cutMycelium():** elvágja az első gombafonalat a tectonján – meghívja a tektonon a cutMycelium() műveletet
- **+eatSpore(): megpróbálja** megenni a legelső spórát a tectonján - meghívja a tektonon a eatSpore(i: Insect) műveletet, ahol i önmaga lesz
- **+move (t: Tecton**): megpróbál elmenni a t tectonra – meghívja a tektonon a moveInsect(i: Insect) műveletet, ahol i önmaga lesz
- **+onTurnBegin():** a játékos körének elején történő dolgoknak létezik;
  Pszeudokód:

```
Procedure onTurnBegin():
	IF effectTimer BIGGER THAN 0 begin
		effectTimer = effectTimer – 1
		IF effectTimer IS SMALLER THAN OR EQUAL TO 0 begin
			this.setState(InsectState.Normal)
			this.setMaxMoves(2)
		end
	end
	setRemainingMoves(getMaxMoves())
end procedure
```

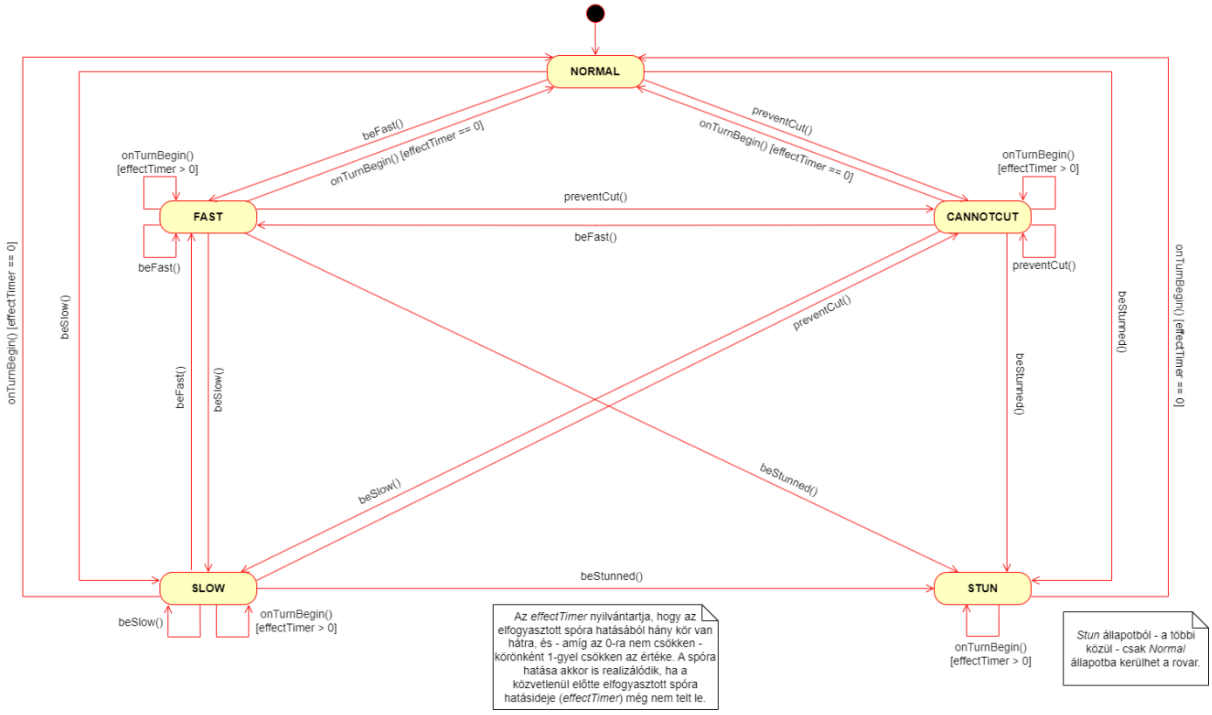
- **+beSlow():** Slow állapotba állítja a rovart, és beállítja a maxMoves-t 1-re
- **+beFast():** Fast állapotba állítja a rovart, és beállítja a maxMoves-t 3-ra
- **+preventCut():** CannotCut állapotba állítja a rovart
- **+beStunned():** Stun állapotba állítja a rovart, és beállítja a maxMoves-t 0ra
- **+split():** Kettészakítja a rovart (létrejön egy új a tectonján)
- +**runAway():** elmenekül egy, véletlenszerűen kiválasztott, alkalmas tektonra
  Pszeudokód:

```
Procedure runAway():
	Set<Tecton> available
	Queue<Tecton> queue
	Set<Tecton> visited

	ENQUEUE getLocation() INTO queue
	ADD getLocation() TO visited

	WHILE queue IS NOT EMPTY begin
		Tecton current = DEQUEUE from queue

		Boolean hasMycelium
		hasMycelium = current.hasMycelium()



		IF hasMycelium IS TRUE begin
			ADD current TO available
		End

		FOR EACH neighbour IN location.getNeighbours() begin
			IF (ADD neighbour TO visited) IS TRUE begin
				ENQUEUE neighbour INTO queue
			end
		end
	end

	IF available IS EMTPY begin
		return
	end
```

```
        Tecton selectedTecton = NULL

        int item = RANDOM int FROM BETWEEN 0 and (SIZE OF available)
        int i = 0
        FOR EACH tecton IN available begin
              IF i IS EQUAL TO item begin
                    selectedTecton = tecton
              end
        end

        IF selectedTecton IS NOT NULL) begin
              setLocation(selectedTecton)

              selectedTecton.addOccupant(this)
        end
end procedure
```

- **+die():** ha Stun állapotban van, meghal (eltűnik a tektonjáról, és a location-je is null lesz, de mint object, megmarad)

- **Állapot Diagramm**



8.1.13 SplitSpore

- **Felelősség**

Ha megette egy rovar, kettészakítja ezt.

- **Interfészek**

Spore

- **Metódusok**
  - **+eatSpore(i: Insect):** az i rovar megette a spórát, ez meg fogja hívni a rovaron a Split() parancsot, hogy szakadjon el

8.1.14 StunSpore

- **Felelősség**

Ha megette egy rovar, lebénítja ezt.

- **Interfészek**

Spore

- **Metódusok**
  - **+eatSpore(i: Insect):** az i rovar megette a spórát, ez meg fogja hívni a rovaron a beStunned() parancsot, hogy kerüljön Stun állapotba

8.1.15 PreventCutSpore

- **Felelősség**

Ha megette egy rovar, ez ne tudjon fonalat vágni.

- **Interfészek**

Spore

- **Metódusok**
  - **+eatSpore(i: Insect):** az i rovar megette a spórát, ez meg fogja hívni a rovaron a PreventCut() parancsot, hogy kerüljön CannotCut állapotba

8.1.16 SpeedSpore

- **Felelősség**

Ha megette egy rovar, legyen több lépése

- **Interfészek**

Spore

- **Metódusok**
  - **+eatSpore(i: Insect):** az i rovar megette a spórát, ez meg fogja hívni rajta a beFast() parancsot, hogy nagyobb legyen a remainingMoves-ja, és kerüljön Fast állapotba.

8.1.17 SlownessSpore

- **Felelősség**

Ha megette egy rovar, legyen kevesebb lépése.

- **Interfészek**

Spore

- **Metódusok**
  - **+eatSpore(i: Insect):** az i rovar megette a spórát, ez meg fogja hívni rajta a beSlow() parancsot, hogy kisebb legyen a remainingMoves-ja, és kerüljön Slow állapotba.

8.1.18 PlayerImpl

- **Felelősség**

Egy játékos állapotával kapcsolatos adatok kezelése.

- **Interfészek**
  ScoreEvaluable, Player, TurnObserver

- **Attribútumok**
  - **~onTurnBeginSubscirbers: List<OnTurnBeginSubscriber>**
    Azoknak az objektumoknak az összegsége, amelyek értesítést szeretnének arról, hogy jelen játékos köre elkezdődött.
  - **~name**: String
    Jelen játékos neve, amellyel azonosítható.

- **Metódusok**
  - **+PlayerImpl(name: String)**
    Konstruktor, a létrehozáshoz a játékos neve szükséges.
  - **+subscibe(subscriber: OnTurnBeginSubscriber)**
    A megadott feliratkozó értesítést kér, arról, hogy jelen játékos köre elkezdődött.
  - **+setName(name: String)**
    A név setterje.
  - **+getName() : String**
    A név getterje.
  - **+notifySubscribers()**
    A feliratkozókat értesíti.
  - ***+calculateScore() : int***
    A játékos pontszámát kiszámolja, majd visszatér vele.

8.1.19 MycologistImpl

- **Felelősség**

Egy gombász játékos állapotát tárolja.

- **Interfészek**
  Mycologist

- **Ősosztályok**
  PlayerImpl

- **Attribútumok**
  - **-insects**: List<Insect>
    A játékos által irányított rovarok.

- **Metódusok**
  - **+addInsect(i: Insect)**
    Egy rovart ad hozzá a játékoshoz, amit irányítani tud.
  - **+removeInsect()**
    Egy játékos által irányított rovart elvesz a játékostól.
  - **+ownsInscect(i: Insect)**
    Megadja, hogy egy rovar a játékoshoz tartozik-e.
  - **+calculateScore() : int**
    A saját rovarjai által megevett spórák összegével tér vissza.

(Implementálja az ős osztályokból fakadó egyéb metódusokat**)**

8.1.20 EntomologistImpl

- **Felelősség**

Egy rovarász játékos állapotát tárolja.

- **Interfészek**
  Entomolgist

- **Ősosztályok**
  PlayerImpl

- **Attribútumok**
  - **-mycelia: List<Mycelium>**
    A gombász gombafonalai.
  - **-mushroomBodies: List<MushroomBody>**
    A gombász gombatestei

- **Metódusok**
  - **+addMycelium(mycelium: Mycelium)**
    Hozzáad egy gombafonalat a gombász saját gombafonalaihoz.
  - **+removeMycelim(mycelium: Mycelium)**
    Elvesz egy gombafonalat a gombásztól.
  - **+ownsMycelium(mycelium: Mycelium)**
    Megmondja, hogy a játékoshoz tartozik-e egy gombafonál.
  - +**addMushroomBody(mushroomBody: MushroomBody)**
    Hozzáad egy gombatest a gombász saját gombatesteihez.
  - **+removeMushroomBody(mushroomBody: MushroomBody)**
    Elvesz egy gombatestet a gombásztól.
  - **+ownsMushroomBody(mushroomBody: MushroomBody)**
    Megmondja, hogy a játékoshoz tartozik-e egy gombatest.

(Implementálja az ős osztályokból fakadó egyéb metódusokat**)**

8.1.21 InputCommand

- **Felelősség**

Tárolja a megadott parancsot és argumentumaikat.

- **Attribútumok**
  - **+commandName: String**
    A parancs neve, ami alapján egyértelműen azonosítható.
  - **+commandParams: List<String>**
    A parancs paraméterei.

- **Metódusok**
  - **+InputCommand(name: String, params: List<String>)**
    A változóit inicializáló konstruktor.

8.1.22 CommandImpl

- **Felelősség**

A parancsot olyan formában tartalmazza, hogy egy megfelelő kezelővel futtatható legyen.

- **Interfészek**
  Command

- **Attribútumok**
  - **~input: InputCommand**
    A megadott parancs.
  - **~actingPlayer: Player**
    A játékos, akinek éppen köre van, ha ez irreleváns a parancshoz, akkor null. Szükséges a parancs helyességének ellenőrzéséhez.

- **Metódusok**
  - **+CommandImpl(actingPlayer: Player, inputCommand: InputCommand)**
    Konstruktor, beállítja a mostani játékost és a parancs adatait.
  - ***+execute(commandHandler: CommandHandler)***
    A parancsot teljesíti a megadott handler által.
  - **+getName() : String**
    Visszaadja a parancs nevét.

(Minden parancsnak van megfelelő CommandImpl osztályból származó saját implantációja, mivel ezekben lényeges változtatás nincs, amit itt fel lehetne tüntetni ezért ebből a dokumentumból olvashatósága megőrzése érdekéből ezeket itt kihagyom.)

8.1.23 CommandFactoryImpl

- **Felelősség**

A parancsok példányosítása.

- **Interfészek**
  CommandFactory

- **Metódusok**
  - **+createCommand(type: String): Command**
    Példányosít egy parancsot a megadott típussal.

8.1.24 PlayerContainerImpl

- **Felelősség**

A játékosok tárolásáért felelős.

- **Interfészek**
  PlayerProvider, PlayerMutator

- **Attribútumok**
  - **-players: List<Player>**
    A játékosoknak listája.
  - **-mycologists: List<Player>**
    A gombászok listája.
  - **-entomologists: List<Player>**
    A rovarászok listája.
  - **-currentIndex: int**
    A mostani játékosnak az indexe.

- **Metódusok**
  - **+addPlayer(player: Player, type: String)**
    Hozzáad egy játékost a játékosok listájához, és a megadott típus alapján a megfelelő tárolóban. (A létrehozó parancsban ez a típus meg van adva, ezért itt nem szükséges dinamikus típus lekérdezés.)
  - **+removePlayer(player: Player)**
    Kivesz egy játékost a játékosok listájából.
  - **+getNextPlayer() : Player**
    A következő játékost visszaadja, és a mostani játékos a listában a következő lesz.
  - **+getCurrentPlayer(): Player**
    Visszaadja a jelenlegi játékost.
  - **+getPlayers(): Iterable<Player>**
    Visszaadja a listáját a játékosoknak.
  - **+getMycologists(): Iterable<Player>**
    Visszaadja a gombászok listáját.
  - **+getEntomologists(): Iterable<Player>**
    Visszaadja a rovarászok listját.

8.1.25 PlayerFactoryImpl

- **Felelősség**

A játékosok példányosítása

- **Interfészek**
  PlayerFactory

- **Metódusok**
  - **+createPlayer(type: String, name: String): Player**
    Példányosít egy játékost a megadott típussal és névvel.

8.1.26 PlayerControllerImpl

- **Felelősség**
  A játékosok létrehozásának irányítása.

- **Interfészek**
  PlayerFactory, CommandHandler

- **Attribútumok**
  - **-playerContainer: PlayerMutator**
    A játékosokat tároló objektum
  - **-playerFactory: PlayerFactory**
    A játékosokat példányosító objektum

- **Metódusok**
  - **+PlayerControllerImpl(factory: PlayerFactory, container: PlayerMutator)**
    Beállítja a példányosító és tároló objektumot.
  - **+handleCommand(command: Command)**
    Kezeli a játékosok létrehozásával kapcsolatos parnacsokat.
  - **+createPlayer(String type, String name)**
    A létrehoz egy játékos példány

8.1.27 RoundObserverImpl

- **Felelősség**

Értesíteni a feliratkozóit arról, hogy a játékkörök körbeértek.

- **Interfészek**
  RoundObserver

- **Attribútumok**
  - **-onRoundBeginSubscirbers: List<OnRoundBeginSubscriber>**
    Az értesítendő objektumok listája.

- **Metódusok**
  - **+subscribe(subscriber: OnRoundBeginSubscriber)**
    Az adott objektum jelentkezik, hogy szeretne értesítést arról, hogy a játékosok köre körbeért.
  - **+notifySubscribers()**
    Értesíti a feliratkozókat.

8.1.28 TurnControllerImpl

- **Felelősség**

A körök elkezdését meghatározza.

- **Interfészek**
  TurnController, TurnInitializer, CommandHandler

- **Attribútumok**
  - **-playerContainer**: **PlayerProvider**
    A játékosok listája.
  - **-roundObserver: RoundObserver**
    Az objektum, ami arról értesít más objektumokat, hogy a játékosok körei véget értek.

- **Metódusok**
  - **+TurnControllerImpl(conatiner: PlayerProvider, observer: RoundObserver)**
    Konstruktor, szükséges megadni egy tárolót és egy figyelő objektumot, ki értesíti a feliratkozóit, ha a körök körbeértek.
  - **+handleCommand(command: Command)**
    A körök kezelésével kapcsolatos parancsokat kezeli.
  - **+endTurn()**
    A mostani játékos körét befejezi.
  - **+beginFirstTurn()**
    Elkezd egy kört, anélkül, hogy a jelenlegit befejezné.
  - **+getCurrentPlayer()**
    Visszaadja azt a játékost, akinek éppen aktív köre van.

8.1.29 ScoreCalculatorImpl

- **Felelősség**

Eldönteni, hogy melyik játékos nyerte meg a játékot.

