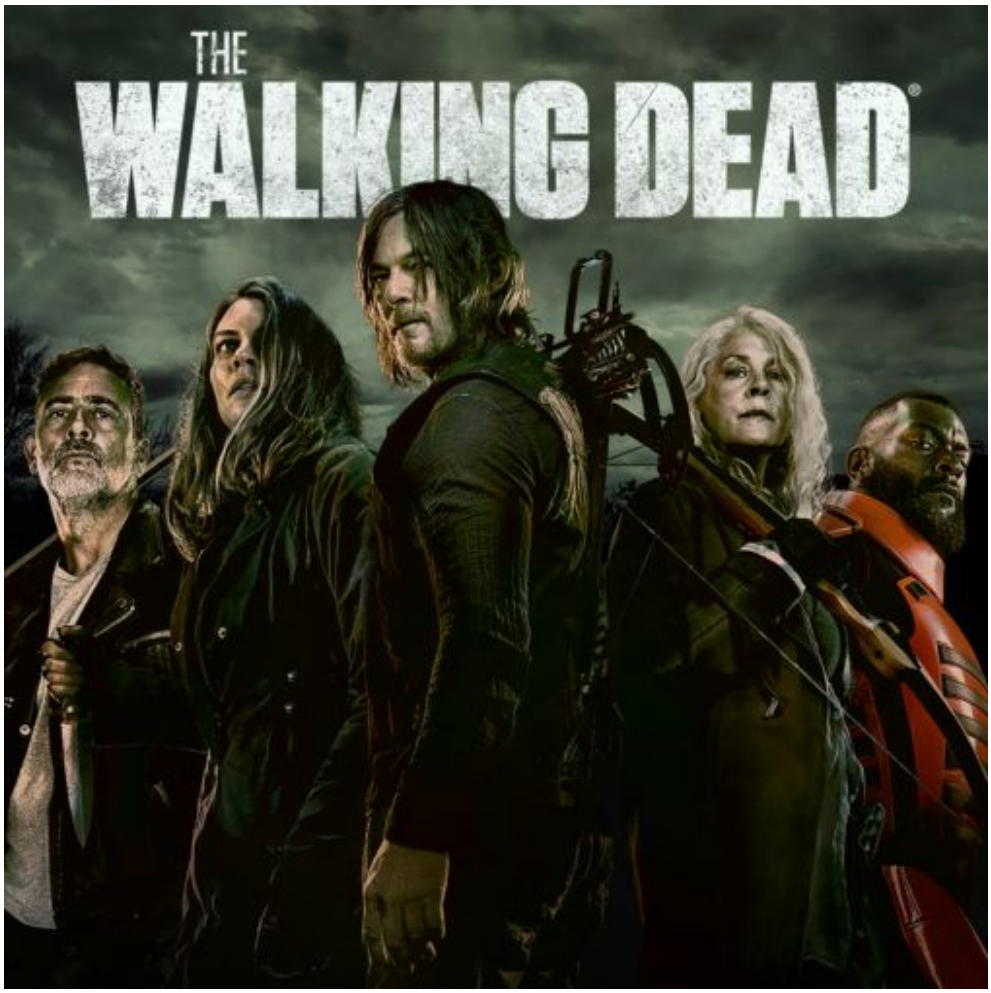
The Walking Dead

Albert Oliveras

30 d'abril de 2024



1 Regles del Joc

Per motius que encara ningú coneix amb certesa, l'apocalipsi zombi s'ha apoderat de la Terra. Els pocs humans que hi queden intenten sobreviure i apoderar-se dels escassos recursos encara existents.

Es tracta d'un joc per a quatre jugadors, identificats amb números de 0 a 3. Cada jugador té el control d'un clan d'unitats vives. Però al joc coexisteixen dos tipus més d'unitats: unitats mortes i zombis.

El joc té una durada de 200 rondes, numerades de l'1 al 200. Cada unitat pot moure's com a màxim una vegada per ronda. Les unitats mortes, com és d'esperar, no es poden moure. Durant aquestes rondes els clans aniran acumulant punts i guanyarà la partida qui tingui més punts en acabar la ronda 200.

