1. **Részletes tervek**
   1. **Osztályok és metódusok tervei.**
      1. **Alagut**
         * **Felelősség**

Alagút megépültségének, és az alagút szájak illetve a köztük menő SínElemek számontartása.

* + - * **Ősosztályok**

Nincs

* + - * **Interfészek**

Nincs

* + - * **Attribútumok**
* **-szajA: AlagutSzaj**: Az alagút egyik szája
* **-szajB: AlagutSzaj**: Az alagút másik szája
* **-allapot: AlagutAllapot**: · NincsAlagutSzaj, EgyAlagutSzaj, VanAlagut
* **-alagutSinek: ArrayList<Sin>**: · Az alagút belsejében lévő sínek tárolása
  + - * **Metódusok**
* **+AddAlagutSzaj(AlagutSzaj a): void**: Alagút száj megépítése, ha kettő alagútszáj van akkor megépül az alagút, a két alagút száj között sinek jönnek létre. Elösször kszámítódik egy út a két alagútszáj között. Ez a két alagút szájat összekötő egyenes lesz. Majd ezen út mentén sin-ek kerülenk az alagutSinek listába, majd ezek bekerülnek a Terepasztal-ba is.
* **+RemoveAlagutSzaj(AlagutSzaj a): void**: A paraméterül kapott AlagutSzaj kivétele az attribútumok közül. Ha két alagút száj volt megépítve, akkor az alagutat lebontja. Végigmegy az alagutSinek lista elemein, és kiszedi ezeket a Terepasztalból.
  + 1. **AlagutSzaj**
       - **Felelősség**

Számon tartja, hogy meg van-e építve. Kezeli az alagút szájra történő kattintást.

* + - * **Ősosztályok**

SinElem->AlagutSzaj

* + - * **Interfészek**

Nincs

* + - * **Attribútumok**
* **-megepitve: Boolean**: Meg van-e építve az alagút ezen szája
* **-alagutSin: SinElem**: A kapcsolódó Sin
  + - * **Metódusok**
* **+onInput(): void**: A kattintásra reagál az osztály. Ha meg van építve lebontja, ha nincs, akkor pedig megépíti az adott alagút szájat.
* **+leptet(Mozdony m, SinElem s): void**: Lépteti a mozdonyt, illetve felrobbantja a vonatot, ha nincs megépítve.
* **+getKovSinElem(elozo: SinElem): SinElem:** A kapott paramétertől függően visszaadja, hogy merre kell továbbhaladnia a mozdonynak.
  + 1. **Allomas**
       - **Felelősség**

Tárol egy Színt, melyet a ráérkező vonat lekérdezhet. Bizonyos állomásokon utasok is felszállhatnak. Az állomás feladata az utasok felszállítása a kocsikra a megfelelő feltételek esetén.

* + - * **Ősosztályok**

SinElem->Allomas

* + - * **Interfészek**

Nincs

* + - * **Attribútumok**
* **-sinB: SinElem**: A következő SinElem
* **-szin: Szin**: A vagon eszerint dönti el, hogy kiürül-e
* **-ures: Boolean**: Jelzi, hogy a vagon üres-e.
  + - * **Metódusok**
* **+leptet(Mozdony m, Sinelem s): void**: Lépteti a mozdonyt.
* **+getSzin(): Szin**: Visszaadja az állomás színét.
* **+getKovSinElem(elozo: SinElem): SinElem:** Visszaadja a kapott paramétertől függőn, hogy a mozdonynak merre kell tovább haladnia.
  + 1. **App**
       - **Felelősség**

Felelőssége a view, controll és modell inicializálása.

* + - * **Ősosztályok**

Nincs

* + - * **Interfészek**

Nincs

* + - * **Attribútumok**
      * **Metódusok**
* **+main(String[] args): void:** létrehozza a Jatek, Menu, Timer osztályokat
  + 1. **BeSin**
       - **Felelősség**

Belépési pontot biztosít az új vonatoknak a Terepasztalra. Nem engedi a vonatot kimenni a terepasztalról, felrobbantja a vonatot különben.