- **Interfészek**

ScoreCalculator

- **Metódusok**
  - **+determineWinner(canidates: Iterable<ScoreEvaluable>) : Iterable<ScoreEvaluable>**
    Visszaadja a játékosokat, akiknek a pontjai a legmagasabbak.

    Pszeudókód:

```
determineWinner(canidates: Iterable<ScoreEvaluable>):
	winners: List<ScoreEvaluable>
	scores: List<int>
	for canidate in canidates do begin
		scores.add(candiate.calculateScore())
	end
	for canidate in canidates do begin
		if score = scores.max() then begin
			winners.add(canidate)
		end
	end
	return winners
```

8.1.30 GameEndManagerImpl

- **Felelősség**

A játék végét vezérlő osztály.

- **Interfészek**
  OnRoundBeginSubscriber, GameEndManager, GameLengthSetter

- **Attribútumok**
  - **-scoreCalculator: ScoreCalculator**
    Az osztály amelyik kiszámolja, hogy ki a győztes.
  - **-gameLength: int**
    A játéknak a hossza, azaz a játék végéhez hányszor érje

- **Metódusok**
  - **+GameEndManagerImpl(scoreCalculator: ScoreCalculator)**
    Konstruktor, szükséges egy objektum ami eldönti, hogy ki a győztes.
  - **+onRoundBegin()**
    Mindig amikor a játékosok köre körbér, a játék maradék ideje eggyel csökken.
  - **+setGameLength(newLength: int)**
    A játéknak hosszának setterje.
  - **+getGameLength(): int**
    Játék hosszának getterje.
  - **+showWinners()**
    Kiírja a győzteseket

8.1.31 GameControllerImpl

- **Felelősség**
  A játék menetével elindításért

- **Interfészek**
  GameController, CommandHandler

- **Attribútumok**
  - **-turnController: TurnInitializer**
    A köröket irányító kontroller, amit a játékkezdetekor elindít.
  - **-gameEndManager: GameLengthSetter**
    A játék végét számontartó objektum, a játék kezdetekor a visszaszámlálása elindul.

- **Metódusok**
  - **+GameControllerImpl(turnIntializer: TurnInitializer, gameLengthSetter: GameLengthSetter)**
    Konstruktor, meg kell adni az osztályt ami elkezdi számolni a köröket és ami beállítja a játék hosszát.
  - **+handleCommand(command: Command)**
    Kezeli a játék kezdésével vagy befejezésével kapcsolatos parancsokat.
  - **+beginGame(length: int)**
    A játék elkezdődik a megadott megadott kör limittel.
  - **+endGame()**
    Befejezi a játékot.

8.1.32 InsectControllerImpl

- **Felelősség**

A rovarokat irányítása.

- **Interfészek**

InsectController, CommandHandler

- **Metódusok**
  - **+handleCommand(command: Command)**
    Kezeli a rovar irányításával kapcsolatos parancsokat.
  - **+cut(insect: Insect)**
    A kiválasztott rovarral elvágja a tektont.
  - **+eat(insect: Insect)**
    A kiválasztott rovarral eszik.
  - **+move(insect: Insect, destination: Tecton)**
    A kiválasztott rovart mozgatja.

8.1.33 MyceliumFactoryImpl

- **Felelősség**

Gombafonalak példányosítása, úgy, hogy a példányosítás a növesztés feltételei szerint történjen.

- **Interfészek**

MyceliumFactory

- **Metódusok**
  - **+createMycelium(type: String, name: String, location: Tecton): Mycelium**
    A növesztési feltételek követésével növeszt egy gomafonalat, a megadott tektonra, a megadott névvel.

8.1.34 CheatMyceliumFactory

- **Felelősség**

Gombafonalak példányosítása, úgy, hogy a növesztés feltételeit nem veszik figyelembe.

- **Interfészek**

MyceliumFactory

- **Metódusok**
  - **+createMycelium (type: String, name: String, location: Tecton): Mycelium**
    A növesztési feltételek nélkül növeszt egy gombafonalat, a megadott tektonra, a megadott névvel.

8.1.35 DefaultMushroomBodyFactory

- **Felelősség**

Gombatestek példányosítása, úgy, hogy a példányosítás a növesztés feltételei szerint történjen.

- **Interfészek**

MushroomBodyFactory

- **Metódusok**
  - +**createMushroomBody(name: String, location: Tecton) : MushroomBody**
    A növesztési feltételek követésével növeszt egy gombatestet, a megadott tektonra, a megadott névvel.

8.1.36 CheatMushroomBodyFactory

- **Felelősség**

Gombatestek példányosítása, úgy, hogy a növesztés feltételeit nem veszik figyelembe.

- **Interfészek**

MushroomBodyFactory

- **Metódusok**
  - +**createMushroomBody(name: String, location: Tecton) : MushroomBody**
    A növesztési feltételek nélkül növeszt egy gombatestet, a megadott tektonra, a megadott névvel.

8.1.37 MushroomBodyControllerImpl

- **Felelősség**

Gombatestek irányítása.

- **Interfészek**
  MushroomBodyController, CommandHandler

- **Attribútumok**
  - **-sporeFactory: SporeFactory**
    A spórát példányosító objektum.

- **Metódusok**
  - **+MushroomBodyControllerImpl(factory: SporeFactory)**
    Konstruktor. Szükséges egy spórákat példányosító objektum megadása.
  - **+eject(source: MushroomBody, target: Tecton)**
    A megadott gombatest spóráit kilöveti a megadott tektonra.
  - **+deactivate(mushroomBody: MushroomBody)**
    A megadott gombatestet deaktiválja, azaz már nem lőhet ki több spórát.
  - **+addSpores(sporeType: String, sporeName: String, target: MushroomBody)**
    Egy új spórát hozzáad a gombatesthez, a megadott paraméterek alapján.

8.1.38 TectonFactoryImpl

- **Felelősség**

Tektonok példányosítása

- **Interfészek**

TectonFactory

- **Metódusok**
  - **+create(type: String, name: String): Tecton**
    Példányosít egy tektont a megadott típussal és névvel.

8.1.39 TectonControllerImpl

- **Felelősség**

Tektonok vezérlése

- **Interfészek**

TectonController, CommandHandler

- **. Attribútumok**
  - **-sporeFactory: SporeFactory**
    A spórát példányosító objektum.

- **Metódusok**
  - **+TectonControllerImpl(factory: SporeFactory)**
    Konstruktor. Szükséges egy spórákat példányosító objektum megadása.
  - **+handleCommand(command: Command)**
    Kezeli a tektonok kezelésével kapcsolatos parancsokat.
  - **+break(tecton: Tecton)**
    Eltöri a megadott tektont.
  - **+setBreakTimer(tecton: Tecton, time: int)**
    A törési időzítőt a megadott értékre.
  - **+addNeighbour(tecton1: Tecton, tecton2: Tecton)**
    A megadott két megadott tektont egymással szomszédossá teszi.
  - **+putSpore(sporeType: String, sporeName: String, target: Tecton)**
    Egy új spórát hozzáad a tektonhoz, a megadott paraméterek alapján.

8.1.40 InsectFactoryImpl

- **Felelősség**

Rovarok példányosítása

- **Interfészek**

InsectFactory

- **Metódusok**
  - **+create(name: String) : Insect**
    Példányosít egy rovart a megadott névvel.

8.1.41 SporeFactoryImpl

- **Felelősség**

Rovarok példányosítása

- **Interfészek**

SporeFactory

- **Metódusok**
  - **+create(name: String, type: String) : Spore**
    Példányosít egy spórát a megadott névvel és típussal.

8.1.42 MapCreationControllerImpl

- **Felelősség**

A játéktér létrehozásának irányítása.

- **Interfészek**

MapCreationController, CommandHandler

- **Attribútumok**
  - **-mushroomBodyFactory: MushroomBodyFactory**
    A gombatesteket a térkép létrehozásakor példányosító objektum.
  - **-myceliumFactory: MyceliumFactory**
    A gombafonalakat a térkép létrehozásakor példányosító objektum.
  - **-insectFactory: InsectFactory**
    A rovarokat a térkép létrehozásakor példányosító objektum.

- **Metódusok**
  - **+MapCreationControllerImpl(**
    mushroomBodyFactory: MushroomBodyFactory,
    myceliumFactory: MyceliumFactory, insectFactory: InsectFactory)
    Konstruktor. Meg kell adni az objektumokat, amelyek példányosítják a gombatesteket, gombafonalakat és a rovarokat.
  - **+handleCommand(command: Command)**
    A térkép létrehozásával kapcsolatos parancsokat kezeli.
  - **+createMycelium(name: String, type: String, location: Tecton)**
    Létrehoz egy gombafonalat a megadott paraméterek alapján.
  - **+createMushroomBody(name: String, location: Tecton)**
    Létrehoz egy gombatestet a megadott paraméterek alapján.
  - +**createInscet(name: String)**
    Létrehoz egy rovart a megadott paraméterek alapján.

8.1.43 GrowthControllerImpl

- **Felelősség**

A gombarészek növekedésének irányítása.

- **Interfészek**

GrowthController, CommandHandler

- **Attribútumok**
  - **-mushroomBodyFactory: MushroomBodyFactory**
    A gombatestet példányosító objektum.
  - **-myceliumFactory: MushroomBodyFactory**
    A gombafonalat példányosító objektum.

- **Metódusok**
  - **+GrowthControllerImpl(mushroomBodyFactory: MushroomBodyFactory myceliumFactory: MushroomBodyFactory)**
    Konstruktor. Meg kell adni az objektumokat, amelyek példányosítják a gombatesteket és a gombafonalakat.
  - **+handleCommand(command: Command)**
    Kezeli a növesztéssel kapcsolatos parancsokat.
  - **+growMycelium(name: String, location: Tecton)**
    Növeszt egy gombafonalat a megadott tektonra, a megadott névvel.
  - **+growMushroomBody(name: String, location: Tecton)**
    Növeszt egy gombatestet a megadott tektonra, a megadott névvel.

8.1.44 CommandRouterImpl

- **Felelősség**
  A parancsokat a megfelelő kezelőnek továbbadja.

- **Interfészek**

CommandRouter

- **Attribútumok**
  - **-commandRepository: Map<String, CommandHandler>**
    Egy tábla, ami leírja, hogy egy adott parancsot melyik kezelő fogadja be.
  - **-commandFactory: CommandFactory**
    A parancsokat példányosító objektum.

- **Metódusok**
  - **+CommandRouterImpl(factory: CommandFactory)**
    Konstruktor, a létrehozáshoz szükséges megadni a parancsokat példányosító objektumot.
  - **+routeCommand(command: InputCommand)**
    A kapott parancsot továbbítja a megfelelő kezelőnek.
    Pszeudókód:

```
routeCommand(command: InputCommand)
      routedCommand: Command := commandFactory.create(command.name)
      handler: CommandHandler := commandRepository[command.name]
      handler.handleCommand(routedCommand)
end
```

  - **+addCommand(commandName: String, commandHandler: CommandHandler)**
    A táblában egy új bejegyzést ír be, ami azt írja le, hogy a parancsot melyik

8.1.45 CommandReaderImpl

- **Felelősség**

A felhasználó által megadott parancsokat olvassa be, majd továbbítja az értelmezőnek.

- **Interfészek**

CommandReader, CommandHandler

- **Attribútumok**
  - **- commandRouter: CommandRouter**
    A parancsokat továbbküldő osztály.
  - **- inputBuffer: Queue<String>**
    Egy input puffer, amiből az olvasó elsődlegesen kiolvas.

- **Metódusok**
  - **+CommandReaderImpl(commandRouter: CommandRouter)**
    Konstruktor, meg kell adni az objektumot aminek továbbküldi a parancsot.
  - **+handleCommand(command: Command)**
    Fogadhat parancsokat is vissza, pl: a run parancs, ami egy fájlal feltölti a puffert.
  - **+ getNextCommand()**
    Értelmezi a következő parancsot a pufferből, ha a puffer üres, akkor a játékostól kér új parancsot.
    Pszeudókód:

```
getNextCommand()
readCommand: String
	if inputbuffer is not empty then begin
		readCommand := inputBuffer.poll()
	end
	else then begin
		readCommand := input()
	end
	splitCommand: String[] := readcommand split by spaces
	inputCommand: InputCommand := new(splitCommand[0],splitCommand[1:n])
	commandRouter.routeCommand(inputCommand)
end
```

  - **+ bufferCommand(input: String)**
    Egy parancsot berak a pufferbe.

8.1.46 TracablePrinterImpl

- **Felelősség**

Az ellenőrizhetőség érdekében, úgy írja ki, hogy visszaolvasható legyen a kódban.

- **Interfészek**

CommandHandler, TracablePrinter

- **Attribútumok**
  - **-printHistory: List<String>**
    A kiírt szövegeket tároló lista.

- **Metódusok**
  - **+clearHistory()**
    Kitölri a visszaolvasható
  - **+readHistory(): Iterable<String>**
    Visszaadja a jelenleg eltárolt régi kiírásokat.
  - **+print(output: String)**
    Kiírja a szöveget egy új sorba, és eltárolja.
  - **+printLine(output: String)**
    Kiírja a szöveget egy új sorba, és eltárolja.
  - **+handleCommand(command: Command)**
    Kezeli a kiírással kapcsolatos parancsokat (pl: STATE)

8.1.47 ObjectRegistry

- **Felelősség**

Számon tartja a névvel ellátott objektumokat

- **Attribútumok**
  - **-registeredObjects: Map<String, Object>**
    Az objektumok szövegesen hivatkozható nevei és a velük asszociált objektumok.

- **Metódusok**
  - **-ObjectRegistry()**
    A konstruktor privát, mert nem szabad példányosítani.
  - **+registerObject(name: String, registeredObject: Object)**
    Beírja az objektumot és a nevét a tárba.
  - **+clearRegistry()**
    Kitörli az összes objektumot a tárból.
  - **+removeFromRegistry(name: String)**
    Töröl egy objektumot a tárból.
  - **+getObject(name: String) : Object**
    Név alapján visszatér a keresett objektummal.

8.2 A tesztek részletes tervei, leírásuk a teszt nyelvén

[A tesztek részletes tervei alatt meg kell adni azokat a bemeneti adatsorozatokat, amelyekkel a program működése ellenőrizhető. Minden bemenő adatsorozathoz definiálni kell, hogy az adatsorozat végrehajtásától a program mely részeinek, funkcióinak ellenőrzését várjuk és konkrétan milyen eredményekre számítunk, ezek az eredmények hogyan vethetők össze a bemenetekkel. A tesztek leírásakor az előző dokumentumban (proto koncepciója) megadott szintakszist kell használni.]

- ***Kiadható parancsok:***

CREATE_TECTON Tecton_Type Tecton_Name

Leírás: Új Tecton létrehozása

Opciók: Első paraméter eldönti milyen típusu legyen az új Tecton, A következő paraméter egy lista mely tartalmazza a Tectonokat amelyek szomszédjai lesznek az új Tectonnak

SET_BREAKTIMER tecton number

Leírás: Tecton BreakTimerjének beallítása egy adott értékre

Opciók: A tekton, melynek az időzítőjét beállítjuk, és a szám ami be lesz állitva mint új BreakTimer

END_GAME

Leírás: Játék végének kezelése

Opciók: Nincsenek paraméterek

SET_ENDGAMETIMER number

Leírás: Az EndgameTimer beállítása

Opciók: Egyetlen paramétere egy szám mely az új maradék Round-okat jelöli

END_TURN

Leírás: End turn küldése

Opciók: Nincsenek paraméterek

ADD_PLAYER player_type player_name

Leírás: Játékosok hozzáadása a játékhoz, ilyen sorrendben fognak jönni a játék során. A bemenet legenerálása során az egyes objektumok (amennyiben ez releváns) a legutoljára hozzáadott játékos tulajdonába kerülnek. Játék kezdete után teremtett objektumok annak a játékosnak a tulajdonába kerülnek melynek jelenleg a Turn-je van.

Opciók: Első paraméter a játékos neve, a második paraméter hogy Gombász vagy Rovarász lesz a játékos

START_GAME

Leírás: Játék indítása

Opciók: Nincsenek paraméterek

ADD_NEIGHBOUR **tecton_name tecton_name**

Leírás: Szomszéd hozzáadása a kiválasztott tektonhoz, ez az hozzáadás forditva is megtörténik

Opciók: 1 paraméter (kit - tekton) 2. paraméter (kinek – tekton)

CREATE_MUSHROOMBODY MushroomBody_Name Tecton_Name

Leírás: A gombatest létrejön egy tektonon (céltekton).