El tauler del joc té dimensions 60×60. Les unitats no es poden moure en cap cas fora d'ell. Una posició del tauler ve determinada per un parell d'enters (f, c) on $0 \leq f < 60$ i $0 \leq c < 60$. La posició de més a dalt i a l'esquerra és la $(0, 0)$, mentre que la de més a baix i a la dreta és $(59, 59)$. Per tant, la primera coordenada (f de fila) és la que indica la posició en l'eix vertical i la segona (c de columna) en l'eix horitzontal. Cada cel·la del tauler o bé forma part d'un carrer o bé està plena de deixalla. Les unitats no poden trepitjar la deixalla i s'han de moure necessàriament pels carrers.

Els clans comencen la partida amb una certa quantitat de punts de força. La *força d'un clan* es defineix com $\left\lfloor \frac{\texttt{punts força}}{\texttt{unitats vives}} \right\rfloor$ i serà clau per determinar el guanyador de les lluites que hi haurà durant la partida. En acabar cada ronda, els punts de força d'una clan es decrementaran en una quantitat igual al nombre d'unitats vives d'aquest clan. No obstant, mai esdevindrà una quantitata negativa. Per tal d'incrementar els punts de força, les unitats vives poden agafar menjar que trobaran al tauler. És fàcil veure que un clan amb moltes unitats vives necessita recollir molt menjar per a poder mantenir una força considerable. És per això que algunes unitats decideixen canviar de clan: cada ronda, amb un 20% de probabilitat, una unitat del clan amb més unitats vives passa a formar part del clan amb menys unitats vives. En cas d'haver-hi diferents clans amb aquestes propietats, se'n selecciona un d'ells de forma aleatòria.

Moviments de les unitats vives. Una unitat viva d'un clan es pot moure pel tauler de la manera següent:

- Només pot accedir a les cel·les adjacents en horitzontal i vertical, mai en diagonal.
- Si es mou cap a una cel·la ocupada per deixalla, per una unitat del seu mateix clan o per una unitat morta, el moviment s'ignorarà.
- Si es mou cap a una cel·la ocupada per menjar, els punts de força del seu clan s'incrementaran, la unitat ocuparà la cel·la tot fent desaparèixer el

menjar i el seu clan posseirà aquesta cel·la. En acabar la ronda, una unitat de menjar reapareixerà en una altra posició. Notem que no trobarem mai una cel·la ocupada per menjar i per una unitat.

- Si es mou cap a una cel·la buida (sense menjar ni cap unitat), el moviment s'efectuarà i el seu clan passarà a posseir aquesta cel·la.
- Si es mou cap a una cel·la ocupada per un zombi, el zombi morirà però la unitat no es mourà. Com a resultat, el clan rebrà un cert nombre de punts i, en acabar la ronda, una unitat viva d'aquest clan reapareixerà en una altra posició reemplaçant al zombi, que desapareixerà.
- Si es mou cap a una cel·la ocupada per una unitat viva d'un altre clan, es produirà una lluita. La unitat que perdi la lluita passarà a ser una unitat morta, i s'inicia el seu procés de conversió: al cap d'un cert nombre de rondes, passarà a ser un zombi. Si ja havia començat el procés de conversió per haver estat mossegat per un zombi, el procés torna a començar. El clan de la unitat que guanyi l'atac rebrà un cert nombre de punts, però no es mourà de posició. El guanyador de la lluita es determina de la manera següent:

  Amb un 30% de probabilitat, la unitat que ha desencadenat l'atac agafarà l'altra per sorpresa i serà la guanyadora sense haver de lluitar. En cas contrari, si les forces dels dos clans involucrats en la lluita són N i M, respectivament, el primer guanyarà amb probabilitat $N/(N+M)$ i el segon amb probabilitat $M/(N+M)$. En cas que $N = M = 0$, les dues unitats tindran la mateixa probabilitat de guanyar.