* + - * **Ősosztályok**

SinElem->BeSin

* + - * **Interfészek**

Nincs

* + - * **Attribútumok**
* **-vonatok: int**: A pályára beadandó vonatok száma.
  + - * **Metódusok**
* **+leptet(Mozdony m, Sinelem s): void**: Lépteti a mozdonyt. Fellrobbantja a vonatot, ha ki akar menni a pályáról.
* **+VonatBead(): void**: Véletlen időközönként vonatokat ad be a pályára. Számolja, hogy mennyi idő telt el a legutóbbi beadott vonat beadása óta, és ha ez elér egy előre megadott időt, akkor új vonatot ad be a pályára. A Vonat vagonjai véletlenszerűen lesznek szeneskocsik, vagy utaskocsik.
  + 1. **Jatek**
       - **Felelősség**

Objektumok létrehozása: Terepasztal és az abban helyet foglaló Sínelemek betöltése fájlból.

Feladata a játék elvesztésének és megnyerésének a kezelése.

* + - * **Ősosztályok**

Nincs

* + - * **Interfészek**

Nincs

* + - * **Attribútumok**

Nincs

* + - * **Metódusok**
* **+start(String palya): void**: Létrehozza a Terepasztalt, és adott fájlból tölti be rá az elemeket.
* **+veszt(): void**: A játékos elveszti a játékot.
* **+nyer(): void**: A játékos megnyeri a játékot.
  + 1. **KeresztSin**
       - **Felelősség**

Pálya teljesítésének nehezítésére szolgál, több dologra kell figyelnia a játékosnak mert a vonatok összeütközhetnek itt.

* + - * **Ősosztályok**

SinElem -> KeresztSin

* + - * **Interfészek**

Nincs

* + - * **Attribútumok**
* **-sinB: SinElem**: Keresztsín egyik ága
* **-sinC: SinElem**: Keresztsín egyik ága
* **-sinD: SinElem**: Keresztsín egyik ága
  + - * **Metódusok**
* **+leptet(Mozdony m, SinElem s): void:** Lépteti a paraméterként kapott mozdonyt a jó irányba (A-ról C-re, B-ről D-re).
* **+getKovSinElem(SinElem elozo): SinElem:** Visszaadja a következő SinElemet, az előző SinElem függvényében (A-ról C-re, B-ről D-re).
  + 1. **Menu**
       - **Felelősség**

A program egyes menüpontjait tárolja. Kezeli az egyes menüpontok kiválasztása esetén bekövetkező funkciókat.

* + - * **Ősosztályok**

Nincs

* + - * **Interfészek**

Nincs

* + - * **Attribútumok**
* **-palya: String**: A kiválasztott pálya neve
* **-progress: int**: Tárolja meddig jutott el a játékos a pályákon
  + - * **Metódusok**
* **+start(String palya): void**: Új játék indítása
* **+save(): void**: Elmenti az állást, hogy mennyi pályát nyertünk meg
* **+select(): int**: Visszatér a pálya számával, amit kiválasztunk
* **+load(): int**: Betölt egy állást, és visszatér azzal, hogy meddig jutottunk el.
* **+exit(): void**: Kilép a játékból
  + 1. **Mozdony**
       - **Felelősség**

Kérdezgeti az alatta álló SínElemet, hogy melyik lesz a következő SínElem (leptet). Szól az első Kocsinak, hogy mozogjon (mozog). Megvizsgálja, hogy történt-e ütközés.

* + - * **Ősosztályok**

Szerelveny->Mozdony

* + - * **Interfészek**

Nincs

* + - * **Attribútumok**
* **-utkozott: Boolean**: Alapesetben false, ha ütközik a vonat akkor true-ra állítódik
  + - * **Metódusok**
* **+utkozik(): void**: Átállítja az utkozott változó értékét true-ra
* **+mozog(): void**: Rálép a következő SinElem-re, és mozgatja a mögötte lévő kocsit.
* **+utkozesVizsgal(): void**: Megnézi, hogy volt-e ütközés a SinElemen amin áll
  + 1. **Sin**
       - **Felelősség**

Továbbirányítja a mozdonyt a következő SínElemre.

* + - * **Ősosztályok**

SinElem->Sin

* + - * **Interfészek**

Nincs

* + - * **Attribútumok**
* **-sinB: SinElem**: A másik kapcsolódó SinElem.
  + - * **Metódusok**
* **+getKovSinElem(elozo: SinElem): SinElem**: Eldönti a paraméterként kapott SinElemből, hogy melyik ágon jön a mozdony, és aszerint tér vissza a következő SinElem-el
  + 1. **SinElem**
       - **Felelősség**

Mozgásteret biztosít a vonatok számára: adott SinElemről jött Mozdonynak megmondja, hogy melyik SinElem következik. Számon tartja, hogy hány Szerelvény tartózkodik rajta.