(Megjegyzések:

- a gombatestek elnevezési konvenciója: mb[1-től kezdődő, folytatólagos számozás];
- a létrejött gombatest hátralévő spórakilövéseinek száma alapértelmezetten 3;
- a létrejött gombatest spóralistája alapértelmezetten üres, azzal, hogy a játék elindítása utáni első kör elején már termelődik spóra a gombatestben.)

Opciók: Az első paraméter meghatározza a létrehozandó gombatestet, a második a céltektont.

GROW_MUSHROOMBODY MushroomBody_Name Tecton_Name

Leírás: A gombatest létrejön és rákerül egy tektonra (céltekton).

(Megjegyzések:

- a gombatestek elnevezési konvenciója: mb[a CREATE_MUSHROOMBODY paranccsal létrehozott gombatestek számozását folytató, ennek hiányában 1-től kezdődő, folytatólagos számozás];
- a létrejött gombatest hátralévő spórakilövéseinek száma 3;
- a létrejött gombatest spóralistája üres, azzal, hogy a gombatest a létrehozatalára vonatkozó parancs kiadását követően azonnal létrejön és az ezt követő első kör elején már termelődik benne spóra.)

Opciók: Az első paraméter meghatározza a létrejövő gombatestet, a második a céltektont.

PUT_SPORE Spore_Type Spore_Name Tecton_Name

Leírás: Egy adott típusú spóra rákerül egy tektonra (céltekton).

(Megjegyzés:

- a spórák elnevezési konvenciója: [a spóra típusára utaló elnevezés, azaz: StunSpore esetén stuns; PreventCutSpore esetén prevents; SlownessSpore esetén slows; SpeedSpore esetén speeds; SplitSpore esetén splits][1-től kezdődő, folytatólagos számozás – minden spóratípus esetén külön-külön].)

Opciók: Az első paraméter meghatározza a spóra típusát, a második a nevét, a harmadik a céltektont.

EJECT_SPORES MushroomBody_Name Tecton_Name

Leírás: A kiválasztott gombatest valamennyi spórája rákerül egy tektonra (céltekton).

(Megjegyzések:

- a gombatestben körönként automatikusan termelődő spórák elnevezési konvenciója: [gombatest neve]-[a spóra típusára utaló elnevezés, azaz: StunSpore esetén stuns; PreventCutSpore esetén prevents; SlownessSpore esetén slows; SpeedSpore esetén speeds; SplitSpore esetén splits][1-től kezdődő, folytatólagos számozás – minden spóratípus esetén külön-külön];
- a tesztesetekben a gombatestben körönként automatikusan termelődő spóra SpeedSpore típusú.)

Opciók: Az első paraméter meghatározza a gombatestet, a második a céltektont.

DEACTIVATE MushroomBody_Name

Leírás: A kiválasztott gombatest inaktívvá válik.

Opciók: A paraméter meghatározza a gombatestet.

ADD_SPORE Spore_Type Spore_Name MushroomBody_Name

Leírás: A rendszer a kiválasztott gombatesthez meghatározott típusú spórát rendel.

(Megjegyzés:

- a spórák elnevezési konvenciója: [a spóra típusára utaló elnevezés, azaz: StunSpore esetén stuns; PreventCutSpore esetén prevents; SlownessSpore esetén slows; SpeedSpore esetén speeds; SplitSpore esetén splits][a PUT_SPORE paranccsal létrehozott spórák számozását folytató, ennek hiányában 1-től kezdődő, folytatólagos számozás].)

Opciók: Az első paraméter meghatározza a spóra típusát, a második a spóra nevét, harmadik a gombatestet.

SET_REMAININGEJECTS MushroomBody_Name RemainingEjects_Count

Leírás: A rendszer a kiválasztott gombatesthez meghatározott számú, hátralévő spórakilövést rendel.

CREATE_MYCELIUM Mycelium_Type Mycelium_Name

Leírás: Létrehoz egy gombafonalat
Opciók: A gombafonál típusa és neve.

ADD_MYCELIUM_TO_TECTON Mycelium_Name Tecton_Name

Leírás: Hozzáadja a kiválasztott fonalat a kiválasztott tektonhoz
Opciók: A gombafonál, ami rajta lesz a tektonon és a tekton, amin lesz a fonál

GROW_MYCELIUM Mycelium_Type Mycelium_Name Tecton_Name

Leírás: Rá-nő egy gombafonál a kiválasztott tektonra
Opciók: A gombafonál, amit növesztünk és a tekton, amin a gombafonál lesz

CREATE_INSECT tecton_name insect_name

Leírás: A rovar létrejön és rákerül az argumentumként megadott céltektonra, ha ezen van gombafonál

Megjegyzég:

Alapértelmezett értékek:
-maxMoves: 2
-remainingMoves: maxMoves
-sporesEaten: 0
-effectTimer: 0
-state: Normal

Opciók: Argumentumok: A céltekton, ahova létrejönne, az objektum neve

MOVE insect_name tecton_name

Leírás: A rovar átmenne az argumentumként megadott céltektonra
Opciók: Argumentumok: A rovar, amelyik mozogna;
A céltekton, ahova mozogna

EAT insect_name

Leírás: A rovar megeszi az első spórát a tektonján
Opciók: Argumentumok: A rovar, amelyik enne

CUT insect_name

Leírás: A rovar elvágja az első gombafonalat a tektonján
Opciók: Argumentumok: A rovar, amelyik vágna

8.2.1 Új Tecton sikeres legyártása

- **Leírás**
  A rendszer sikeresen legyárt egy általa kiválasztott típusú es nevű Tectont. A rendszer egy időben megmondja azt is, hogy ennek az új Tectonnak kik lesznek a szomszédjai.

  (Megjegyzés: SustainingTecton, MultiLayeredTecton és AridTecton esetén a teszt hasonlóképpen működik, mutatis mutandis.)

- **Ellenőrzött funkcionalitás, várható hibahelyek**
  Új tecton létrehozása, szomszédok hozzáadása az új tectonhoz a szomszédokhoz hozzáadni az új tectont, BreakTimer helyes beállítása

- **Bemenet**

```
CREATE_TECTON FertileTecton ft1
SET_BREAKTIMER ft1 5
CREATE_TECTON FertileTecton ft2
SET_BREAKTIMER ft2 5
CREATE_TECTON FertileTecton ft3
SET_BREAKTIMER ft3 5
ADD_NEIGHBOUR ft1 ft2
ADD_NEIGHBOUR ft1 ft3
ADD_NEIGHBOUR ft2 ft3
STATE ft1
STATE ft2
STATE ft3
```

- **Elvárt kimenet**

```
ft1: FertileTecon
        breakTimer int = 5
        neighbours List<Tecton>  = {
                ft2
                ft3
        }
        myceliumCapacity int = 1
        spores Queue<Spore> = {
        }
        mushroomBody MushroomBody = null
        mycelia Queue<Mycelium> = {
        }
        occupants List<Insect> = {
        }

ft2: FertileTecon
        breakTimer int = 5
        neighbours List<Tecton>  = {
                ft1
                ft3
        }
```

myceliumCapacity int = 1
spores Queue<Spore> = {
}
mushroomBody MushroomBody = null
mycelia Queue<Mycelium> = {
}
occupants List<Insect> = {
}

ft3: FertileTecon
breakTimer int = 5
neighbours List<Tecton> = {
ft1
ft2
}
myceliumCapacity int = 1
spores Queue<Spore> = {
}
mushroomBody MushroomBody = null
mycelia Queue<Mycelium> = {
}
occupants List<Insect> = {
}

8.2.2 Tectontörés

- **Leírás**
  A BreakTimert beállítjuk 0-ra egy adott Tectonon, majd megvizsgáljuk, hogy az új kör után lett-e neki új szomszédja és hogy a gombatesten kívül minden más megsemmisült róla.

  (Megjegyzés: SustainingTecton, MultiLayeredTecton és AridTecton esetén a teszt hasonlóképpen működik, mutatis mutandis.)

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Az adott tectonról tényleg minden letörlődik-e (gombatesten kívül), hozzáadódik-e az új tecton az adott tecton szomszédsági listájába

- **Bemenet**

CREATE_TECTON FertileTecton ft1
SET_BREAKTIMER ft1 1
PUT_SPORE SpeedSpore speeds1 ft1
ADD_PLAYER Mycologist mycologist1
CREATE_MUSHROOMBODY mb1 ft1
CREATE_MYCELIUM Mycelium m1
ADD_MYCELIUM_TO_TECTON m1 ft1
ADD_PLAYER Entomologist entomologist1
CREATE_INSECT ft1 i1
START_GAME
ENDTURN
ENDTURN
STATE ft1
STATE ft1-1

- **Elvárt kimenet**

ft1: FertileTecon
    breakTimer int = 2
    neighbours List<Tecton> = {
        ft1-1
    }
    myceliumCapacity int = 1
    spores Queue<Spore> = {
    }
    mushroomBody MushroomBody = mb1
    mycelia Queue<Mycelium> = {
    }
    occupants List<Insect> = {
        i1
    }

ft1-1: FertileTecon
    breakTimer int = 2

```
neighbours List<Tecton>  = {
        ft1
}
myceliumCapacity int = 1
spores Queue<Spore> = {
}
mushroomBody MushroomBody = null
mycelia Queue<Mycelium> = {
}
occupants List<Insect> = {
}
```

8.2.3 Játék végének kezelése

- **Leírás**

Miután lejárt az utolsó Round is, meghatározni mind a
Gombászok közül egy nyertest, mind a Rovarászok
közül egy nyertest majd kiírni a pontszámukat.
Erre felhasználom a Rovar által elvágott gombafonál elsorvadása és az elfogyasztott spóra rovarra gyakorolt hatása tesztet

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Megnézni, hogy jól számolódik-e ki a pontszám és hogy
helyesen íródnak ki a végeredmények

- **Bemenet**

SET_ENDGAMETIMER 5
CREATE_TECTON FertileTecton ft1
SET_BREAKTIMER ft1 5
CREATE_TECTON FertileTecton ft2
SET_BREAKTIMER ft2 5
CREATE_TECTON FertileTecton ft3
SET_BREAKTIMER ft3 5
CREATE_TECTON FertileTecton ft4
SET_BREAKTIMER ft4 5
CREATE_TECTON FertileTecton ft5
SET_BREAKTIMER ft5 5
CREATE_TECTON FertileTecton ft6
SET_BREAKTIMER ft6 5
PUT_SPORE StunSpore stuns1 ft6
ADD_NEIGHBOUR ft1 ft2
ADD_NEIGHBOUR ft1 ft3
ADD_NEIGHBOUR ft1 ft6
ADD_NEIGHBOUR ft2 ft3
ADD_NEIGHBOUR ft2 ft6
ADD_NEIGHBOUR ft3 ft4
ADD_NEIGHBOUR ft4 ft5
ADD_PLAYER Mycologist mycologist1
CREATE_MUSHROOMBODY mb1 ft1
CREATE_MYCELIUM Mycelium m1
ADD_MYCELIUM_TO_TECTON m1 ft1
CREATE_MYCELIUM Mycelium m2
ADD_MYCELIUM_TO_TECTON m2 ft2
CREATE_MYCELIUM Mycelium m3
ADD_MYCELIUM_TO_TECTON m3 ft4
CREATE_MYCELIUM Mycelium m4

ADD_MYCELIUM_TO_TECTON m4 ft5
CREATE_MYCELIUM Mycelium m5
ADD_MYCELIUM_TO_TECTON m5 ft6
CREATE_MYCELIUM CarnivorousMycelium cm1
ADD_MYCELIUM_TO_TECTON cm1 ft3
ADD_PLAYER Mycologist mycologist2
ADD_PLAYER Entomologist entomologist1
CREATE_INSECT ft5 i1
ADD_PLAYER Entomologist entomogolist2
START_GAME
ENDTURN
ENDTURN
MOVE i1 ft4
MOVE i1 ft3
CUT i1
ENDTURN
ENDTURN
ENDTURN
ENDTURN
MOVE i1 ft2
MOVE i1 ft1
ENDTURN
ENDTURN
EJECT_SPORES mb1 ft2
ENDTURN
ENDTURN
MOVE i1 ft6
EAT i1
ENDTURN
ENDTURN
ENDTURN
ENDTURN
ENDTURN
ENDTURN

- **Elvárt kimenet**

Kiirodik ez a szöveg ilyen formátumban, és ezt a szöveget ellenőrizzük hogy az elvárt-e:
WINNERS:
MYCOLOGIST:
mycologist1
ENTOMOLOGIST:
entomologist1

8.2.4 Gombatest sikeres növesztése FertileTectonra (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton) gombafonál által

- **Leírás**

  Gombafonál sikeresen növeszt gombatestet olyan FertileTectonra, amely nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton.

  (Megjegyzés: SustainingTecton, MultiLayeredTecton és AridTecton esetén a teszt hasonlóképpen működik, mutatis mutandis.)

- **Ellenőrzött funkcionalitás, várható hibahelyek**
  - gombatest sikeres növesztése az ehhez szükséges feltételekkel: a FertileTectonon mint céltektonon van legalább 3 db spóra és nincs gombatest
  - a létrejött gombatest spóráinak meghatározása
  - a létrejött gombatest megmaradt spórakilövései számának beállítása

- **Bemenet**

  CREATE_TECTON FertileTecton ft1
  SET_BREAKTIMER ft1 5
  PUT_SPORE SpeedSpore speeds1 ft1
  PUT_SPORE SpeedSpore speeds2 ft1
  PUT_SPORE SpeedSpore speeds3 ft1
  ADD_PLAYER Mycologist mycologist1
  CREATE_MYCELIUM Mycelium m1
  ADD_MYCELIUM_TO_TECTON m1 ft1
  START_GAME
  GROW_MUSHROOMBODY mb1 ft1
  ENDTURN
  STATE ft1
  STATE m1
  STATE mb1

- **Elvárt kimenet**

ft1: FertileTecton
breakTimer int = 3
neighbours List<Tecton> = {
}
myceliumCapacity int = 1
spores Queue<Spore> = {
speeds1
speeds2
speeds3
}
mushroomBody MushroomBody = mb1
mycelia Queue<Mycelium> = {
m1
}
occupants List<Insect> = {
}

m1: Mycelium
growing boolean = false
location Tecton = ft1
growTimer int = 0
deathTimer int = -1

mb1: MushroomBody
remainingEjects int = 3
location Tecton = ft1
mushroomSpores List<Spore> = {
mb1-speeds1
}

8.2.5 Gombatest spórahiány miatti sikertelen növesztése FertileTectonra (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton) gombafonál által

- **Leírás**

  Gombafonál sikertelenül kísérel meg gombatestet létrehozni olyan FertileTectonra, amely nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton, és amelyen nem található elegendő spóra.

  (Megjegyzés: SustainingTecton, MultiLayeredTecton és AridTecton esetén a teszt hasonlóképpen működik, mutatis mutandis.)

- **Ellenőrzött funkcionalitás, várható hibahelyek**
  - gombatest sikertelen növesztése a következő feltételek fennállása mellett: a FertileTectonon mint céltektonon nincs legalább 3 db spóra és nincs gombatest
  - az objektumok állapotában nem következik be változás

- **Bemenet**

  CREATE_TECTON FertileTecton ft1
  SET_BREAKTIMER ft1 5
  CREATE_MYCELIUM Mycelium m1
  PUT_SPORE SpeedSpore speeds1 ft1
  PUT_SPORE SpeedSpore speeds2 ft1
  ADD_PLAYER Mycologist mycologist1
  ADD_MYCELIUM_TO_TECTON m1 ft1
  START_GAME
  GROW_MUSHROOMBODY mb1 ft1
  ENDTURN
  STATE ft1
  STATE m1

- **Elvárt kimenet**

  ft1: FertileTecton
  breakTimer int = 3
  neighbours List<Tecton> = {
  }
  myceliumCapacity int = 1
  spores Queue<Spore> = {
  speeds1
  speeds2
  }
  mushroomBody MushroomBody = null

mycelia Queue<Mycelium> = {
m1
}
occupants List<Insect> = {
}

m1: Mycelium
growing boolean = false
location Tecton = ft1
growTimer int = 0
deathTimer int = -1

8.2.6 Gombatest sikertelen növesztése gombafonál által olyan FertileTectonra (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton), amelyen már van gombatest

- **Leírás**

  Gombatest sikertelen növesztése gombafonál által olyan FertileTectonra (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton), amelyen már van gombatest.

  (Megjegyzés: SustainingTecton, MultiLayeredTecton és AridTecton esetén a teszt hasonlóképpen működik, mutatis mutandis.)