Moviments dels zombis. Els zombis no formen part de cap clan i per tant no podran ser controlats per cap jugador. Els zombis sempre es mouran cap a la unitat viva més propera a ells, tenint en compte la presència d'obstacles. En cas d'haver-n'hi diverses a la mínima distància, n'escolliran una a l'atzar. Els moviments d'un zombi es regiran per les següent regles:

- Pot accedir a les cel·les adjacents en horitzontal i vertical, i també en diagonal.
- Mai es mourà cap a una cel·la ocupada per deixalla, per una unitat morta o per un zombi.
- Si es mou cap a una cel·la ocupada per menjar, el menjar desapareixerà. En acabar la ronda, una unitat de menjar reapareixerà en una posició aleatòria. Si un clan posseïa aquesta cel·la, deixarà de fer-ho.
- Si es mou cap a una cel·la buida (sense menjar ni cap unitat) posseïda per un clan, aquest perdrà la possessió.
- Si es vol moure cap a una cel·la ocupada per una unitat viva, el moviment no s'efectuarà. No obstant, mossegarà a la unitat i aquesta iniciarà el seu procés de conversió a zombi, que tindrà lloc al cap d'un cert nombre de

rondes. Si aquest procés ja s'havia iniciat per una mossegada prèvia, el procés de conversió no tornarà a començar sinó que continuarà el seu curs. Durant la conversió a zombi, la unitat es comportarà igual que una unitat viva.

Com a resultat de les anterior regles, el nombre d'unitats totals és constant al llarg de tot el joc.

Regeneració d'objectes. Cada vegada que cal regenerar una unitat de menjar o una unitat viva, aquesta reapareixerà sempre en una cel·la buida C i tal que no hi ha cap unitat ni menjar en les posicions que l'envolten (les marcades amb una x a la taula):

x	x	x	x	x
x	x	x	x	x
x	x	C	x	x
x	x	x	x	x
x	x	x	x	x

Si en el moment de reaparèixer no existeix cap cel·la segura en aquest sentit, l'objecte reapareixerà en una cel·la buida, que no tingui cap unitat ni menjar.

És important remarcar que les unitats tenen un identificador que mai canvia, ni tan sols durant el procés de regeneració. És a dir, si una unitat amb un cert identificador inicial es converteix en zombi, aquest continuarà amb el mateix identificador. Si posteriorment el zombi mor i reapareix com una unitat viva d'un altre clan, l'identificador es mantindrà.

Càlcul de la puntuació. La puntuació d'un clan en una ronda ve determinada per dos components. D'una banda, per cada zombi que el clan hagi matat fins aquell moment s'obtindran 10 punts. Per cada unitat que hagi matat s'obtindran 50 punts.

D'altra banda, per cada cel·la posseïda pel clan **en aquesta ronda** s'obtindrà 1 punt. La puntuació del clan és la suma d'aquestes quantitats. Així doncs, la puntuació es pot decrementar si es perd la possessió d'algunes cel·les. Les constants 10, 50 i 1, així com d'altres que especifiquen els paràmetres inicials del joc, estan definides a l'arxiu d'entrada `default.cnf`. Totes les partides es jugaran amb exactament els valors donats en aquest arxiu.

Execució d'ordres. A cada ronda es pot donar més d'una ordre a la mateixa unitat, tot i que només se seleccionarà la primera d'elles (si n'hi ha alguna). Tot programa que intenti donar més de 1000 comandes durant la mateixa ronda s'avortarà.

Cada ronda, les ordres seleccionades dels quatre jugadors s'executaran amb ordre aleatori, però respectant l'ordre relatiu entre les unitats d'un mateix clan.

Com a conseqüència de la norma anterior, considereu la possibilitat de donar les ordres a les vostres unitats a cada ronda de més urgent a menys urgent.

Tingueu en compte que s'aplica cada moviment sobre el tauler que resulta dels moviments anteriors. Per exemple, considereu el tauler

x	x	x	x
x	M	U	x
x	V	x	x
x	x	x	x

on M representa menjar i U i V dues unitats vives de diferents clans. Imaginem que el jugador que controla U ha decidit que aquest vagi cap a l'esquerra, i el jugador que controla V ha decidit que vagi amunt. Si s'executa primer el moviment de V, aleshores U s'ha quedat sense menjar, perquè V ja l'ha consumit i a més, la posterior execució del moviment de U és un atac cap a V del qual en pot sortir malparat. Si U mor durant aquest atac, en la visualització de la partida veurem una transició de la matriu anterior cap a una situació on U ha desaparegut. Òbviament, no hi podia haver un atac entre U i V des de la configuració inicial, perquè les unitats vives no es poden moure en diagonal, però l'ordre d'execució de les comandes sí que ho ha fet possible. Tingueu això en compte quan no entengueu certes situacions durant la visualització de les partides.