* + - * **Ősosztályok**

Nincs

* + - * **Interfészek**

Nincs

* + - * **Attribútumok**
* **-sinA: SinElem**: Az egyik kapcsolódó SinElem
* **-szerelvenyek: int**: Számolja a rajta tartózkodó Szerelvények száma
  + - * **Metódusok**
* **+leptet(Mozdony m, SinElem s)**: Lépteti a paraméterként kapott mozdonyt a jó irányba.
* **+elLep(): void**: Csökketni a Szerelvenyek számát
* **+raLep(UtasKocsi kocsi): void**: Növeli a Szerelvenyek számát. Akkor hívódik meg, ha UtasKocsi lép a Sinlem-re.
* **+raLep(SzenesKocsi kocsi): void**: Növeli a Szerelvenyek számát. Akkor hívódik meg, a SzenesKocsi lép a SinElem-re.
* **+onInput(): void**: A SinElem-re történő kattintást kezeli.
* **+getUtkozes(): Boolean**: Megnézi, hogy történt-e ütközés a SinElemen, vagyis több Szerelvény tartózkodik-e itt, mint 1.
* **+getKovSinElem(elozo: Sinelem): SinElem**: Visszaadja a következő SinElemet, a kapott paraméter alapján. Az egyes leszármazottak implementálják.
  + 1. **SzenesKocsi**
       - **Felelősség**

Ugyan az a felelőssége, mint az ősosztályának, a Vagonnak. Az állomásnál nem csinál semmit.

* + - * **Ősosztályok**

Szerelveny -> Vagon -> SzenesKocsi

* + - * **Interfészek**

Nincs

* + - * **Attribútumok**

Nincs

* + - * **Metódusok**

Nincs

* + 1. **Szerelveny**
       - **Felelősség**

Számon tartja, az előző és az aktuális SínElemet, amin tartózkodik. Tárolja, hogy alagútban van-e. Jelre tovább mozog.

* + - * **Ősosztályok**

Nincs

* + - * **Interfészek**

Nincs

* + - * **Attribútumok**
* **-sinElem: SinElem**: Melyik sínen található a szerelvény
* **-alagutban: Boolean**: Alagútban van-e az adott szerelvény vagy sem
* **-elozoSinElem: SinElem**: Melyik sínen volt utoljára
* **kovKocsi: Vagon**: Következő Vagon referenciája
  + - * **Metódusok**
* **+getSin(): SinElem**: Visszatér a SinElemmel amin áll éppen a szerelvény
* **+alagutValt(): void**: Átkapcsolja a szerelvényt, hogy alagútban van-e vagy sem.
* **+alagutbanVan(): Boolean**:Visszatér az alagutban attribútum értékével
  + 1. **Terepasztal**
       - **Felelősség**

A terepasztal felelőssége, hogy tárolja a SínElem-eket, BeSín-eket és Mozdony-okat. Figyeli, hogy van e a pályán még teli kocsi (győzelem ha nincs).

* + - * **Ősosztályok**

Nincs

* + - * **Interfészek**

Nincs

* + - * **Attribútumok**
* **-mozdonyok: List<Mozdony>:** A mozdonyok listája
* **-sinelemek:** **List<SinElem>:** A sínelemek listája
* **-besinek: List<BeSin>:** A bemeneti sínpárok listája
* **-teliVonatSzam: int:** a pályán utassal rendelkező vonatok száma
  + - * **Metódusok**
* **+tick(): void**: Szól a Mozdonyoknak, hogy mozogjanak.
* **+init(): void**: Feltölti elemekkel a terepasztalt.
* **+onInput(Int x, Int y): void**: Inputra reagál.
* **+addAlagutSzaj(): void**: sinelemek listához ad egy AlagutSzaj-at
* **+addMozdony(Mozdony m): void**: mozdonyok listához ad egy Mozdonyt.
* **+addBeSin(BeSin b): void**: besinek listához ad egy BeSint.
* **+addSinElem(SinElem s): void**: sinelemek listához ad egy SinElemet.
* **+removeSinElem(SinElem s): void**: sinelemek listából eltávolít egy SinElemet
* **+getAlagut(): Alagut**: Visszaadja az alagutat.
* **+vonatKiurult(): void**: teliVonatSzam-ot csökkenti 1-el.
  + 1. **Timer**
       - **Felelősség**

Felelőssége a periodikus jelgenerálás. A játék időbeli szimulálásának alapja.