- **Ellenőrzött funkcionalitás, várható hibahelyek**
  - gombatest sikertelen növesztése a következő feltételek fennállása mellett: a FertileTectonon mint céltektonon van gombatest és legalább 3 db spóra
  - az objektumok állapotában nem következik be változás

- **Bemenet**

  CREATE_TECTON FertileTecton ft1
  SET_BREAKTIMER ft1 5
  PUT_SPORE SpeedSpore speeds1 ft1
  PUT_SPORE SpeedSpore speeds2 ft1
  PUT_SPORE SpeedSpore speeds3 ft1
  ADD_PLAYER Mycologist mycologist1
  CREATE_MYCELIUM Mycelium m1
  ADD_MYCELIUM_TO_TECTON m1 ft1
  CREATE_MUSHROOMBODY mb1 ft1
  START_GAME
  GROW_MUSHROOMBODY mb2 ft1
  ENDTURN
  STATE ft1
  STATE m1
  STATE mb1

- **Elvárt kimenet**

  ft1: FertileTecton
  breakTimer int = 3
  neighbours List<Tecton> = {
  }
  myceliumCapacity int = 1
  spores Queue<Spore> = {
  speeds1
  speeds2

```
            speeds3
        }
        mushroomBody MushroomBody = mb1
        mycelia Queue<Mycelium> = {
            m1
        }
        occupants List<Insect> = {
        }

m1: Mycelium
        growing boolean = false
        location Tecton = ft1
        growTimer int = 0
        deathTimer int = -1

mb1: MushroomBody
        remainingEjects int = 3
        location Tecton = ft1
        mushroomSpores List<Spore> = {
            mb1-speeds1
            mb1-speeds2
        }
```

8.2.7 Gombatest sikertelen növesztése gombafonál által SemiFertileTectonra

- **Leírás**

  Gombafonál sikertelenül kísérel meg gombatestet létrehozni SemiFertileTectonra, amelyen van legalább 3db spóra (és nincs, mert nem is lehet rajta gombatest).

- **Ellenőrzött funkcionalitás, várható hibahelyek**
  - gombatest sikertelen növesztése a következő feltételek fennállása mellett: a SemiFertileTectonon mint céltektonon van legalább 3 db spóra (és nincs, mert nem is lehet rajta gombatest).
  - az objektumok állapotában nem következik be változás

- **Bemenet**

  CREATE_TECTON SemiFertileTecton sft1
  SET_BREAKTIMER sft1 5
  PUT_SPORE SpeedSpore speeds1 sft1
  PUT_SPORE SpeedSpore speeds2 sft1
  PUT_SPORE SpeedSpore speeds3 sft1
  ADD_PLAYER Mycologist mycologist1
  CREATE_MYCELIUM Mycelium m1
  ADD_MYCELIUM_TO_TECTON m1 sft1
  START_GAME
  GROW_MUSHROOMBODY mb1 sft1
  ENDTURN
  STATE sft1
  STATE m1

- **Elvárt kimenet**

  sft1: SemiFertileTecton
  breakTimer int = 3
  neighbours List<Tecton> = {
  }
  myceliumCapacity int = 1
  spores Queue<Spore> = {
  speeds1
  speeds2
  speeds3
  }
  mushroomBody MushroomBody = null
  mycelia Queue<Mycelium> = {
  m1
  }

occupants List<Insect> = {
}

m1: Mycelium
growing boolean = false
location Tecton = sft1
growTimer int = 0
deathTimer int = -1

8.2.8 Gombatest sikeres spórakilövése a gombatest elhelyezkedése szerinti tektonnal szomszédos FertileTectonra (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton)

- **Leírás**

  Gombatest sikeresen kilövi a spóráit a gombatest elhelyezkedése szerinti tektonnal szomszédos FertileTectonra (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton).

  (Megjegyzés: SustainingTecton, MultiLayeredTecton, AridTecton és SemiFertileTecton mint céltektonok esetén a teszt hasonlóképpen működik, mutatis mutandis.)

- **Ellenőrzött funkcionalitás, várható hibahelyek**
  - gombatest sikeres spórakilövése a következő feltételek fennállása mellett: gombatest az utolsó, azaz a harmadik kilövése előtt csak valamely szomszédjára lőhet spórát
  - a gombatest spóráinak száma a kilövés következtében 0-ra csökken
  - a kilőtt spórákat a továbbiakban a céltekton tartja nyilván

- **Bemenet**

  CREATE_TECTON FertileTecton ft1
  SET_BREAKTIMER ft1 5
  CREATE_TECTON FertileTecton ft2
  SET_BREAKTIMER ft2 5
  ADD_NEIGHBOUR ft1 ft2
  ADD_PLAYER Mycologist mycologist1
  CREATE_MUSHROOMBODY mb1 ft1
  ADD_SPORE SpeedSpore speeds1 mb1
  ADD_SPORE SpeedSpore speeds2 mb1
  ADD_SPORE SpeedSpore speeds3 mb1
  START_GAME
  EJECT_SPORES mb1 ft2
  ENDTURN
  STATE ft1
  STATE ft2
  STATE mb1

- **Elvárt kimenet**

  ft1: FertileTecton
  breakTimer int = 3
  neighbours List<Tecton> = {
  ft2

```
        }
        myceliumCapacity int = 1
        spores Queue<Spore> = {
        }
        mushroomBody MushroomBody = mb1
        mycelia Queue<Mycelium> = {
        }
        occupants List<Insect> = {
        }

    ft2: FertileTecton
        breakTimer int = 3
        neighbours List<Tecton> = {
            ft1
        }
        myceliumCapacity int = 1
        spores Queue<Spore> = {
            speeds1
            speeds2
            speeds3
            mb1-speeds1
        }
        mushroomBody MushroomBody = null
        mycelia Queue<Mycelium> = {
        }
        occupants List<Insect> = {
        }

    mb1: MushroomBody
        remainingEjects int = 2
        location Tecton = ft1
        mushroomSpores List<Spore> = {
            mb1-speeds2
        }
```

8.2.9 Gombatest sikeres spórakilövése olyan FertileTectonra (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton), amely a gombatest elhelyezkedése szerinti tektonnal szomszédos tekton szomszédja

- **Leírás**

  Gombatest sikeres, összesen a harmadik (utolsó) spórakilövése olyan FertileTectonra (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton), amely a gombatest elhelyezkedése szerinti tektonnal szomszédos tekton szomszédja.

  (Megjegyzés: SustainingTecton, MultiLayeredTecton, AridTecton és SemiFertileTecton mint céltektonok esetén a teszt hasonlóképpen működik, mutatis mutandis.)

- **Ellenőrzött funkcionalitás, várható hibahelyek**
  - gombatest sikeres spórakilövése a következő feltételek fennállása mellett: gombatest az utolsó, azaz a harmadik kilövése előtt csak valamely szomszédjára lőhet spórát. Fejlettnek az utolsó kilövése során minősül, amely abban nyilvánul meg, hogy ekkor a szomszédja szomszédjára is tud lőni
  - a gombatest a harmadik kilövését követően inaktívvá válik
  - a kilőtt spórákat a továbbiakban a céltekton tartja nyilván

- **Bemenet**

```
CREATE_TECTON FertileTecton ft1
SET_BREAKTIMER ft1 5
CREATE_TECTON FertileTecton ft2
SET_BREAKTIMER ft2 5
CREATE_TECTON FertileTecton ft3
SET_BREAKTIMER ft3 5
ADD_NEIGHBOUR ft1 ft2
ADD_NEIGHBOUR ft2 ft3
ADD_PLAYER Mycologist mycologist1
CREATE_MUSHROOMBODY mb1 ft1
ADD_SPORE SpeedSpore speeds1 mb1
ADD_SPORE SpeedSpore speeds2 mb1
ADD_SPORE SpeedSpore speeds3 mb1
SET_REMAININGEJECTS mb1 1
START_GAME
EJECT_SPORES mb1 ft3
ENDTURN
STATE ft1
STATE ft2
STATE ft3
```

STATE mb1

- **Elvárt kimenet**

ft1: FertileTecton
breakTimer int = 3
neighbours List<Tecton> = {
ft2
}
myceliumCapacity int = 1
spores Queue<Spore> = {
}
mushroomBody MushroomBody = mb1
mycelia Queue<Mycelium> = {
}
occupants List<Insect> = {
}

ft2: FertileTecton
breakTimer int = 3
neighbours List<Tecton> = {
ft1
ft3
}
myceliumCapacity int = 1
spores Queue<Spore> = {
}
mushroomBody MushroomBody = null
mycelia Queue<Mycelium> = {
}
occupants List<Insect> = {
}

ft3: FertileTecton
breakTimer int = 3
neighbours List<Tecton> = {
ft2
}
myceliumCapacity int = 1
spores Queue<Spore> = {
speeds1
speeds2
speeds3
mb1-speeds1
}

mushroomBody MushroomBody = null
mycelia Queue<Mycelium> = {
}
occupants List<Insect> = {
}

mb1: MushroomBody
remainingEjects int = 0
location Tecton = ft1
mushroomSpores List<Spore> = {
}

8.2.10 Gombatest spórahiány miatti sikertelen spórakilövése a gombatest elhelyezkedése szerinti tektonnal szomszédos FertileTectonra (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton)

- **Leírás**

  Gombatest egy-egy spórakilövést kísérel meg az elhelyezkedése szerinti tektonnal szomszédos két FertileTectonra (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton). Sorrendben a második kísérlet sikertelen, tekintettel arra, hogy a gombatestnek ekkor már nincsen kilőhető spórája.

  (Megjegyzés: SustainingTecton, MultiLayeredTecton, AridTecton és SemiFertileTecton mint céltektonok esetén a teszt hasonlóképpen működik, mutatis mutandis.)

- **Ellenőrzött funkcionalitás, várható hibahelyek**
  - gombatest sikertelen spórakilövése a következő feltételek fennállása mellett: gombatest az utolsó, azaz a harmadik kilövése előtt csak valamely szomszédjára lőheti ki meglévő spóráit

- **Bemenet**

  CREATE_TECTON FertileTecton ft1
  SET_BREAKTIMER ft1 5
  CREATE_TECTON FertileTecton ft2
  SET_BREAKTIMER ft2 5
  CREATE_TECTON FertileTecton ft3
  SET_BREAKTIMER ft3 5
  ADD_NEIGHBOUR ft1 ft2
  ADD_NEIGHBOUR ft1 ft3
  ADD_PLAYER Mycologist mycologist1
  CREATE_MUSHROOMBODY mb1 ft1
  START_GAME
  EJECT_SPORES mb1 ft2
  EJECT_SPORES mb1 ft3
  ENDTURN
  STATE ft1
  STATE ft2
  STATE ft3
  STATE mb1

- **Elvárt kimenet**

  ft1: FertileTecton
  breakTimer int = 3
  neighbours List<Tecton> = {

```
                ft2
                ft3
        }
        myceliumCapacity int = 1
        spores Queue<Spore> = {
        }
        mushroomBody MushroomBody = mb1
        mycelia Queue<Mycelium> = {
        }
        occupants List<Insect> = {
        }
ft2: FertileTecton
        breakTimer int = 3
        neighbours List<Tecton> = {
                ft1
        }
        myceliumCapacity int = 1
        spores Queue<Spore> = {
                mb1-speeds1
        }
        mushroomBody MushroomBody = null
        mycelia Queue<Mycelium> = {
        }
        occupants List<Insect> = {
        }

ft3: FertileTecton
        breakTimer int = 3
        neighbours List<Tecton> = {
                ft1
        }
        myceliumCapacity int = 1
        spores Queue<Spore> = {
        }
        mushroomBody MushroomBody = null
        mycelia Queue<Mycelium> = {
        }
        occupants List<Insect> = {
        }

mb1: MushroomBody
        remainingEjects int = 2
        location Tecton = ft1
        mushroomSpores List<Spore> = {
```

```
                mb1-speeds2
        }
```

8.2.11 Gombatest sikertelen spórakilövése olyan FertileTectonra (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton), amely a gombatest elhelyezkedése szerinti tektonnal szomszédos tekton szomszédja

- **Leírás**

  Gombatest sikertelenül kísérel meg spórakilövést olyan FertileTectonra (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton), amely a gombatest elhelyezkedése szerinti tektonnal szomszédos tekton szomszédja, mert nem ez a gombatest harmadik (összességében az utolsó) spórakilövése.

  (Megjegyzés: SustainingTecton, MultiLayeredTecton, AridTecton és SemiFertileTecton mint céltektonok esetén a teszt hasonlóképpen működik, mutatis mutandis.)

- **Ellenőrzött funkcionalitás, várható hibahelyek**
  - gombatest sikertelen spórakilövése a következő feltételek fennállása mellett: gombatest az utolsó, azaz a harmadik kilövése előtt csak valamely szomszédjára lőhet spórát. Fejlettnek az utolsó kilövése során minősül, amely abban nyilvánul meg, hogy ekkor a szomszédja szomszédjára is tud lőni
  - az objektumok állapotában nem következik be változás

- **Bemenet**

  CREATE_TECTON FertileTecton ft1
  SET_BREAKTIMER ft1 5
  CREATE_TECTON FertileTecton ft2
  SET_BREAKTIMER ft2 5
  CREATE_TECTON FertileTecton ft3
  SET_BREAKTIMER ft3 5
  ADD_NEIGHBOUR ft1 ft2
  ADD_NEIGHBOUR ft2 ft3
  ADD_PLAYER Mycologist mycologist1
  CREATE_MUSHROOMBODY mb1 ft1
  ADD_SPORE SpeedSpore speeds1 mb1
  ADD_SPORE SpeedSpore speeds2 mb1
  ADD_SPORE SpeedSpore speeds3 mb1
  START_GAME
  EJECT_SPORES mb1 ft3
  ENDTURN
  STATE ft1
  STATE ft2
  STATE ft3
  STATE mb1

- **Elvárt kimenet**

ft1: FertileTecton
breakTimer int = 3
neighbours List<Tecton> = {
ft2
}
myceliumCapacity int = 1
spores Queue<Spore> = {
}
mushroomBody MushroomBody = mb1
mycelia Queue<Mycelium> = {
}
occupants List<Insect> = {
}

ft2: FertileTecton
breakTimer int = 3
neighbours List<Tecton> = {
ft1
ft3
}
myceliumCapacity int = 1
spores Queue<Spore> = {
}
mushroomBody MushroomBody = null
mycelia Queue<Mycelium> = {
}
occupants List<Insect> = {
}

ft3: FertileTecton
breakTimer int = 3
neighbours List<Tecton> = {
ft2
}
myceliumCapacity int = 1
spores Queue<Spore> = {
}
mushroomBody MushroomBody = null
mycelia Queue<Mycelium> = {
}
occupants List<Insect> = {
}

mb1: MushroomBody
remainingEjects int = 3
location Tecton = ft1
mushroomSpores List<Spore> = {
speeds1
speeds2
speeds3
mb1-speeds1
mb1-speeds2
}

8.2.12 Gombatest sikertelen spórakilövése olyan FertileTectonra (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton), amely a gombatest elhelyezkedése szerinti tekton harmadik szomszédja

- **Leírás**

  Gombatest sikertelenül kísérel meg spórakilövést olyan FertileTectonra (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton, céltekton), amely a gombatest elhelyezkedése szerinti tekton harmadik szomszédja. [Azaz létezik A, B, C és D FertileTecton, amelyek a következőképpen szomszédosak (a szomszédosságot a – jelöli): A – B – C – D. (A tektonok egyéb módon nem szomszédosak egymással.) A gombatest A FertileTectonon található. A tekton harmadik szomszédja D tekton.]

  (Megjegyzés: SustainingTecton, MultiLayeredTecton, AridTecton és SemiFertileTecton mint céltektonok esetén a teszt hasonlóképpen működik, mutatis mutandis.)

