**Virologist CLASS:**

*immunity: boolean* -> immunitásjelző

*paralyzed: boolean* -> bénulásjelző

*dancing: boolean* -> vitustáncjelző

*step()* -> végigiterál a felhasználható ágenseken és a szavatossági időt(remainingTime) csökkenti

egyel; amennyiben a remainingTime 0 egy ágensen, akkor az ágens eltűnik.

*equipGloves(g: Gloves)* -> hozzáad egy kesztyűt a listához, amikor felveszi a kezére, törli az

Inventoryból, kivéve, ha le van bénulva

*equipSacks(s: Sack)* -> hozzáad egy zsákot a listához, amikor felveszi a hátára, törli az

Inventoryból, növeli az inventory MaxMaterial számát a zsák kapacitásával, kivéve, ha le van bénulva

*equipCloack(c: Cloak)* -> hozzáad egy köpenyt a listához, törli az Inventoryból, kivéve, ha le van bénulva

*defense(Vfr: Virologist, a: Agent): boolean* -> a kesztyűt használhatja, ha van , utána a köpenyeneken végig megy: ha egy köpeny kivédte, a köpeny elhasználódik, ha nem védte ki és több köpenye van, akkor megy a kövi köpenyre és így tovább, ha nincs több köpenye, akkor a védekezés sikertelen és false-al tér vissza

*takeOffGloves()*-> eltávolítja a virológusról az eszközt, visszarakja az inventoryba, ha van ott még hely, egyébként eltűnik, kivéve, ha le van bénulva

*takeOffSack()*-> eltávolítja a virológusról az eszközt, visszarakja az inventoryba, ha van ott még hely, egyébként eltűnik, kivéve, ha le van bénulva

*takeOffCloak()*-> eltávolítja a virológusról az eszközt, visszarakja az inventoryba, ha van ott még hely, egyébként eltűnik, kivéve, ha le van bénulva

*touchy(l: Lab)* ->meghívja a *learnGCode*-t, ha a visszatérés true, akkor megvizsgálja, hogy meg van-e mindegyik kód, ha igen, akkor meghívja az *endGame()*-t.

*createAgent(g: GCode)* -> meghívja a *create()*-t, az ágens időzítőjét elindítja, eltárolja az ágenst a

felhasználható ágensek listájába, ha le van bénulva, akkor nem tud ágenst létrehozni

*attack(vTo: Virologist, a: Agent)* -> meghívja a megtámadott virológuson az *assaulted()*-t;

ha a támadás sikeres volt és a bénitó ágenssel támadott, akkor a kiválasztott anyagokat és felszereléseket elveheti a megtámadott virológustól, amennyiben az inventory kapacitása engedi(*stealBy() meghívása*), ha le van bénulva, akkor nem tud támadni

*assaulted(vFr: Virologist, a: Agent): boolean* -> megnézi, hogy van-e már rajta hatás, ha igen, akkor nem kapja meg, ha nincsen, akkor meghívja a *defense()*-t aminek a visszatérése alapján kapja

meg a hatást; visszatérés megadja, hogy az ágens hatása rákerül-e a megtámadott virológusra;

ha rákerül a hatás, akkor az ágens *virologistUnderEffect*-je lesz ez a virológus és az ágens *remaingingTime*-ját inicializáljuk az *effectTime*-al

*move(f: Field)* -> a játékos választása alapján egy szomszédos mezőre lép, meghívja annak a mezőnek

az *arrive()-*ját, ha a virológus le van bénulva, akkor nem tud lépni

*stealBy(v: Virologist)* -> megnézi a virológus inventory-ját, kiválogatja a lopandó dolgokat, azokat átadja a másik virológusnak; amennyiben zsákot loptak és a meglopott virológusnak ezáltal kevesebb a tárhelye, mint amennyi anyag van nála, akkor a kölünbséget elveszti(random anyagokat)

**Equipment CLASS:**

*pickUp(v: Virologist)* -> absztrakt

**Gloves CLASS:**

*pickUp(v : VIrologist)* -> Inventoryba rakja a kesztyűt

*use(vFr: Virologist, a: Agent): boolean* -> eldönti, hogy használja-e vagy sem, ha igen, akkor

visszadobja a támadóra az ágenst, visszatérés: használta v sem

**Sack CLASS:**

*capacity: int* -> a szám, amivel növeljük az inventory *MaxMaterial*-ját

*pickUp(v: Virologist)* –> hozzáadja az Inventoryhoz a zsákot

**Cloak CLASS:**

*pickUp(v :Virologist)* -> hozzáadja az Inventoryhoz a köpenyt

*evade(): boolean* -> eldönti, hogy kivédi-e a támadást(82,3% eséllyel)

**Inventory CLASS:**

*MaxMaterial: int* -> maximálisan eltárolható anyagok száma

*MaxEquipment: int* -> maximálisan tárolható felszerelések száma

*addEqToInv(sh: Shelter):* -> ha van hely az Inventoryban, akkor elkéri az óvóhelytól a felszerelést