* + - * **Ősosztályok**

Nincs

* + - * **Interfészek**

Nincs

* + - * **Attribútumok**

Nincs

* + - * **Metódusok**
* **+start(): void**: Elindítja a jelgenerálási folyamatot.
  + 1. **UtasKocsi**
       - **Felelősség**

Ugyanaz a felelőssége, mint szülőjének, a Vagonnak. Ezenkívül tárolja a színét, ami alapján az utasok leszállnak a kocsiból, illetve tárolja, hogy vannak-e utasok a kocsin.

* + - * **Ősosztályok**

Szerelveny -> Vagon -> UtasKocsi

* + - * **Interfészek**

Nincs

* + - * **Attribútumok**
* **-szin: Szin**: A kocsi színe. Ez alapján dönti el hogy a kocsi kiürül-e az Állomáson.
* **-ures: Boolean**: Tárolja, hogy utaznak-e a kocsiban
  + - * **Metódusok**
* **+leszallit(Allomas a): void**: Állomáshoz érve hívódik meg a függvény. A kocsi eldönti, hogy az utasok leszállnak-e, vagyis hogy az előtte lévő kocsi üres-e, és a paraméterként kapott állomás színe megegyezik-e a kocsiéval.
* **+felszall(): void**: Az állomásnál új utasok szállnak a kocsira.
* **+getSzin(): Szin**: Visszaadja a kocsi színét.
* **+getUres(): Boolean**: Visszaadja, hogy üres-e a kocsi, vagy sem.
  + 1. **Vagon**
       - **Felelősség**

Absztrakt osztály amely a mozdony után álló vagonokat testesíti meg. Feladata, hogy a következő SinElemre mozogjon, illetve értesítse a vonatban a következő vagont, hogy melyik SinElemre kell lépnie.

* + - * **Ősosztályok**

Szerelveny -> Vagon

* + - * **Interfészek**

Nincs

* + - * **Attribútumok**

Nincs

* + - * **Metódusok**
* **+mozog(SinElem kovSin): void**: frissíti az aktuális sínelemet, ahol tartózkodik, és mozgatja a következő vagont (rekurzív)
  + 1. **Valto**
       - **Felelősség**

SinElem listájából az éppen aktívnak választott ág felé irányítja a mozdonyt. Ha ág felől jön, akkor SinA felé irányítja, és automatikusan vált aktív ágat.

* + - * **Ősosztályok**

SinElem->Valto

* + - * **Interfészek**

Nincs

* + - * **Attribútumok**
* **-aktivAg: SinElem**: Aktív ág referenciája
* **-kimenetek: ArrayList<SinElem>**: Az összes kimenő SinElem referenciája
  + - * **Metódusok**
* **+setAktualisAg(SinElem s): void**: aktivAg-at átállítja s-re
* **+kovAg(): SinElem**: A kimenetektből visszaadja az aktivAg utáni SinElemet.
* **+onInput(): void**: A bevitelre az aktivAg átáll a kovAg()-ra
* **+leptet(Mozdony m, SinElem s): void**: Tovább küldi a mozdonyt a jó irányba
* **+getKovSinElem(elozo: SinElem): SinElem**: Visszatér a váltó éppen aktuális ágával, vagy a bejövő ágával, a kapott pramétertől függően.
  1. ***A tesztek részletes tervei, leírásuk a teszt nyelvén***
     1. **BeSin működik**
* **Leírás**

A Teszt célja ellenőrizni, hogy a BeSin működik, és jönnek rajta vonatok. Ehhez egy egyszerű pályát építünk, és a 2. tickben egy vonatot hozunk be a pályára.