- **Ellenőrzött funkcionalitás, várható hibahelyek**
  - gombatest az utolsó, azaz a harmadik kilövése előtt csak valamely szomszédjára lőhet spórát. Fejlettnek az utolsó kilövése során minősül, amely abban nyilvánul meg, hogy ekkor a szomszédja szomszédjára is tud lőni. Érettségétől függetlenül azonban nem képes spórakilövésre az elhelyezkedése szerinti tekton harmadik szomszédjára
  - az érettnek minősülő gombatestnek az elhelyezkedése szerinti tekton harmadik szomszédja tekintetében megkísérelt spórakilövését teszteljük
  - az objektumok állapotában nem következik be változás

- **Bemenet**

```
CREATE_TECTON FertileTecton ft1
SET_BREAKTIMER ft1 5
CREATE_TECTON FertileTecton ft2
SET_BREAKTIMER ft2 5
CREATE_TECTON FertileTecton ft3
SET_BREAKTIMER ft3 5
CREATE_TECTON FertileTecton ft4
SET_BREAKTIMER ft4 5
ADD_NEIGHBOUR ft1 ft2
ADD_NEIGHBOUR ft2 ft3
ADD_NEIGHBOUR ft3 ft4
ADD_PLAYER Mycologist mycologist1
CREATE_MUSHROOMBODY mb1 ft1
ADD_SPORE SpeedSpore speeds1 mb1
ADD_SPORE SpeedSpore speeds2 mb1
```

ADD_SPORE SpeedSpore speeds3 mb1
SET_REMAININGEJECTS mb1 1
START_GAME
EJECT_SPORES mb1 ft4
ENDTURN
STATE ft1
STATE ft2
STATE ft3
STATE ft4
STATE mb1

- **Elvárt kimenet**

ft1: FertileTecton
breakTimer int = 3
neighbours List<Tecton> = {
ft2
}
myceliumCapacity int = 1
spores Queue<Spore> = {
}
mushroomBody MushroomBody = mb1
mycelia Queue<Mycelium> = {
}
occupants List<Insect> = {
}

ft2: FertileTecton
breakTimer int = 3
neighbours List<Tecton> = {
ft1
ft3
}
myceliumCapacity int = 1
spores Queue<Spore> = {
}
mushroomBody MushroomBody = null
mycelia Queue<Mycelium> = {
}
occupants List<Insect> = {
}

ft3: FertileTecton
breakTimer int = 3
neighbours List<Tecton> = {

ft2
ft4
}
myceliumCapacity int = 1
spores Queue<Spore> = {
}
mushroomBody MushroomBody = null
mycelia Queue<Mycelium> = {
}
occupants List<Insect> = {
}

ft4: FertileTecton
breakTimer int = 3
neighbours List<Tecton> = {
ft3
}
myceliumCapacity int = 1
spores Queue<Spore> = {
}
mushroomBody MushroomBody = null
mycelia Queue<Mycelium> = {
}
occupants List<Insect> = {
}

mb1: MushroomBody
remainingEjects int = 1
location Tecton = ft1
mushroomSpores List<Spore> = {
speeds1
speeds2
speeds3
mb1-speeds1
mb1-speeds2
}

8.2.13 Inaktív gombatest sikertelen spórakilövése a gombatest elhelyezkedése szerinti tektonnal szomszédos FertileTectonra (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton)

- **Leírás**

  Inaktív gombatest sikertelenül próbál spórakilövést végrehajtani a gombatest elhelyezkedése szerinti tektonnal szomszédos FertileTectonra (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton; céltekton).

  (Megjegyzés: SustainingTecton, MultiLayeredTecton, AridTecton és SemiFertileTecton mint céltektonok esetén a teszt hasonlóképpen működik, mutatis mutandis.)

- **Ellenőrzött funkcionalitás, várható hibahelyek**
  - Inaktív gombatest semmilyen cselekvésre, így spórakilövésre sem képes (ebbe az állapotba a harmadik spórakilövése után kerül a gombatest, és ekkor már nincsen spórája)
  - az objektumok állapotában nem következik be változás

- **Bemenet**

  CREATE_TECTON FertileTecton ft1
  SET_BREAKTIMER ft1 5
  CREATE_TECTON FertileTecton ft2
  SET_BREAKTIMER ft2 5
  ADD_NEIGHBOUR ft1 ft2
  ADD_PLAYER Mycologist mycologist1
  CREATE_MUSHROOMBODY mb1 ft1
  DEACTIVATE mb1
  START_GAME
  EJECT_SPORES mb1 ft2
  ENDTURN
  STATE ft1
  STATE ft2
  STATE mb1

- **Elvárt kimenet**

  ft1: FertileTecton
  breakTimer int = 3
  neighbours List<Tecton> = {
  ft2
  }
  myceliumCapacity int = 1

spores Queue<Spore> = {
}
mushroomBody MushroomBody = mb1
mycelia Queue<Mycelium> = {
}
occupants List<Insect> = {
}

ft2: FertileTecton
breakTimer int = 3
neighbours List<Tecton> = {
ft1
}
myceliumCapacity int = 1
spores Queue<Spore> = {
}
mushroomBody MushroomBody = null
mycelia Queue<Mycelium> = {
}
occupants List<Insect> = {
}

mb1: MushroomBody
remainingEjects int = 0
location Tecton = ft1
mushroomSpores List<Spore> = {
}

8.2.14 StunSpore sikeres elhelyezése FertileTectonon (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton)

- **Leírás**

  Egy StunSpore sikeresen elhelyezésre kerül egy FertileTectonra (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton; céltekton).

  (Megjegyzés: SustainingTecton, MultiLayeredTecton, AridTecton és SemiFertileTecton mint céltektonok, valamint SplitSpore, PreventCutSpore, SpeedSpore és SlownessSpore esetén a teszt hasonlóképpen működik, mutatis mutandis.)

- **Ellenőrzött funkcionalitás, várható hibahelyek**
  - a spóra elhelyezése következtében a céltekton az elhelyezett spórát nyilvántartásba veszi
  - az objektumok állapota egyebekben nem változik

- **Bemenet**

  CREATE_TECTON FertileTecton ft1
  SET_BREAKTIMER ft1 5
  PUT_SPORE StunSpore stuns1 ft1
  STATE ft1

- **Elvárt kimenet**

  ft1: FertileTecton
  breakTimer int = 5
  neighbours List<Tecton> = {
  }
  myceliumCapacity int = 1
  spores Queue<Spore> = {
  stuns1
  }
  mushroomBody MushroomBody = null
  mycelia Queue<Mycelium> = {
  }
  occupants List<Insect> = {
  }

8.2.15 Gombatest inaktívvá válása a harmadik spórakilövését követően

- **Leírás**

  Gombatest három spórakilövését követően inaktívvá válik.

  Egy FertileTectonon (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton; céltekton) található gombatest mindhárom alkalommal egy darab SpeedSpore típusú spórát lő ki (funkcionálisan mindig az összes spóráját kilövi; körönként egy-egy SpeedSpore-t termel) különböző FertileTectonokra (fenti jellegű; céltekton). A kilövések során a céltekton mindig szomszédos. A rovar tektonról-tektonra mozogva bejárja a pályát és közben nem eszik spórát, nem rág el fonalat.

  (Megjegyzés: SustainingTecton, MultiLayeredTecton és AridTecton mint a gombatest elhelyezkedése szerinti tekton és céltekton, CarnivorousMycelium, valamint SplitSpore, StunSpore, PreventCutSpore és SlownessSpore esetén a teszt hasonlóképpen működik, mutatis mutandis.)

- **Ellenőrzött funkcionalitás, várható hibahelyek**
  - a gombatest a harmadik spórakilövését követően inaktívvá válik
  - a többi objektum állapotában bekövetkezett változások vizsgálata

- **Bemenet**

```
CREATE_TECTON FertileTecton ft1
SET_BREAKTIMER ft1 6
CREATE_TECTON FertileTecton ft2
SET_BREAKTIMER ft2 6
CREATE_TECTON FertileTecton ft3
SET_BREAKTIMER ft3 6
CREATE_TECTON FertileTecton ft4
SET_BREAKTIMER ft4 6
ADD_NEIGHBOUR ft1 ft2
ADD_NEIGHBOUR ft1 ft3
ADD_NEIGHBOUR ft1 ft4
ADD_NEIGHBOUR ft2 ft3
ADD_NEIGHBOUR ft3 ft4
ADD_PLAYER Mycologist mycologist1
CREATE_MUSHROOMBODY mb1 ft1
CREATE_MYCELIUM Mycelium m1
ADD_MYCELIUM_TO_TECTON m1 ft1
CREATE_MYCELIUM Mycelium m2
ADD_MYCELIUM_TO_TECTON m2 ft2
CREATE_MYCELIUM Mycelium m3
ADD_MYCELIUM_TO_TECTON m3 ft3
```

CREATE_MYCELIUM Mycelium m4
ADD_MYCELIUM_TO_TECTON m4 ft4
ADD_PLAYER Entomologist entomologist1
CREATE_INSECT ft1 i1
START_GAME
EJECT_SPORES mb1 ft2
ENDTURN
MOVE i1 ft2
ENDTURN
EJECT_SPORES mb1 ft3
ENDTURN
MOVE i1 ft3
ENDTURN
EJECT_SPORES mb1 ft4
ENDTURN
MOVE i1 ft4
ENDTURN
STATE ft1
STATE ft2
STATE ft3
STATE ft4
STATE mb1
STATE m1
STATE m2
STATE m3
STATE m4
STATE i1

- **Elvárt kimenet**

ft1: FertileTecton
    breakTimer int = 2
    neighbours List<Tecton> = {
        ft2
        ft3
        ft4
    }
    myceliumCapacity int = 1
    spores Queue<Spore> = {
    }
    mushroomBody MushroomBody = mb1
    mycelia Queue<Mycelium> = {
        m1
    }
    occupants List<Insect> = {

```
        }

    ft2: FertileTecton
        breakTimer int = 2
        neighbours List<Tecton> = {
            ft1
            ft3
        }
        myceliumCapacity int = 1
        spores Queue<Spore> = {
            mb1-speeds1
        }
        mushroomBody MushroomBody = null
        mycelia Queue<Mycelium> = {
            m2
        }
        occupants List<Insect> = {
        }

    ft3: FertileTecton
        breakTimer int = 2
        neighbours List<Tecton> = {
            ft1
            ft2
            ft4
        }
        myceliumCapacity int = 1
        spores Queue<Spore> = {
            mb1-speeds2
        }
        mushroomBody MushroomBody = null
        mycelia Queue<Mycelium> = {
            m3
        }
        occupants List<Insect> = {
        }

    ft4: FertileTecton
        breakTimer int = 2
        neighbours List<Tecton> = {
            ft1
            ft3
        }
        myceliumCapacity int = 1
```

```
        spores Queue<Spore> = {
            mb1-speeds3
        }
        mushroomBody MushroomBody = null
        mycelia Queue<Mycelium> = {
            m4
        }
        occupants List<Insect> = {
            i1
        }

mb1: MushroomBody
        remainingEjects int = 0
        location Tecton = ft1
        mushroomSpores List<Spore> = {
        }

m1: Mycelium
        growing boolean = false
        location Tecton = ft1
        growTimer int = 0
        deathTimer int = -1

m2: Mycelium
        growing boolean = false
        location Tecton = ft2
        growTimer int = 0
        deathTimer int = -1

m3: Mycelium
        growing boolean = false
        location Tecton = ft3
        growTimer int = 0
        deathTimer int = -1

m4: Mycelium
        growing boolean = false
        location Tecton = ft4
        growTimer int = 0
        deathTimer int = -1

i1: Insect
        location Tecton = ft4
        maxMoves int = 2
```

remainingMoves int = 1
sporesEaten int = 0
effectTimer int = 0
state InsectState = NORMAL

8.2.16 Rovar által elvágott gombafonál elsorvadása és az elfogyasztott spóra rovarra gyakorolt hatása

- **Leírás**

  A rovar által elvágott CarnivorousMycelium 3 kör elteltével elpusztul. Az elfogyasztott StunSpore hatására a rovar 1 körön keresztül semmilyen aktivitást nem képes kifejteni.

  Az első körben a rovar ft3-an állva elvágja az ott lévő húsevő gombafonalat (CarnivorousMycelium). A gombatest a harmadik körben kilövi az időközben termelődött spóráit ft2-re. A rovar ugyanebben a körben elfogyasztja az ft6-on található StunSpore-t. Ennek hatása az ötödik kör elejére megszűnik, akkor, amikor az elvágott gombafonál elsorvad és vele együtt a gombatesttel való összeköttetés nélkül maradt gombafonalak is ft4-en és ft5-ön.

- **Ellenőrzött funkcionalitás, várható hibahelyek**
  - a rovar által elvágott gombafonál elsorvadása
  - a rovar által elfogyasztott spóra hatása
  - a többi objektum állapotában bekövetkezett változások vizsgálata

- **Bemenet**

```
CREATE_TECTON FertileTecton ft1
SET_BREAKTIMER ft1 6
CREATE_TECTON FertileTecton ft2
SET_BREAKTIMER ft2 6
CREATE_TECTON FertileTecton ft3
SET_BREAKTIMER ft3 6
CREATE_TECTON FertileTecton ft4
SET_BREAKTIMER ft4 6
CREATE_TECTON FertileTecton ft5
SET_BREAKTIMER ft5 6
PUT_SPORE StunSpore stuns1 ft5
ADD_NEIGHBOUR ft1 ft2
ADD_NEIGHBOUR ft1 ft3
ADD_NEIGHBOUR ft1 ft5
ADD_NEIGHBOUR ft2 ft3
ADD_NEIGHBOUR ft2 ft5
ADD_NEIGHBOUR ft3 ft4
ADD_PLAYER Mycologist mycologist1
CREATE_MUSHROOMBODY mb1 ft1
CREATE_MYCELIUM Mycelium m1
ADD_MYCELIUM_TO_TECTON m1 ft1
CREATE_MYCELIUM Mycelium m2
ADD_MYCELIUM_TO_TECTON m2 ft2
```

CREATE_MYCELIUM Mycelium m3
ADD_MYCELIUM_TO_TECTON m3 ft4
CREATE_MYCELIUM Mycelium m4
ADD_MYCELIUM_TO_TECTON m4 ft5
CREATE_MYCELIUM CarnivorousMycelium cm1
ADD_MYCELIUM_TO_TECTON cm1 ft3
ADD_PLAYER Entomologist entomologist1
CREATE_INSECT ft4 i1
START_GAME
ENDTURN
MOVE i1 ft3
CUT i1
ENDTURN
ENDTURN
MOVE i1 ft2
MOVE i1 ft1
ENDTURN
EJECT_SPORES mb1 ft2
ENDTURN
MOVE i1 ft5
EAT i1
ENDTURN
ENDTURN
ENDTURN
STATE ft1
STATE ft2
STATE ft3
STATE ft4
STATE ft5
STATE mb1
STATE m1
STATE m2
STATE m4
STATE i1

- **Elvárt kimenet**

ft1: FertileTecton
    breakTimer int = 1
    neighbours List<Tecton> = {
        ft2
        ft3
        ft5
    }
    myceliumCapacity int = 1

spores Queue<Spore> = {
}
mushroomBody MushroomBody = mb1
mycelia Queue<Mycelium> = {
m1
}
occupants List<Insect> = {
}

ft2: FertileTecton
breakTimer int = 1
neighbours List<Tecton> = {
ft1
ft3
ft5
}
myceliumCapacity int = 1
spores Queue<Spore> = {
mb1-speeds1
mb1-speeds2
mb1-speeds3
}
mushroomBody MushroomBody = null
mycelia Queue<Mycelium> = {
m2
}
occupants List<Insect> = {
}

ft3: FertileTecton
breakTimer int = 1
neighbours List<Tecton> = {
ft1
ft2
ft4
}
myceliumCapacity int = 1
spores Queue<Spore> = {
}
mushroomBody MushroomBody = null
mycelia Queue<Mycelium> = {
}
occupants List<Insect> = {
}

ft4: FertileTecton
    breakTimer int = 1
    neighbours List<Tecton> = {
        ft3
    }
    myceliumCapacity int = 1
    spores Queue<Spore> = {
    }
    mushroomBody MushroomBody = null
    mycelia Queue<Mycelium> = {
    }
    occupants List<Insect> = {
    }

ft5: FertileTecton
    breakTimer int = 1
    neighbours List<Tecton> = {
        ft1
        ft2
    }
    myceliumCapacity int = 1
    spores Queue<Spore> = {
    }
    mushroomBody MushroomBody = null
    mycelia Queue<Mycelium> = {
        m4
    }
    occupants List<Insect> = {
        i1
    }

mb1: MushroomBody
    remainingEjects int = 2
    location Tecton = ft1
    mushroomSpores List<Spore> = {
        mb1-speeds4
        mb1-speeds5
    }

m1: Mycelium
    growing boolean = false
    location Tecton = ft1
    growTimer int = 0

deathTimer int = -1

m2: Mycelium
growing boolean = false
location Tecton = ft2
growTimer int = 0
deathTimer int = -1

m4: Mycelium
growing boolean = false
location Tecton = ft5
growTimer int = 0
deathTimer int = -1

i1: Insect
location Tecton = ft5
maxMoves int = 2
remainingMoves int = 2
sporesEaten int = 1
effectTimer int = 0
state InsectState = NORMAL

8.2.17 Gombafonál sikeres (lassú) növesztése

- **Leírás**

Gombafonál sikeres (lassú) növesztése gombatestből FertileTectonra (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton)

(Megjegyzés: a növesztés hasonlóan működik, ha a tekton, ahova növesztünk SustainingTectonMultiLayeredTecton, AridTecton SustainingTecton vagy SemiFertileTecton. Akkor is hasonló a teszteset, ha gombafonálból növesztünk.)