és törli az óvóhelyről a felszerelést, ha nincs hely nem történik semmi

*removeGloves()* -> eltávolítja az Inventoryból a felszerelést

*removeSack()* -> eltávolítja az Inventoryból a felszerelést

*removeCloak()* -> eltávolítja az Inventoryból a felszerelést

*learnGCode(l: Lab): boolean* -> letapogatja a kódot, de előtte megnézi, hogy már ismeri-e, ha

igen, akkor nem adja hozzá, ha nem, akkor hozzáadja a listához. Visszatérés: megismert-e új kódot

*addAmino()* -> hozzáad egy aminosavat az inventoryhoz

*addNukleo()* -> hozzáad egy nukleotidot az inventoryhoz

*removeAmino()* -> eltávolít egy aminosavat az inventoryból

*removeNukleo()* -> eltávolít egy nukleotidot az inventoryból

**Shelter CLASS:**

*arrive(v: Virologist) ->* ha rálép egy virológus, akkor meghívja a *wannaPickUp()-*t

ha visszatérés true, akkor meghívja a *pickUp()*-t

*removeEp(ep: Equipment)*-> leveszi a felszerelést az óvóhely mezőről

*refillEq(eq: Equipment)* -> újratölti felszerelésekkel az óvóhelyet

*wannaPickUP(): boolean* -> megkérdezi a felhasználót(ha a virológus inventory-ja nem fullos), hogy akar-e eszközt felvenni

**Field CLASS:**

*arrive(v: Virologist)* -> a virológust eltávolítja az előző mezőjéről és hozzáadja ahhoz, amelyikre

lépni akar, beállítjuk a virológus *field* változóját az új mezőre, végig iterál a mezőn lévő virológusokon és amennyiben bénult virológust talál, meghívja a *wannaSteal()*-t, ha ez true, akkor meghívja a *steal()*-t

*addVirologist(v: Virologist)* ->hozzáadja a mező listájához a virológust

*remove*

**Lab CLASS:**

*arrive(v: Virologist)* -> basic arrive(Field CLASS) + *wannaTouchy()*, ha true, akkor *touchy()*

*wannaTouchy(): boolean* -> megkérdezzük a felhasználót, hogy akar-e kódot letapogatni

**Storage CLASS:**

*arrive(v: Virologist)* -> basic arrive(Field CLASS)+ *wannPickUp()*, ez true akkor *pickUp()*

*removeMat()* -> kiveszi a raktár listájából a legfelső anyagot, amennyiben nincs több anyag, akkor *refill()*

*refill()* -> újratölti a raktárt anyagokkal

*wannaPickUp(): boolean* -> megkérdezi a felhasználót, hogy akar-e anyagot felvenni

**GCode CLASS:**

*gCName: String* ->kód által előállítható ágens neve

*create(inv: Inventory): Agent* -> az ágens létrehozásához szükséges anyagokat ellenőrzi az

inventoryban, ha nincs benne, akkor 0, ha van elég anyag, akkor kiveszi őket az

inventoryból és létrehozza az ágenst (megfelelő konstruktort meghívja, amiben az *remainingTime* inicializálódik az *experiation* értékével) és visszaadja az ágenst

**Game CLASS:**

*MaxGCode: int* -> maximális megismerhető genetikai kódok száma.

*startGame()* -> elndítja a játékot, inicializál, létrehozza a világot

*endGame(v: Virologist)* -> leállítja a játékot, minden benn futó folyamatot, értesíti a játokosokat

a játék végéről

**Agent CLASS:**

*remainingTime: int* -> fennmaradt szavatossági idő, fennmaradt hatás ideje

*exepriation: int* -> szavatossági idő (const)

*effectTime: int* -> hatás ideje (const)

*step()* -> absztrakt

**VitusDance CLASS:**

*step()* -> minden körben csökkenti a *remainingTime*-t, *dancing-*et beállítja true-ra és egy random

szomszédos mezőre lép a virológus minden körben, ameddig a *remainingTime* nem 0, ha 0, akkor

törlődik az ágens és a *dancing*-et false-ra állítja

**Immunity CLASS:**

*step()* -> a virológus *immunity—*jét true-ra állítja minden körben, a *remainingTime*-t csökkenti 1-

l, ha ez 0, akkor az  *immunity*-t false-ra állítja és törlődik az ágens

**Paralisys CLASS:**

*step()* -> a virológus *paralyzed*-jét true-ra állítja minden körben a *remainingTime*-t csökkenti 1-el

ha ez 0, akkor a *paralyzed*-t false-ra állítja és törlődik az ágens

**Oblivion CLASS:**

*step()* -> időtartama 1 kör! Törli a virológus invetory-jából az összes ismert kódot

**Material CLASS:**

*pickUp(v: Virologist)* -> absztrakt függvény

**AminoAcid CLASS:**

*pickUp(v: Virologist)* -> meghívja az *addAmino()*-t

**Nukleotid CLASS:**

*pickUp(v: Virologist)* -> meghívja az *addNukleo()*-t