* **Ellenőrzött funkcionalitás, várható hibahelyek**

Timer tick, Terepasztal vonatbead, BeSin vonatbead függény, várható hibahely a vonatbead függvényekben

* **Bemenet**

besinpalya.txt  
besin b1

besin b2

normalsin n1

.

b1-a n1-a

n1-b b2-a

.

b1 2 m1 p

loadmap besinpalya.txt

play

tick 3

info m1

* **Elvárt kimenet**

besinpalya.txt palya kivalasztva

jatek elindult

3 tick megtortent

m1:

sinElem: n1

alagutban: false

elozoSinElem: b1

kovKocsi: k1,

utkozott: false

* + 1. **AlagútSzáj módosít**
* **Leírás**

Ellenőrizzük, hogy az alagút felépülése után ha az egyik alagutat elvesszük, akkor megszűnik-e az alagút

* **Ellenőrzött funkcionalitás, várható hibahelyek**

Terepasztal onInputja, Alagut AddAlagutSzaj, RemoveAlagutSzaj függvénye, AlagutSzaj onInput függvénye.  
Várható hibahelyek: Nem épül meg rendesen az alagút, nem szűnik meg rendesen az alagút amikor lebontjuk

* **Bemenet**

alagutteszt.txt

besin b1

besin b2

normalsin n1

normalsin n2

alagutszaj a1

alagutszaj a2

.

b1-a n1-a

n1-b a1-a

n2-a a2-a

n2-b b2-a

.

b1 2 m1 p

loadmap alaguttesztt.txt

play

alagutszaj a1

alagutszaj a2

alagutszaj a1

info t1 alagut

* **Elvárt kimenet**

alagutteszt.txt palya kivalasztva

jatek elindult

a1 alagutszaj megepitve

a2 alagutszaj megepitve

alagut megepitve

a1 alagutszaj lerombolva

alagut lerombolva

t1:

szajA: null

szajB: a2

allapot: EgyAlagutSzaj

* + 1. **Alagút épít**
* **Leírás**

Megnézzük, hogy két alagútszáj megépítése után felépül-e az alagút rendesen.

* **Ellenőrzött funkcionalitás, várható hibahelyek**

Terepasztal onInputja, Alagut AddAlagutSzaj függvénye, AlagutSzaj oninputja.   
Várható hibahely: nem jók a referenciák, nem épül meg az alagút

* **Bemenet**

alagutteszt.txt  
besin b1

besin b2

normalsin n1

normalsin n2

alagutszaj a1

alagutszaj a2

.

b1-a n1-a

n1-b a1-a

n2-a a2-a

n2-b b2-a

.

b1 2 m1 p

loadmap alagutteszt.txt

play

alagutszaj a1

alagutszaj a2

info t1 alagut

* **Elvárt kimenet**

alagutteszt.txt palya kivalasztva

jatek elindult

a1 alagutszaj megepitve

a2 alagutszaj megepitve

alagut megepitve

t1:

szajA: a1

szajB: a2

allapot: VanAlagut

* + 1. **Alagúton átmegy**
* **Leírás**

Ellenőrizzük, hogy a vonat át tud-e menni az alagúton, ami meg van építve.

* **Ellenőrzött funkcionalitás, várható hibahelyek**

Mozdony mozog függvénye, AlagútSzáj léptet függvénye  
Várható hibahelyek: Az alagútszájnál a referencia beállítása nem történik meg rendesen

* **Bemenet**

alagutteszt.txt  
besin b1

besin b2

normalsin n1

normalsin n2

alagutszaj a1

alagutszaj a2

.

b1-a n1-a

n1-b a1-a

n2-a a2-a

n2-b b2-a

.

b1 2 m1 p

loadmap alagutteszt.txt

play

alagutszaj a1

alagutszaj a2

tick 5

info m1

* **Elvárt kimenet**

alagutteszt.txt palya kivalasztva

jatek elindult

a1 alagutszaj megepitve

a2 alagutszaj megepitve

alagut megepitve

5 tick megtortent

m1:

sinElem: as1

alagutban: true

elozoSinElem: a1

kovKocsi: k1,

utkozott: false

* + 1. **Vonat ütközik**
* **Leírás**

A megfelelő pálya elindítása után, két vonat rövid időn belül ütközik egymással. A vonatok egymással szemben haladnak a pályán.

* **Ellenőrzött funkcionalitás, várható hibahelyek**

A két vonat valóban ütközik.