- **Ellenőrzött funkcionalitás, várható hibahelyek**

A gombafonál növési folyamatát, az ahhoz szükséges feltételeket és a tektonon történt változást vizsgáljuk.
Ellenőrizzük, hogy a fonál valóban rákerült-e a tektonra.
A kiválasztott FertileTectonon nincs spóra, ezért a gombafonál lassan (2 kör alatt) fog nőni.

- **Bemenet**

```
CREATE_TECTON FertileTecton ft1
SET_BREAKTIMER ft1 5
CREATE_TECTON FertileTecton ft2
SET_BREAKTIMER ft2 5
ADD_NEIGHBOUR ft1 ft2
ADD_PLAYER Mycologist mycologist1
CREATE_MUSHROOMBODY mb1 ft1
START_GAME
GROW_MYCELIUM Mycelium m1 ft2
ENDTURN
STATE ft2
STATE m1
ENDTURN
STATE ft1
STATE ft2
STATE mb1
STATE m1
```

Elvárt kimenet

```
ft2: FertileTecton
        breakTimer int = 3
        neighbours List<Tecton> = {
                ft1
        }
        myceliumCapacity int = 1
        spores Queue<Spore> = {
        }
        mushroomBody MushroomBody = null
        mycelia Queue<Mycelium> = {
                m1
```

```
        }
        occupants List<Insect> = {
        }

m1: Mycelium
        growing boolean = true
        location Tecton = ft2
        growTimer int = 1
        deathTimer int = -1

ft1: FertileTecton
        breakTimer int = 2
        neighbours List<Tecton> = {
                ft2
        }
        myceliumCapacity int = 1
        spores Queue<Spore> = {
        }
        mushroomBody MushroomBody = mb1
        mycelia Queue<Mycelium> = {
        }
        occupants List<Insect> = {
        }

ft2: FertileTecton
        breakTimer int = 2
        neighbours List<Tecton> = {
                ft1
        }
        myceliumCapacity int = 1
        spores Queue<Spore> = {
        }
        mushroomBody MushroomBody = null
        mycelia Queue<Mycelium> = {
                m1
        }
        occupants List<Insect> = {
        }

mb1: MushroomBody
        remainingEjects int = 3
        location Tecton = ft1
        mushroomSpores List<Spore> = {
                mb1-speeds1
                mb1-speeds2
                mb1-speeds3
        }

m1: Mycelium
        growing boolean = false
```

location Tecton = ft2
growTimer int = 0
deathTimer int = -1

8.2.18 Gombafonál sikeres gyors növesztése

- **Leírás**

Gombafonál sikeres gyors növesztése gombatestből FertileTectonra (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton)

(Megjegyzés: a növesztés hasonlóan működik, ha a tekton, ahova növesztünk MultiLayeredTecton, AridTecton, SustainingTecton vagy SemiFertileTecton. Akkor is hasonló a teszteset, ha gombafonálból növesztünk. A céltektonon lehetne több mint egy spóra is, ez nem változtatna a működésen.)

- **Ellenőrzött funkcionalitás, várható hibahelyek**

A gombafonál növési folyamatát, az ahhoz szükséges feltételeket és a tektonon történt változást vizsgáljuk.
Ellenőrizzük, hogy a fonál valóban rákerült-e a tektonra.
A kiválasztott FertileTectonon van egy darab spóra, ezért a gombafonál gyorsan (1 kör alatt) fog nőni.

- **Bemenet**

```
CREATE_TECTON FertileTecton ft1
SET_BREAKTIMER ft1 5
CREATE_TECTON FertileTecton ft2
SET_BREAKTIMER ft2 5
ADD_NEIGHBOUR ft1 ft2
PUT_SPORE SpeedSpore speeds1 ft2
ADD_PLAYER Mycologist mycologist1
CREATE_MUSHROOMBODY mb1 ft1
START_GAME
GROW_MYCELIUM Mycelium m1 ft2
ENDTURN
STATE ft1
STATE ft2
STATE mb1
STATE m1
```

Elvárt kimenet

```
ft1: FertileTecton
        breakTimer int = 3
        neighbours List<Tecton> = {
                ft2
        }
        myceliumCapacity int = 1
        spores Queue<Spore> = {
        }
        mushroomBody MushroomBody = mb1
        mycelia Queue<Mycelium> = {
        }
        occupants List<Insect> = {
        }
```

ft2: FertileTecton
	breakTimer int = 3
	neighbours List<Tecton> = {
		ft1
	}
	myceliumCapacity int = 1
	spores Queue<Spore> = {
		speeds1
	}
	mushroomBody MushroomBody = null
	mycelia Queue<Mycelium> = {
		m1
	}
	occupants List<Insect> = {
	}

mb1: MushroomBody
	remainingEjects int = 3
	location Tecton = ft1
	mushroomSpores List<Spore> = {
		mb1-speeds1
		mb1-speeds2
	}

m1: Mycelium
	growing boolean = false
	location Tecton = ft2
	growTimer int = 0
	deathTimer int = -1

8.2.19 Gombafonál sikertelen növesztése gombatestből, olyan FertileTectonra (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton), ahol már van gombafonál

- **Leírás**

A gombafonál nem nő rá a kiválasztott FertileTectonra, mert az kiválasztott FertileTecton már „tele” van fonállal (kapacitása és rajta lévő fonalak száma egyenlő).

(Megjegyzés: a növesztés hasonlóan működik, ha a tekton, ahova növesztünk MultiLayeredTecton, AridTecton, SustainingTecton vagy SemiFertileTecton. Akkor is hasonló a teszteset, ha gombafonálból növesztünk.)

- **Ellenőrzött funkcionalitás, várható hibahelyek**

A gombafonál növési folyamatát, az ahhoz szükséges feltételeket és a tektonon történt változást vizsgáljuk.
Ellenőrizzük, hogy a fonál valóban nem került-e a tektonra.

- **Bemenet**

```
CREATE_TECTON FertileTecton ft1
SET_BREAKTIMER ft1 5
CREATE_TECTON FertileTecton ft2
SET_BREAKTIMER ft2 5
ADD_NEIGHBOUR ft1 ft2
ADD_PLAYER Mycologist mycologist1
CREATE_MUSHROOMBODY mb1 ft1
CREATE_MYCELIUM Mycelium m1
ADD_MYCELIUM_TO_TECTON m1 ft2
START_GAME
GROW_MYCELIUM Mycelium m2 ft2
ENDTURN
ENDTURN
STATE ft1
STATE ft2
STATE mb1
STATE m1
```

- **Elvárt kimenet**

```
ft1: FertileTecton
        breakTimer int = 2
        neighbours List<Tecton> = {
                ft2
        }
        myceliumCapacity int = 1
        spores Queue<Spore> = {
        }
        mushroomBody MushroomBody = mb1
        mycelia Queue<Mycelium> = {
        }
        occupants List<Insect> = {
        }
```

ft2: FertileTecton
    breakTimer int = 2
    neighbours List<Tecton> = {
        ft1
    }
    myceliumCapacity int = 1
    spores Queue<Spore> = {
    }
    mushroomBody MushroomBody = null
    mycelia Queue<Mycelium> = {
        m1
    }
    occupants List<Insect> = {
    }

mb1: MushroomBody
    remainingEjects int = 3
    location Tecton = ft1
    mushroomSpores List<Spore> = {
        mb1-speeds1
        mb1-speeds2
        mb1-speeds3
    }

m1: Mycelium
    growing boolean = false
    location Tecton = ft2
    growTimer int = 0
    deathTimer int = -1

8.2.20 Gombafonál sikertelen növesztése gombatestből, olyan FertileTectonra (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton), ami a növést kezdeményező gombatest tektonjával nem közvetlenül szomszédos.

- **Leírás**

A gombafonál nem nő rá a kiválasztott FertileTectonra, mert az nem szomszédja a másik tektonnak, amin van a növesztést kezdeményező gombatest. Az igazi játékban nem kell kiválasztani a növést kezdeményező gombát csak azt, hogy hova növesztünk és ellenőrizzük, hogy oda tudna-e gomba növeszteni.

(Megjegyzés: a növesztés hasonlóan működik, ha a tekton, ahova növesztünk MultiLayeredTecton, AridTecton, SustainingTecton vagy SemiFertileTecton. Akkor is hasonló a teszteset, ha gombafonálból növesztünk.)

- **Ellenőrzött funkcionalitás, várható hibahelyek**

A gombafonál növési folyamatát, az ahhoz szükséges feltételeket és a tektonon történt változást vizsgáljuk.
Ellenőrizzük, hogy a fonál valóban nem került-e a tektonra.

- **Bemenet**

```
CREATE_TECTON FertileTecton ft1
SET_BREAKTIMER ft1 5
CREATE_TECTON FertileTecton ft2
SET_BREAKTIMER ft2 5
ADD_PLAYER Mycologist mycologist1
CREATE_MUSHROOMBODY mb1 ft1
START_GAME
GROW_MYCELIUM Mycelium m1 ft2
ENDTURN
ENDTURN
STATE ft1
STATE ft2
STATE mb1
```

- **Elvárt kimenet**

```
ft1: FertileTecton
        breakTimer int = 2
        neighbours List<Tecton> = {
        }
        myceliumCapacity int = 1
        spores Queue<Spore> = {
        }
        mushroomBody MushroomBody = mb1
        mycelia Queue<Mycelium> = {
        }
        occupants List<Insect> = {
        }

ft2: FertileTecton
```

breakTimer int = 2
neighbours List<Tecton> = {
}
myceliumCapacity int = 1
spores Queue<Spore> = {
}
mushroomBody MushroomBody = null
mycelia Queue<Mycelium> = {
}
occupants List<Insect> = {
}

mb1: MushroomBody
remainingEjects int = 3
location Tecton = ft1
mushroomSpores List<Spore> = {
mb1-speeds1
mb1-speeds2
mb1-speeds3
}

8.2.21 Húsevő fonál általi rovarevés és gombatest növesztés

- **Leírás**

Új kör kezdetekor a FertileTectonon (nem SustainingTecton, nem MultiLayeredTecton és nem AridTecton) lévő húsevő fonál megeszi a rajta lévő bénult állapotban lévő rovarokat és gombatestet növeszt. (Jelen esetben a vizsgált FertileTectonon még nincs gombatest.)

(Megjegyzés: a művelet hasonlóan működik, ha a tekton, ahol a gombafonál van MultiLayeredTecton, AridTecton SustainingTecton.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Megvizsgáljuk a CarnivorousMycelium működését, ha teljesülnek az ahhoz szükséges feltételek. Valamint azt, hogy megtörténik-e az ebből következő gombatest növesztés és rovarok halála.
A FertileTectonon meg kell halni a rovaroknak és nőnie kell egy új gombatestnek.

- **Bemenet**

```
CREATE_TECTON FertileTecton ft1
SET_BREAKTIMER ft1 5
PUT_SPORE StunSpore stuns1 ft1
ADD_PLAYER Mycologist mycologist1
CREATE_MYCELIUM CarnivorousMycelium cm1
ADD_MYCELIUM_TO_TECTON cm1 ft1
ADD_PLAYER Entomologist entomologist1
CREATE_INSECT ft1 i1
START_GAME
ENDTURN
EAT i1
ENDTURN
ENDTURN
STATE ft1
STATE mb-ft1
```

- **Elvárt kimenet**

```
ft1: FertileTecton
        breakTimer int = 3
        neighbours List<Tecton> = {
        }
        myceliumCapacity int = 1
        spores Queue<Spore> = {
        }
        mushroomBody MushroomBody = mb-ft1
        mycelia Queue<Mycelium> = {
                cm1
        }
        occupants List<Insect> = {
        }

mb-ft1: MushroomBody
        remainingEjects int = 3
        location Tecton = ft1
```

```
mushroomSpores List<Spore> = {
}
```

8.2.22 Gombafonál elhalása AridTectonon

- **Leírás**

AridTectonon lévő fonál elpusztul, mert már 5 köre óta van ott.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Megvizsgáljuk az AridTecton többi tektontól különböző működését.
A vizsgált AridTectonon lévő fonál el kell, hogy pusztuljon.

- **Bemenet**

CREATE_TECTON AridTecton at1
SET_BREAKTIMER at1 7
ADD_PLAYER Mycologist mycologist1
CREATE_MYCELIUM Mycelium m1
ADD_MYCELIUM_TO_TECTON m1 at1
START_GAME
ENDTURN
ENDTURN
ENDTURN
ENDTURN
ENDTURN
STATE at1

- **Elvárt kimenet**

at1: AridTecton
breakTimer int = 1
neighbours List<Tecton> = {
}
myceliumCapacity int = 1
spores Queue<Spore> = {
}
mushroomBody MushroomBody = null
mycelia Queue<Mycelium> = {
}
occupants List<Insect> = {
}

8.2.23 Rovarász megpróbál a rovarral műveletet (evés, vágás, mozgás) végrehajtani, amikor már nincs több művelete

- **Leírás**

A rovar alapesetben kétszer tud mozogni és egyszer tud fonalat vágni, valamint spórát enni. Ha már vágott fonalat vagy evett spórát vagy elhasználta az összes lépését, akkor nem tud utána semmilyen műveletet végrehajtani a rovarával. Ilyenkor csak átadni képes a kört a következő játékosnak.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Nem szabad, hogy a játékos bármilyen műveletet végre tudjon hajtani, ha már nincs több művelete.

- **Bemenet**

```
CREATE_TECTON FertileTecton ft1
SET_BREAKTIMER ft1 5
CREATE_TECTON FertileTecton ft2
SET_BREAKTIMER ft2 5
CREATE_TECTON FertileTecton ft3
SET_BREAKTIMER ft3 5
ADD_NEIGHBOUR ft1 ft2
ADD_NEIGHBOUR ft2 ft3
PUT_SPORE SpeedSpore speeds1 ft3
ADD_PLAYER Entomologist entomologist1
CREATE_INSECT ft1 i1
ADD_PLAYER Mycologist mycologist1
CREATE_MYCELIUM Mycelium m1
ADD_MYCELIUM_TO_TECTON m1 ft1
CREATE_MYCELIUM Mycelium m2
ADD_MYCELIUM_TO_TECTON m2 ft2
CREATE_MYCELIUM Mycelium m3
ADD_MYCELIUM_TO_TECTON m3 ft3
START_GAME
MOVE i1 ft2
MOVE i1 ft3
MOVE i1 ft2
EAT i1
CUT i1
STATE ft3
STATE m3
STATE i1
```

- **Elvárt kimenet**

```
ft3: FertileTecton
        breakTimer int = 4
        neighbours List<Tecton> = {
                ft2
        }
        myceliumCapacity int = 1
        spores Queue<Spore> = {
                speeds1
```

```
        }
        mushroomBody MushroomBody = null
        mycelia Queue<Mycelium> = {
                m3
        }
        occupants List<Insect> = {
                i1
        }

m3: Mycelium
        growing boolean = false
        location Tecton = ft3
        growTimer int = 0
        deathTimer int = -1

i1: Insect
        location Tecton = ft3
        maxMoves int = 2
        remainingMoves int = 0
        sporesEaten int = 0
        effectTimer int = 0
        state InsectState = NORMAL
```

8.2.24 Gombász megpróbál a körében olyan műveletet végezni, amire már nincs lehetősége

- **Leírás**

A gombász a saját körében 3 fajta műveletet képes végrehajtani. Fonalat tud növeszteni összesen egyszer. Minden hozzá tartozó gombatestel képes egyszer spórát lőni. Akármennyi új gombatestet tud növeszteni (a gombatest növesztési feltételeknek megfelelően).

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Ha a gombász megpróbálna valamit végrehajtani a műveletre vonatkozó korlát elérése után, nem szabad, hogy képes legyen rá.