Després de l'execució de tots els moviments dels jugadors, es procedeixen a efectuar els moviments dels zombis. En acabar, els següents processos s'executaran en aquest ordre: les unitats que han acabat el seu procés de conversió passaran a ser zombis, els zombis morts es convertiran en unitats vives, una unitat del clan amb més unitats vives potser passarà a formar part del clan amb menys unitats vives, es regeneraran les unitats de menjar consumides i s'actualitzaran les puntuacions.

2 El Visor

A continuació descrivim el visor de partides:

- A la part superior hi ha botons que permeten reproduir o pausar la partida, anar al començament o al final de la partida, activar o desactivar el mode d'animació o obtenir una finestra d'ajuda amb més maneres de controlar com es reprodueix la partida. També hi trobareu la ronda actual i un botó per tancar el visor. Una barra de desplaçament horitzontal mostra visualment en quin punt de la partida es troba la ronda actual.
- A la columna de l'esquerra, apareix cada jugador amb el nom i color corresponents. A sota es mostra la puntuació actual, el nombre d'unitats vives i la força del jugador corresponent. A les partides jugades a Jutge.org, també es mostra el percentatge de temps de CPU que s'ha con-

sumit fins ara (si està esgotat, s'indica amb un 'out'). A la part superior dreta apareixen els colors dels jugadors ordenats per puntuació.

- Les cel·les no posseïdes per ningú són de color blanc. En cas contrari, tenen el color del jugador que les posseeix.
- Les cel·les amb deixalles tenen color gris fosc.
- Les unitats vives es representen amb un cercle del color corresponent. En cas d'estar en procés de conversió, a zombi, tenen forma quadrada.
- Les unitats mortes es representen amb una creu.
- Els zombis es representen amb un quadrat vermell i contorn exterior negre.
- Les unitats de menjar es representen amb un cercle vermell i contorn exterior negre.

3 Com programar un jugador

El primer que heu de fer és descarregar-vos el codi font. Aquest inclou un programa C ++ que executa les partides i un visualitzador HTML per veure-les en un format raonable i animat. A més, us proporcionem un jugador "Null" i un jugador "Demo" per facilitar el començament de la codificació del jugador.

3.1 Executar la primera partida

Aquí us explicarem com executar el joc sota Linux, però hauria de funcionar també sota Windows, Mac, FreeBSD, OpenSolaris, ... Només necessiteu una versió recent `g ++`, el `make` instal·lat al sistema, a més d'un navegador modern com Firefox o Chrome.

1. Obriu una consola i feu `cd` al directori on us heu descarregat el codi font.
2. Si, per exemple, teniu una versió de Linux en 64 bits, executeu:

   ```
   cp AIDummy.o.Linux64 AIDummy.o
   ```

   ```
   cp Board.o.Linux64 Board.o
   ```

   Amb altres arquitectures, cal escollir els objectes adequats que trobareu al directori.
3. Executeu

   ```
   make all
   ```

   per compilar el joc i tots els jugadors. Tingueu en compte que el `Makefile` identifica com a jugador qualsevol fitxer que coincideixi amb `AI*.cc`

4. Es crea un fitxer executable anomenat `Game`. Aquest executable us permet executar una partida mitjançant una comanda com la següent:

   ```
   ./Game Demo Demo Demo Demo -s 30 < default.cnf > default.res
   ```

   Aquesta comanda comença una partida, amb la llavor aleatòria 30, amb quatre instàncies del jugador Demo, al tauler definit a `default.cnf`. La sortida d'aquesta partida es redirigeix a `default.res`.

5. Per veure una partida, obriu el fitxer visualitzador `viewer.html` amb el navegador, per exemple executant des d'un terminal la comanda `firefox viewer.html`, i carregueu el fitxer `default.res`.