Várható hibahelyek: az *utkozesVizsgal* függvényben

* **Bemenet**

utkozes.txt

besin b1

besin b2

normalsin n1

normalsin n2

normalsin n3

normalsin n4

normalsin n5

.

b1-a n1-a

n1-b n2-a

n2-b n3-a

n3-b n4-a

n4-b n5-a

b2-a n5-b

.

b1 1 m1 p

b2 1 m2 p

loadmap utkozes.txt

play

tick 2

info m1 sinelem

info m2 sinelem

tick 1

* **Elvárt kimenet**

utkozes.txt palya kivalasztva

Jatek elindult

2 tick megtortent

m1:

sinElem: n2

m2:

sinElem n4

1 tick megtortent

Utkozes tortent

* + 1. **Állomásnál leszáll**
* **Leírás**

A játék elindul, betöltődik egy teszt pálya, aminek a terepasztalán található BeSin, Sin és Állomás. Utána jön a BeSinen egy vonat, ami tartalmaz legalább egy Kocsit, a vonat halad a Sinen, amíg el nem ér egy olyan színű Állomást, ahol le tudnak szállni az utasok. Az utasok leszállnak, és lesz egy üres kocsija a vonatnak

* **Ellenőrzött funkcionalitás, várható hibahelyek**

Leszállás rendesen működik-e az Allomas és a Kocsi osztályban, azaz hogy helyesen értékeli-e ki az előtte lévő üres kocsik létét, majd helyesen ürül-e ki a Kocsi.

Várható hibahelyek: a *leszallit* függvényben

* **Bemenet**

allomas1.txt

besin b1

besin b2

normalsin n1

normalsin n2

normalsin n3

normalsin n4

normalsin n5

allomas a1 p 0

.

b1-a n1-a

n1-b n2-a

n2-b a1-a

a1-b n3-a

n3-b n4-a

n4-b n5-a

b2-a n5-b

.

b1 1 m1 p

loadmap allomas1.txt

play

tick 3

info m1-u1 ures

tick 1

info m1-u1 sinelem

info m1-u1 ures

* **Elvárt kimenet**

allomas1.txt palya kivalasztva

Jatek elindult

3 tick megtortent

m1-u1:

ures: false

1 tick megtortent

m1-u1:

sinElem: a1

m1-u1:

ures: true

* + 1. **Állomásnál nem száll le**
* **Leírás**

A játék elindul, betöltődik egy teszt pálya, aminek a terepasztalán található BeSin, Sin és Állomás. Utána jön a BeSinen egy legalább kettő, különböző színű Kocsit tartalmazó vonat, a vonat halad a Sinen, amíg el nem ér egy Állomást, aminek a Szin-e nem egyezik meg az első nemüres Kocsiéval, de megegyezik bármelyik más Kocsi színével. Az utasok nem szállnak le, és megy tovább a vonat.

* **Ellenőrzött funkcionalitás, várható hibahelyek**

Leszállás rendesen működik-e az Allomas és a Kocsi osztályban, azaz hogy helyesen értékeli-e ki az előtte lévő üres kocsik létét, majd helyesen kiürülés nélkül továbbmegy.

Várhtó hibahelyek: a *leszallit* függvényben.

* **Bemenet**

allomas2.txt

besin b1

besin b2

normalsin n1

normalsin n2

normalsin n3

normalsin n4

normalsin n5

allomas a1 p 0

.

b1-a n1-a

n1-b n2-a

n2-b a1-a

a1-b n3-a

n3-b n4-a

n4-b n5-a

b2-a n5-b

.

b1 1 m1 kp

loadmap allomas2.txt

play

tick 6

info m1-u2 ures

info m1-u2 sinelem

tick 1

info m1-u2 ures

* **Elvárt kimenet**

allomas2.txt palya kivalasztva

Jatek elindult

4 tick megtortent

m1-u2:

ures: false

m1-u2:

sinElem: a1

1 tick megtortent

m1-u2:

ures: false

* + 1. **Állomásnál felszáll**
* **Leírás**

A játék elindul, betöltődik egy teszt pálya, aminek a terepasztalán található BeSin, Sin és Állomás. Utána jön a BeSinen egy vonat, ami tartalmaz legalább egy Kocsit. A vonat halad a Sinen, amíg el nem ér egy Állomás-t, ahol leszállnak az utasok az első kocsiból. Utána halad a Sinen, és elér egy olyan Állomást, ahol fel akarnak szállni. Az utasok felszállnak, és megy tovább a vonat.