- **Bemenet**

CREATE_TECTON FertileTecton ft1
SET_BREAKTIMER ft1 5
CREATE_TECTON FertileTecton ft2
SET_BREAKTIMER ft2 5
CREATE_TECTON FertileTecton ft3
SET_BREAKTIMER ft3 5
CREATE_TECTON MultiLayeredTecton mlt1
SET_BREAKTIMER mlt1 5
ADD_NEIGHBOUR ft1 ft3
ADD_NEIGHBOUR ft2 ft3
ADD_NEIGHBOUR ft1 mlt1
ADD_NEIGHBOUR ft2 mlt1
PUT_SPORE SpeedSpore speeds1 ft3
PUT_SPORE SpeedSpore speeds2 mlt1
PUT_SPORE SpeedSpore speeds3 mlt1
PUT_SPORE SpeedSpore speeds4 mlt1
ADD_PLAYER Mycologist mycologist1
CREATE_MUSHROOMBODY mb1 ft1
CREATE_MUSHROOMBODY mb2 ft2
CREATE_MYCELIUM Mycelium m1
ADD_MYCELIUM_TO_TECTON m1 mlt1
START_GAME
GROW_MYCELIUM Mycelium m2 ft3
EJECT_SPORES mb1 ft3
EJECT_SPORES mb2 ft3
GROW_MUSHROOMBODY mb3 ft3
GROW_MUSHROOMBODY mb4 mlt1
GROW_MYCELIUM Mycelium m3 mlt1
EJECT_SPORES mb1 mlt1
ENDTURN
ENDTURN
STATE ft1
STATE ft2
STATE ft3
STATE mlt1

- **Elvárt kimenet**

ft1: FertileTecton

breakTimer int = 2
neighbours List<Tecton> = {
ft3
mlt1
}
myceliumCapacity int = 1
spores Queue<Spore> = {
}
mushroomBody MushroomBody = mb1
mycelia Queue<Mycelium> = {
}
occupants List<Insect> = {
}

ft2: FertileTecton
breakTimer int = 2
neighbours List<Tecton> = {
ft3
mlt1
}
myceliumCapacity int = 1
spores Queue<Spore> = {
}
mushroomBody MushroomBody = mb2
mycelia Queue<Mycelium> = {
}
occupants List<Insect> = {
}

ft3: FertileTecton
breakTimer int = 2
neighbours List<Tecton> = {
ft1
ft2
}
myceliumCapacity int = 1
spores Queue<Spore> = {
speeds1
mb1-speeds1
mb2-speeds1
}
mushroomBody MushroomBody = mb3
mycelia Queue<Mycelium> = {
m2
}
occupants List<Insect> = {
}

mlt1: MultiLayeredTecton
breakTimer int = 2

neighbours List<Tecton> = {
ft1
ft2
}
myceliumCapacity int = 3
spores Queue<Spore> = {
speeds2
speeds3
speeds4
}
mushroomBody MushroomBody = mb4
mycelia Queue<Mycelium> = {
m1
}
occupants List<Insect> = {
}

8.2.25 Összetett teszteset, amiben rovarász és gombász és is van és a játék a valósághoz hasonlóan megy.

- **Leírás**

A rovarász a rovarjával mozog vág és eszik. A gombász pedig egyszer lő ki spórát.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

A rovarász és a gombász alapvető funkcióit ellenőrizjük.

- **Bemenet**

CREATE_TECTON FertileTecton ft1
SET_BREAKTIMER ft1 3
CREATE_TECTON FertileTecton ft2
SET_BREAKTIMER ft2 5
CREATE_TECTON FertileTecton ft3
SET_BREAKTIMER ft3 5
CREATE_TECTON MultiLayeredTecton mlt1
SET_BREAKTIMER mlt1 5
ADD_NEIGHBOUR ft1 mlt1
ADD_NEIGHBOUR ft2 mlt1
ADD_NEIGHBOUR ft3 mlt1
ADD_PLAYER Entomologist entomologist1
CREATE_INSECT ft1 i1
ADD_PLAYER Mycologist mycologist1
CREATE_MUSHROOMBODY mb1 ft1
CREATE_MYCELIUM Mycelium m1
ADD_MYCELIUM_TO_TECTON m1 ft1
CREATE_MYCELIUM Mycelium m2
ADD_MYCELIUM_TO_TECTON m2 mlt1
CREATE_MYCELIUM Mycelium m3
ADD_MYCELIUM_TO_TECTON m3 mlt1
CREATE_MYCELIUM Mycelium m4
ADD_MYCELIUM_TO_TECTON m4 ft2
START_GAME
MOVE i1 mlt1
CUT i1
ENDTURN
EJECT_SPORES mb1 mlt1
ENDTURN
EAT i1
ENDTURN
ENDTURN
SET_BREAKTIMER ft1-1 5
STATE ft1
STATE ft2
STATE ft3
STATE mlt1
STATE ft1-1
STATE mb1
STATE i1

- **Elvárt kimenet**

ft1: FertileTecton
    breakTimer int = 2
    neighbours List<Tecton> = {
        mlt1
        ft1-1
    }
    myceliumCapacity int = 1
    spores Queue<Spore> = {
    }
    mushroomBody MushroomBody = mb1
    mycelia Queue<Mycelium> = {
    }
    occupants List<Insect> = {
    }

ft2: FertileTecton
    breakTimer int = 2
    neighbours List<Tecton> = {
        mlt1
    }
    myceliumCapacity int = 1
    spores Queue<Spore> = {
    }
    mushroomBody MushroomBody = null
    mycelia Queue<Mycelium> = {
        m4
    }
    occupants List<Insect> = {
    }

ft3: FertileTecton
    breakTimer int = 2
    neighbours List<Tecton> = {
        mlt1
    }
    myceliumCapacity int = 1
    spores Queue<Spore> = {
    }
    mushroomBody MushroomBody = null
    mycelia Queue<Mycelium> = {
    }
    occupants List<Insect> = {
    }

mlt1: MultiLayeredTecton
    breakTimer int = 2
    neighbours List<Tecton> = {
        ft1
        ft2

```
            ft3
        }
        myceliumCapacity int = 3
        spores Queue<Spore> = {
        }
        mushroomBody MushroomBody = null
        mycelia Queue<Mycelium> = {
            m3
        }
        occupants List<Insect> = {
            i1
        }

ft1-1: FertileTecton
        breakTimer int = 5
        neighbours List<Tecton> = {
            ft1
        }
        myceliumCapacity int = 1
        spores Queue<Spore> = {
        }
        mushroomBody MushroomBody = null
        mycelia Queue<Mycelium> = {
        }
        occupants List<Insect> = {
        }

mb1: MushroomBody
        remainingEjects int = 2
        location Tecton = ft1
        mushroomSpores List<Spore> = {
            mb1-speeds2
        }

i1: Insect
        location Tecton = mlt1
        maxMoves int = 3
        remainingMoves int = 3
        sporesEaten int = 1
        effectTimer int = 1
        state InsectState = FAST
```

8.2.26 Rovar létrehozása és letevése

- **Leírás**

Létrehozunk egy rovarat egy tektonra. Beállitja a létrehozott rovar tektonját arra amelyikre létrehozták, és ennek az occupants listájára hozzáadja az újonnan létrehozott rovart.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Új rovar létrejött-e jó változókkal, tekton-on rajta van-e

- **Bemenet**

ADD_PLAYER Mycologist player1
CREATE_TECTON FertileTecton ft1
SET_BREAKTIMER ft1 5
CREATE_MYCELIUM Mycelium m1
ADD_MYCELIUM_TO_TECTON m1 ft1
ADD_PLAYER Entomologist player2
CREATE_INSECT ft1 i1
START_GAME
ENDTURN
STATE ft1
STATE m1
STATE i1

- **Elvárt kimenet**

ft1: FertileTecton
breakTimer int = 5
neighbours List<Tecton> = {
}
myceliumCapacity int = 1
spores Queue<Spore> = {
}
mushroomBody MushroomBody = null
mycelia Queue<Mycelium> = {
m1
}
occupants List<Insect> = {
i1
}

m1: Mycelium
growing boolean = false
location Tecton = ft1
growTimer = 0
deathTimer = -1

i1: Insect
location Tecton = ft1
maxMoves int = 2
remainingMoves int = 2
sporesEaten int = 0

effectTimer int = 0
state InsectState = NORMAL

8.2.27 Rovar mozgatása

- **Leírás**

A rovar mozog egy, a tektonjával szomszédos, tektonra

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Rovar tektonja megváltozott-e, eredeti tektonról eltűnt-e, új tektonon rajta van-e. . Rovarnak 1-e a remainingMoves-ja

- **Bemenet**

ADD_PLAYER Mycologist player1
CREATE_TECTON FertileTecton ft1
CREATE_TECTON FertileTecton ft2
ADD_NEIGHBOUR ft1 ft2
SET_BREAKTIMER ft1 5
SET_BREAKTIMER ft2 5
CREATE_MYCELIUM Mycelium m1
ADD_MYCELIUM_TO_TECTON m1 ft1
CREATE_MYCELIUM Mycelium m2
ADD_MYCELIUM_TO_TECTON m2 ft2
ADD_PLAYER Entomologist player2
CREATE_INSECT ft1 i1
START_GAME
ENDTURN
MOVE i1 ft2
STATE ft1
STATE ft2
STATE m1
STATE m2
STATE i1

- **Elvárt kimenet**

ft1: FertileTecton
breakTimer int = 5
neighbours List<Tecton> = {
ft2
}
myceliumCapacity int = 1
spores Queue<Spore> = {
}
mushroomBody MushroomBody = null
mycelia Queue<Mycelium> = {
m1
}
occupants List<Insect> = {
}

ft2: FertileTecton
breakTimer int = 5

neighbours List<Tecton> = {
ft1
}
myceliumCapacity int = 1
spores Queue<Spore> = {
}
mushroomBody MushroomBody = null
mycelia Queue<Mycelium> = {
m2
}
occupants List<Insect> = {
i1
}

m1: Mycelium
growing boolean = false
location Tecton = ft1
growTimer = 0
deathTimer = -1

m2: Mycelium
growing boolean = false
location Tecton = ft2
growTimer = 0
deathTimer = -1

i1: Insect
location Tecton = ft2
maxMoves int = 2
remainingMoves int = 1
sporesEaten int = 0
effectTimer int = 0
state InsectState = NORMAL

8.2.28 Rovar sikertelen mozgatása nem-szomszédos tektonra

- **Leírás**

A rovar mozogna egy, a tektonjával nem szomszédos, tektonra, ami nem sikerül

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Rovar tektonja megmaradt-e a régi, nem változott-e a két tekton. Rovar megtartja-e a 2 remainingMoves-t

- **Bemenet**

ADD_PLAYER Mycologist player1
CREATE_TECTON FertileTecton ft1
CREATE_TECTON FertileTecton ft2
SET_BREAKTIMER ft1 5
SET_BREAKTIMER ft2 5
CREATE_MYCELIUM Mycelium m1
ADD_MYCELIUM_TO_TECTON m1 ft1
CREATE_MYCELIUM Mycelium m2
ADD_MYCELIUM_TO_TECTON m2 ft2
ADD_PLAYER Entomologist player2
CREATE_INSECT ft1 i1
START_GAME
ENDTURN
MOVE i1 ft2
STATE ft1
STATE ft2
STATE m1
STATE m2
STATE i1

- **Elvárt kimenet**

ft1: FertileTecton
breakTimer int = 5
neighbours List<Tecton> = {
}
myceliumCapacity int = 1
spores Queue<Spore> = {
}
mushroomBody MushroomBody = null
mycelia Queue<Mycelium> = {
m1
}
occupants List<Insect> = {
i1
}

ft2: FertileTecton

breakTimer int = 5
neighbours List<Tecton> = {
}
myceliumCapacity int = 1
spores Queue<Spore> = {
}
mushroomBody MushroomBody = null
mycelia Queue<Mycelium> = {
m2
}
occupants List<Insect> = {
}

m1: Mycelium
growing boolean = false
location Tecton = ft1
growTimer = 0
deathTimer = -1

m2: Mycelium
growing boolean = false
location Tecton = ft2
growTimer = 0
deathTimer = -1

i1: Insect
location Tecton = ft1
maxMoves int = 2
remainingMoves int = 2
sporesEaten int = 0
effectTimer int = 0
state InsectState = NORMAL

8.2.29 Rovar sikertelen mozgatása olyan tektonra, amelyen nincs gombafonál

- **Leírás**

A rovar mozog egy, a tektonjával szomszédos, tektonra, de nem sikerül, mert nincs rajta gombafonál

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Rovar tektonja megmaradt-e a régi, nem változott-e a két tekton. Rovar megtartja-e a 2 remainingMoves-t

- **Bemenet**

ADD_PLAYER Mycologist player1
CREATE_TECTON FertileTecton ft1
CREATE_TECTON FertileTecton ft2
ADD_NEIGHBOUR ft1 ft2
SET_BREAKTIMER ft1 5
SET_BREAKTIMER ft2 5
CREATE_MYCELIUM Mycelium m1
ADD_MYCELIUM_TO_TECTON m1 ft1
ADD_PLAYER Entomologist player2
CREATE_INSECT ft1 i1
START_GAME
ENDTURN
MOVE i1 ft2
STATE ft1
STATE ft2
STATE m1
STATE i1

- **Elvárt kimenet**

ft1: FertileTecton
breakTimer int = 5
neighbours List<Tecton> = {
ft2
}
myceliumCapacity int = 1
spores Queue<Spore> = {
}
mushroomBody MushroomBody = null
mycelia Queue<Mycelium> = {
m1
}
occupants List<Insect> = {
i1
}

ft2: FertileTecton

breakTimer int = 5
neighbours List<Tecton> = {
ft1
}
myceliumCapacity int = 1
spores Queue<Spore> = {
}
mushroomBody MushroomBody = null
mycelia Queue<Mycelium> = {
}
occupants List<Insect> = {
}

m1: Mycelium
growing boolean = false
location Tecton = ft1
growTimer = 0
deathTimer = -1

i1: Insect
location Tecton = ft1
maxMoves int = 2
remainingMoves int = 2
sporesEaten int = 0
effectTimer int = 0
state InsectState = NORMAL

8.2.30 Rovar általi spóraevés következtében kettészakadás

- **Leírás**

A rovar megeszik egy SplitSpore-t a tektonján, aminek a hatására kettészakad

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Spóraevést követően a tektonról eltűnik-e a spóra, illetve ennek hatására létrejön-e a második rovar; mindkettő rovarnak 0 lesz-e a remainingMoves

- **Bemenet**

ADD_PLAYER Mycologist player1
CREATE_TECTON FertileTecton ft1
SET_BREAKTIMER ft1 5
CREATE_MYCELIUM Mycelium m1
ADD_MYCELIUM_TO_TECTON m1 ft1
PUT_SPORE SplitSpore splits1 ft1
ADD_PLAYER Entomologist player2
CREATE_INSECT ft1 i1
START_GAME
ENDTURN
EAT i1
STATE ft1
STATE m1
STATE i1
STATE i1-1

- **Elvárt kimenet**

ft1: FertileTecton
breakTimer int = 5
neighbours List<Tecton> = {
}
myceliumCapacity int = 1
spores Queue<Spore> = {
}
mushroomBody MushroomBody = null
mycelia Queue<Mycelium> = {
m1
}
occupants List<Insect> = {
i1
i1-1
}

m1: Mycelium
growing boolean = false
location Tecton = ft1
growTimer = 0
deathTimer = -1

i1: Insect
location Tecton = ft1
maxMoves int = 2
remainingMoves int = 0
sporesEaten int = 1
effectTimer int = 0
state InsectState = NORM AL

i1-1: Insect
location Tecton= ft1
maxMoves int = 2
remainingMoves int = 0
sporesEaten int = 0
effectTimer int = 0
state InsectState = NORMAL

8.2.31 Rovar általi spóraevés következtében Slow állapotba kerülés

- **Leírás**

A rovar megeszik egy SlownessSpore-t a tektonján, aminek a hatására Slow állapotba kerül

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Spóraevést követően a tektonról eltűnik-e a spóra, illetve ennek hatása beállitódik-e a rovaron: Slow state, 1 maxMoves, 0 remainingMoves

- **Bemenet**

ADD_PLAYER Mycologist player1
CREATE_TECTON FertileTecton ft1
SET_BREAKTIMER ft1 5
CREATE_MYCELIUM Mycelium m1
ADD_MYCELIUM_TO_TECTON m1 ft1
PUT_SPORE SlownessSpore slows1 ft1
ADD_PLAYER Entomologist player2
CREATE_INSECT ft1 i1
START_GAME
ENDTURN
EAT i1
STATE ft1
STATE m1
STATE i1