* **Ellenőrzött funkcionalitás, várható hibahelyek**

A felszállás rendesen működik-e, azaz hogy felszállnak az utasok az arra alkalmas Állomáson.

Várható hibahelyek: a *felszall* függvényben.

* **Bemenet**

allomas3.txt

besin b1

besin b2

normalsin n1

normalsin n2

normalsin n3

normalsin n4

allomas a1 p 0

allomas a2 p 1

.

b1-a n1-a

n1-b n2-a

n2-b a1-a

a1-b n3-a

n3-b a2-a

a2-b n4-a

b2-a n4-b

.

b1 1 m1 p

loadmap allomas3.txt

play

tick 5

info m1-u1 ures

tick 2

info m1-u1 ures

info m1-u1 sinelem

* **Elvárt kimenet**

allomas3.txt palya kivalasztva

Jatek elindult

4 tick megtortent

m1-u1:

ures: true

2 tick megtortent

m1-u1:

ures: false

m1-u1:

sinElem: a2

* + 1. **Állomásnál nem száll fel**
* **Leírás**

A játék elindul, betöltődik egy teszt pálya, aminek a terepasztalán található BeSin, Sin és Állomás. Utána jön a BeSinen egy vonat, ami tartalmaz legalább egy Kocsit. A vonat halad a Sinen, amíg el nem ér egy Állomás-t, ahol leszállnak az utasok az első kocsiból. Utána halad a Sinen, és elér egy olyan Állomást, ahol nem akarnak felszállni. Az utasok nem szállnak fel, és megy tovább a vonat.

* **Ellenőrzött funkcionalitás, várható hibahelyek**

A felszállás rendesen működik-e, azaz hogy nem szállnak fel az utasok az arra alkalmatlan Állomáson.

Várható hibahelyek: a *felszáll* függvényben.

* **Bemenet**

allomas4.txt

besin b1

besin b2

normalsin n1

normalsin n2

normalsin n3

normalsin n4

allomas a1 p 0

allomas a2 k 1

.

b1-a n1-a

n1-b n2-a

n2-b a1-a

a1-b n3-a

n3-b a2-a

a2-b n4-a

b2-a n4-b

.

b1 1 m1 p

loadmap allomas4.txt

play

tick 5

info m1-u1 ures

tick 2

info m1-u1 ures

info m1-u1 sinelem

* **Elvárt kimenet**

allomas3.txt palya kivalasztva

Jatek elindult

4 tick megtortent

m1-u1:

ures: true

2 tick megtortent

m1-u1:

ures: true

m1-u1:

sinElem: a2

* + 1. **Alagútszájnál felrobban**
* **Leírás**

A játék elindul, betöltődik a tesztpálya. Besinről a vonat rálép egy meg nem épült alagútszájra és megvizsgáljuk, hogy felrobban-e a vonat, illetve hogy elvesztjük-e a játékot*.*

* **Ellenőrzött funkcionalitás, várható hibahelyek**

Annak az esetnek a helyes működését ellenőrizzük, amikor a vonat meg nem épült alagútszájra lép. Hibának számít, ha nem kapunk visszajelzést a játék elvesztéséről.

* **Bemenet**

besin b1

alagutszaj s1

.

b1-a s1-a

.

b1 1 m1 p

loadmap szajnalrobban.txt

play

tick 2

* **Elvárt kimenet**

szajnalrobban.txt kivalasztva

jatek elindult

2 tick megtortent

vege vesztett

* + 1. **Váltó nem tud váltani**
* **Leírás**

Játék indulás után tick, vonat rálép besinről a váltóra. A tesztelő próbálja átváltani azt, és megvizsgáljuk, hogy sikerült-e.

* **Ellenőrzött funkcionalitás, várható hibahelyek**

A teszt célja ellenőrizni, hogy a váltót valóban nem lehet e átváltani, ha éppen vonat áll rajta. Hibának számít, ha arról kapunk visszajelzést, hogy a váltó átváltott.

* **Bemenet**

besin b1

valto v1

normalsin n1

normalsin n2

besin b2

besin b3

.

b1-a v1-a

v1-b n1-a

n1-b b2-a

v1-c n2-a

n2-b b3-a

.

b1 1 m1 p

loadmap valto.txt

play

tick 2

valto v1

* **Elvárt kimenet**

valto.txt kivalasztva

jatek elindult

2 tick megtortent

* + 1. **Váltó vált**
* **Leírás**

Játék indulása után a tesztelő átállítja a váltót, majd besinről jön a vonat, tick és megnézzük, hogy a várt ágra érkezett e a vonat.

* **Ellenőrzött funkcionalitás, várható hibahelyek**

A váltó átváltásának helyességét teszteljük. Hibának számít, ha vonatunk nem a várt sínre érkezik.

* **Bemenet**

loadmap valto.txt

play

valto v1

tick 3

info m1 sinElem

* **Elvárt kimenet**

valto.txt kivalasztva

jatek elindult

v1 valto atvaltott a c agba

3 tick megtortent

m1:

sinElem: n2

* + 1. **Játék nyer**
* **Leírás**

Egyetlen teli utaskocsiból álló vonatot engedünk besinről egy azonos színű állomásra, és megnézzük, hogy mi történik amikor kiürül.

* **Ellenőrzött funkcionalitás, várható hibahelyek**

A teszt célja a nyerés észlelésének helyessége. Hiba, ha a kocsi kiürülése után helytelen visszajelzést kapunk a játék állapotával kapcsolatban.

* **Bemenet**

besin b1

allomas a1 p

normalsin n1

besin b2

.

b1-a a1-a

a1-b n1-a

n1-b b2-a

.

b1 1 m1 p

loadmap jateknyer.txt

play

tick 3

* **Elvárt kimenet**

jateknyer.txt kivalasztva

jatek elindult

3 tick megtörtént

vege nyert

* + 1. **Játék fut**
* **Leírás**

A tesztelő kiválaszt egy pályát és a játék elindul. A teszt az első tick erejéig fut, és ha erről sikeres visszajelzés jön, akkor sikerült a teszt.

* **Ellenőrzött funkcionalitás, várható hibahelyek**

Ellenőrizzük a pálya betöltésének, játék elindításának helyességét. Hibának számít, ha nem jön visszajelzés a betöltés sikerességéről, vagy a játék állapotáról (elindult). Továbbá az is probléma, ha az első tickről nem érkezik nyugta.

* **Bemenet**

besin b1

valto v1

normalsin n1

.

b1-a v1-b

v1-a n1-a

n1-b v1-c

.

b1 1 m1 p

load jatekfut.txt

play

tick

* **Elvárt kimenet**

jatekfut.txt kivalasztva

jatek elindult

1 tick megtortent

* 1. ***A tesztelést támogató programok tervei***

A tesztelést támogató illetve kiértékelést segítő program a különböző tesztesetekre a bemeneti nyelv használatával a programnak megadja a bemeneteket, majd leellenőrzi a kimeneti nyelven érkezett adatokat, hogy megfelelnek-e az elvártnak.   
A pálya fájljait nem a program generálja, azok a tesztesetek alapján lesznek megírva, illetve a tesztelést segítő program nem nyitja meg külön a pályákat, csak utasítást ad a főprogramnak a loadmap paranccsal.

* 1. **Napló**

|  |  |  |  |
| --- | --- | --- | --- |
| **Kezdet** | **Időtartam** | **Résztvevők** | **Leírás** |
| 2017.04.01. 14:00 | 3.5 óra | Sillye | Tesztesetek: Besín működik, Alagútszáj módosít, Alagút épít, Alagúton átmegy |
| 2017.04. 01. 15:40 | 2 óra | Varga | Tesztesetek: Vonat ütközik, Állomásnál leszáll, Állomásnál nem száll le, Állomásnál felszáll, Állomásnál nem száll fel |
| 2017.04.01. 16:30 | 2.5 óra | Dócs | Tesztesetek: Játék fut, Játék nyer, Váltó vált, Váltó nem tud váltani, Alagútszájnál felrobban |
| 2017.04.01. 19:00 | 3 óra | Szili | Osztályleírások:  Alagut, AlagutSzaj, Allomas, Mozdony, Sin, SinElem, Szerelveny, Valto |
| 2017.04.01. 22:00 | 2.5 óra | Krátky | Osztályleírások: app, besin, jatek, menu, szeneskocsi, terepasztal, timer, utaskocsi, vagon |
| 2017.04.02 01:00 | 0.5 óra | Varga | Osztályleírások átnézése |
| 2017.04.02 23:00 | 0.5 óra | Szili | Dokumentum formázás |