- **Elvárt kimenet**

ft1: FertileTecton
breakTimer int = 5
neighbours List<Tecton> = {
}
myceliumCapacity int = 1
spores Queue<Spore> = {
}
mushroomBody MushroomBody = null
mycelia Queue<Mycelium> = {
m1
}
occupants List<Insect> = {
i1
}

m1: Mycelium
growing boolean = false
location int = ft1
growTimer int = 0
deathTimer int = -1

i1: Insect

location Tecton= ft1
maxMoves int = 1
remainingMoves int = 0
sporesEaten int = 1
effectTimer int = 3
state InsectState = SLOW

8.2.32 Rovar általi spóraevés következtében Fast állapotba kerülés

- **Leírás**

A rovar megeszik egy SpeedSpore-t a tektonján, aminek a hatására Fast állapotba kerül

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Spóraevést követően a tektonról eltűnik-e a spóra, illetve ennek hatása beállitódik-e a rovaron: Fast state, 3 maxMoves, 0 remainingMoves

- **Bemenet**

ADD_PLAYER Mycologist player1
CREATE_TECTON FertileTecton ft1
SET_BREAKTIMER ft1 5
CREATE_MYCELIUM Mycelium m1
ADD_MYCELIUM_TO_TECTON m1 ft1
PUT_SPORE SpeedSpore speeds1 ft1
ADD_PLAYER Entomologist player2
CREATE_INSECT ft1 i1
START_GAME
ENDTURN
EAT i1
STATE ft1
STATE m1
STATE i1

- **Elvárt kimenet**

ft1: FertileTecton
breakTimer int = 5
neighbours List<Tecton> = {
}
myceliumCapacity int = 1
spores Queue<Spore> = {
}
mushroomBody MushroomBody = null
mycelia Queue<Mycelium> = {
m1
}
occupants List<Insect> = {
i1
}

m1: Mycelium
growing boolean = false
location Tecton = ft1
growTimer int = 0
deathTimer int = -1

i1: Insect

location Tecton = ft1
maxMoves int = 3
remainingMoves int = 0
sporesEaten int = 1
effectTimer int = 3
state InsectState = FAST

8.2.33 Rovar általi spóraevés következtében PreventCut állapotba kerülés

- **Leírás**

A rovar megeszik egy spórát a tektonján, aminek a hatására PreventCut állapotba kerül

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Spóraevést követően a tektonról eltűnik-e a spóra, illetve ennek hatása beállitódik-e a rovaron: CannotCut state, 0 remainingMoves

- **Bemenet**

ADD_PLAYER Mycologist player1
CREATE_TECTON FertileTecton ft1
SET_BREAKTIMER ft1 5
CREATE_MYCELIUM Mycelium m1
ADD_MYCELIUM_TO_TECTON m1 ft1
PUT_SPORE PreventCutSpore prevents1 ft1
ADD_PLAYER Entomologist player2
CREATE_INSECT ft1 i1
START_GAME
ENDTURN
EAT i1
STATE ft1
STATE m1
STATE i1

- **Elvárt kimenet**

ft1: FertileTecton
breakTimer int = 5
neighbours List<Tecton> = {
}
myceliumCapacity int = 1
spores Queue<Spore> = {
}
mushroomBody MushroomBody = null
mycelia Queue<Mycelium> = {
m1
}
occupants List<Insect> = {
i1
}

m1: Mycelium
growing boolean = false
location Tecton = ft1
growTimer int = 0
deathTimer int = 0

i1: Insect
location Tecton = ft1
maxMoves int = 2
remainingMoves int = 0
sporesEaten int = 1
effectTimer int = 3
state InsectState = CANNOT_CUT

8.2.34 Rovar általi spóraevés következtében Stunned állapotba kerülés

- **Leírás**

A rovar megeszik egy spórát a tektonján, aminek a hatására PreventCut állapotba kerül

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Spóraevést követően a tektonról eltűnik-e a spóra, illetve ennek hatása beállitódik-e a rovaron: Stun state, 0 maxMoves, 0 remainingMoves

- **Bemenet**

ADD_PLAYER Mycologist player1
CREATE_TECTON FertileTecton ft1
SET_BREAKTIMER ft1 5
CREATE_MYCELIUM Mycelium m1
ADD_MYCELIUM_TO_TECTON m1 ft1
PUT_SPORE StunSpore stuns1 ft1
ADD_PLAYER Entomologist player2
CREATE_INSECT ft1 i1
START_GAME
ENDTURN
EAT i1
STATE ft1
STATE m1
STATE i1

- **Elvárt kimenet**

ft1: FertileTecton
breakTimer int = 5
neighbours List<Tecton> = {
}
myceliumCapacity int = 1
spores Queue<Spore> = {
}
mushroomBody MushroomBody = null
mycelia Queue<Mycelium> = {
m1
}
occupants List<Insect> = {
i1
}

m1: Mycelium
growing boolean = false
location Tecton = ft1
growTimer = 0
deathTimer int = -1

i1: Insect
location Tecton = ft1
maxMoves int = 0
remainingMoves int = 0
sporesEaten int = 1
effectTimer int = 1
state InsectState = STUN

8.2.35 Rovar általi sikertelen spóraevés

- **Leírás**

A rovar megenne egy spórát a tektonján, de nem sikerül neki, mert nincs spóra

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Minden megmarad-e úgy, ahogy volt

- **Bemenet**
- *ADD_PLAYER Mycologist player1*
  CREATE_TECTON FertileTecton ft1
  SET_BREAKTIMER ft1 5
  CREATE_MYCELIUM Mycelium m1
  ADD_MYCELIUM_TO_TECTON m1 ft1
  ADD_PLAYER Entomologist player2
  CREATE_INSECT ft1 i1
  START_GAME
  ENDTURN
  EAT i1
  STATE ft1
  STATE m1
  STATE i1

- **Elvárt kimenet**

ft1: FertileTecton
breakTimer int = 5
neighbours List<Tecton> = {
}
myceliumCapacity int = 1
spores Queue<Spore> = {
}
mushroomBody MushroomBody = null
mycelia Queue<Mycelium> = {
m1
}
occupants List<Insect> = {
i1
}

m1: Mycelium
growing boolea = false
location Tecton= ft1
growTimer int = 0
deathTimer int = -1

i1: Insect
location Tecton = ft1
maxMoves int = 2

remainingMoves int = 2
sporesEaten int = 0
effectTimer int = 0
state InsectState = NORMAL

8.2.36 Rovar általi gombafonál elvágás

- **Leírás**

A rovar elvág egy fonalat a tektonján

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Rovar tektonja megváltozott-e 2 kör eltentével, eltünt-e a fonal a tektonról, elmenekült-e a rovar

- **Bemenet**

ADD_PLAYER Mycologist player1
CREATE_TECTON FertileTecton ft1
CREATE_TECTON FertileTecton ft2
ADD_NEIGHBOUR ft1 ft2
SET_BREAKTIMER ft1 5
SET_BREAKTIMER ft2 5
CREATE_MYCELIUM Mycelium m1
ADD_MYCELIUM_TO_TECTON m1 ft1
CREATE_MYCELIUM Mycelium m2
ADD_MYCELIUM_TO_TECTON m2 ft2
CREATE_MUSHROOMBODY mb1 ft2
ADD_PLAYER Entomologist player2
CREATE_INSECT ft1 i1
START_GAME
ENDTURN
CUT i1
ENDTURN
ENDTURN
ENDTURN
STATE ft1
STATE ft2
STATE mb1
STATE m1
STATE i1

- **Elvárt kimenet**

ft1: FertileTecton
breakTimer int = 3
neighbours List<Tecton> = {
ft2
}
myceliumCapacity int = 1
spores Queue<Spore> = {
}

mushroomBody MushroomBody = null
mycelia Queue<Mycelium> = {
}
occupants List<Insect> = {
}

ft2: FertileTecton
breakTimer int = 3
neighbours List<Tecton> = {
ft1
}
myceliumCapacity int = 1
spores Queue<Spore> = {
}
mushroomBody MushroomBody = mb1
mycelia Queue<Mycelium> = {
m2
}
occupants List<Insect> = {
i1
}

mb1: MushroomBody
remainingEjects int = 3
location Tecton = ft2
mushroomSpores List<Spore> = {
mb1-speeds1
mb1-speeds2
mb1-speeds3
}

m1: Mycelium
growing boolean = false
location Tecton = null
growTimer int = 0
deathTimer int = 0

i1: Insect
location Tecton = ft2
maxMoves int = 2
remainingMoves int = 2
sporesEaten int = 0
effectTimer int = 0
state InsectState = NORMAL

8.3 A tesztelést támogató programok tervei

A program JUnit 5 1.12.2-es verziójának a „Console Standalone” változatát használja. Minden teszthez tartozik egy külön metódus, amiben a teszt futtatásához szükséges parancsok le vannak írva.
A teszt „arrange” és „act” része kiadható parancsokkal van leírva. Az „assert” része a kiírt kimenetet ellenőrzi. Ezt a „TracablePrinter” osztállyal valósítsuk meg, ami a kiírt kimenetet eltárolja, ezzel vissza lehet olvasni. Az ellenőrzését a JUnit „Assertions” osztályának megfelelő függvényeivel valósítjuk meg.
A tesztek szét vannak választva külön-külön tesztosztályokba tematikájuk alapján. Ha a felhasználó egy bizonyos tesztet szeretne futtatni, vagy egy tesztosztályba tartozó teszteket akkor azt megteheti a futtatás --select-class <Tesztosztály> vagy --select-method ’<Tesztosztály>#<Tesztmetódus>’ paraméterével. Ha az összes tesztet futtatni szeretné, akkor a futtatáshoz a --scan-classpath paramétert kell megadni.

8.4 Napló

Kezdet	Időtartam	Résztvevők	Leírás
2025.04.03 ., 16:30	1 óra 30 perc	Kohár	Tevékenység: Controller osztályok kezdetleges terveinek leírása
2025.04.04 ., 17:20	2 óra	Kohár	Tevékenység: Controller osztályok tagfüggvényeinek kidolgozása
2025.04.04 ., 19:00	2 óra	Bencze Guzmics Kohár Rakos Taba	Értekezlet. Döntések: Komplexebb tesztek kigondolása Munka kiosztása
2025.04.05 ., 09:45	1 óra 45 perc	Kohár	Tevékenység: Controller osztályok javítása
2025.04.05 ., 10:00	2 óra 30 perc	Taba	Tevékenység: Parancsok és egyszerű tesztesetek kidolgozása
2025.04.05 ., 12:00	2 óra	Rakos	Tevékenység: Tesztesetek és parancsok kezdetleges megirása
2025.04.06 ., 15:00	2 óra	Taba	Tevékenység: Parancsok és egyszerű tesztesetek módosítása
2025.04.06 ., 16:00	2 óra	Bencze	Tevékenység: Parancsok és tesztesetek átemelése, illetve tesztesetek be- és kimenetének leírása
2025.04.06 ., 16:00	2 óra	Guzmics	Tevékenység: Teszt leírások elkezdése

2025.04.06 ., 18:00	3 óra 45 perc	Bencze Guzmics Rakos Taba	Értekezlet. Döntések: Tesztek egyeztetése Kiirási és használati szabályok kitalálása és pontositása A játék menetének és játékosok kiosztásának pontositása
2025.04.06 ., 22:00	30 perc	Bencze	Tevékenység: Gyűlésen elhangzottak alapján, egységesítés céljából, írási hibák javítása
2025.04.07 ., 13:00	2 óra	Taba	Tevékenység: Parancsok és egyszerű tesztesetek felülvizsgálata, összetett tesztesetek kidolgozása
2025.04.07 ., 14:00	2 óra	Rakos	Tevékenység: Tesztesetek és parancsok javitása és kiegészitése megbeszéltek alapján
2025.04.07 ., 18:20	1 óra 45 perc	Kohár	Tevékenység: Controller osztályok kiegészítése hiányzó osztályokkal
2025.04.07 ., 21:00	1 óra 30 perc	Guzmics Kohár Rakos Taba	Értekezlet. Döntések: Playerek kiosztása, tulajdonok kiosztása és világ legenerálásával kapcsolatos hiányosságok pontositása
2025.04.07 ., 23:00	1 óra 30 perc	Rakos	Tevékenység: Tesztesetek és parancsok javitása és véglegesitése

2025.04.08 ., 17:25	1 óra	Kohár	Tevékenység: Controller osztályok bonyolultabb tagfüggvényeinek leírása pszeudókóddal
2025.04.08 ., 18:00	2 óra	Guzmics	Tevékenység: Teszt leírások folytatása
2025.04.08 ., 20:00	2 óra	Bencze Guzmics Kohár Rakos Taba	Értekezlet. Döntések: Kohár által kidolgozott kontrollerek átnézése Taba és Guzmics nagyobb teszteket kezdi el Bencze és Rakos az osztály leirásokat kezdi el
2025.04.09 ., 11:15	15 perc	Kohár	Tevékenység: Pszeudókód javítása
2025.04.09 ., 18:00	2 óra 30 perc	Taba	Tevékenység: Parancsok és egyszerű tesztesetek felülvizsgálata, összetett tesztesetek módosítása
2025.04.09 ., 20:00	1 óra 30 perc	Rakos	Tevékenység: Tekton objektumok kezdetleges leirása
2025.04.10 ., 15:00	1 óra	Bencze	Tevékenység: Insect osztály leírása, tesztesetek kiegészítése játékossal
2025.04.10 ., 16:00	2 óra	Guzmics	Tevékenység: Alapvető teszt leírások befejezése
2025.04.10 ., 16:15	45 perc	Kohár	Tevékenység: Controller hiányzó függvényeinek leírása
2025.04.10 ., 18:00	2 óra	Bencze Guzmics Kohár Rakos Taba	Értekezlet. Döntések: Tesztek átnézése Osztályok kezdetleges leirásának átnézése

2025.04.10 ., 21:00	30 perc	Bencze	Tevékenység: További javítások
2025.04.11 ., 8:00	2 óra	Rakos	Tevékenység: Objektum leirásnál metódusok megirása és javitások a megbeszéltek szerint
2025.04.11 ., 10:00	2 óra	Taba	Tevékenység: Parancsok és tesztesetek felülvizsgálata, osztályleírás kidolgozása
2025.04.11 ., 19:00	2 óra	Guzmics	Tevékenység: Összetett teszt leírások és eddigi tesztek javítása
2025.04.11 ., 19:30	1 óra	Kohár	Tevékenység: Tesztelést támogató programok terveinek leírása
2025.04.11 ., 21:00	2 óra	Bencze Guzmics Kohár Rakos Taba	Értekezlet. Döntések: Az objektumok leirásával előforduló hiányosságok megbeszélése Az objektumok kihagyott metódusainak leirásának átbeszélése
2025.04.12 ., 12:00	1 óra 30 perc	Taba	Tevékenység: Parancsok és tesztesetek felülvizsgálata, osztályleírás módosítása
2025.04.12 ., 21:00	2 óra 30 perc	Bencze Guzmics Kohár Rakos Taba	Értekezlet. Döntések: Metódusok és pszeudokódok átnézése
2025.04.12 ., 22:00	1 óra	Rakos	Tevékenység: Pszeudokóddal kiegésziteni a hiányos metódusokat

2025.04.12 ., 23:30	15 perc	Bencze	Tevékenység: Megbeszélésen elhangzottak feljegyzése és elkezdése
2025.04.13 ., 9:00	2 óra	Taba	Tevékenység: Parancsok, tesztesetek és osztályleírás felülvizsgálata
2025.04.13 ., 13:00	3 óra	Bencze	Tevékenység: Megbeszélésen elhangzottak folytatása, pszeudokód írás, meglévő dolgok javítása
2025.04.13 ., 13:00	2 óra	Guzmics	Tevékenység: Objektumleírások készítése
2025.04.13 ., 16:00	1 óra	Bencze Guzmics Kohár Rakos Taba	Értekezlet. Döntések: Átnézése a legutolsó hibáknak és problémáknak
2025.04.13 ., 18:00	1 óra 15 perc	Rakos	Tevékenység: Maradék hibák kijavitása a tesztekben és objektum leirásokban Napló összesitése és megirása
2025.04.13 ., 18:00	30 perc	Taba	Tevékenység: Parancsok, tesztesetek és osztályleírás véglegesítése
2025.04.13 ., 21:00	1.5 óra	Bencze	Tevékenység: Dokumentum összevágása, átnézése és véglegesítése