Utilitzeu

```
./Game --help
```

per veure la llista de paràmetres que es poden usar. Particularment útil és

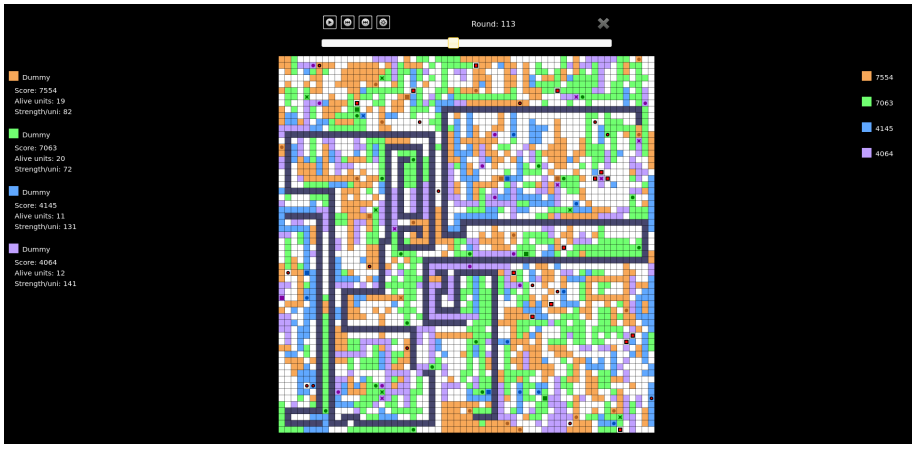
```
./Game --list
```

per veure tots els noms de jugadors reconeguts.

En cas que sigui necessari, recordeu que podeu executar

```
make clean
```

per esborrar l'executable i els objectes i començar la compilació de nou.



3.2 Afegir el vostre jugador

Per crear un jugador nou amb, per exemple, nom `Rick`, copieu `AINull.cc` (un jugador buit que proporcionem com a plantilla) a un fitxer nou `AIRick.cc`. A continuació, editeu el fitxer nou i canvieu la línia

#define PLAYER_NAME Null

a

#define PLAYER_NAME Rick

El nom que trieu pel vostre jugador ha de ser únic, no ofensiu i tenir com a màxim 12 caràcters. Aquest nom es mostrarà al lloc web i durant les partides.

A continuació, podeu començar a implementar el mètode virtual *play* (), heretat de la classe base *Player*. Aquest mètode, que serà cridat a cada ronda, ha de determinar les ordres que s'enviaran a les vostres unitats.

Podeu utilitzar definicions de tipus, variables i mètodes a la vostra classe de jugador, però el punt d'entrada del vostre codi serà sempre el mètode *play* ().

Des de la vostra classe jugador també podeu cridar funcions que trobareu especificades als arxius següents:

- `State.hh`: accedir a l'estat del joc.
- `Action.hh`: donar ordres a les vostres unitats.
- `Structs.hh`: estructures de dades útils.
- `Settings.hh`: accedir als paràmetres del joc.
- `Player.hh`: mètode *me*().
- `Random.hh`: generar nombre aleatoris.

Trobareu un resum de tota aquesta informació a l'arxiu `api.pdf`. També podeu examinar el codi del jugador "Demo" a `AIDemo.cc` com a exemple de com usar aquestes funcions.

Tingueu en compte que no heu d'editar el mètode *factory* () de la classe del vostre jugador, ni l'última línia que afegeix el vostre jugador a la llista de jugadors disponibles.

3.3 Restriccions en enviar el vostre jugador

Quan creieu que el vostre jugador és prou fort per entrar a la competició, podeu enviar-lo al Jutge. Degut a que s'executarà en un entorn segur per prevenir trampes, algunes restriccions s'apliquen al vostre codi:

- Tot el vostre codi font ha d'estar en un sol fitxer (com AIRick.cc).
- No podeu utilitzar variables globals (en el seu lloc, utilitzeu atributs a la vostra classe).
- Només teniu permès utilitzar biblioteques estàndard com `iostream`, `vector`, `map`, `set`, `queue`, `algoritme`, `cmath`, ... En molts casos, ni tan sols cal incloure la biblioteca corresponent.
- No podeu obrir fitxers ni fer cap altra crida a sistema (threads, forks, ...)

- El vostre temps de CPU i la memòria que utilitzeu seran limitats, mentre que no ho són al vostre entorn local quan executeu `./Game`.
- El vostre programa no ha d'escriure a **cout** ni llegir de **cin**. Podeu escriure informació de depuració a **cerr**, però recordeu que fer això en el codi que envieu al Jutge pot fer malgastar innecessàriament temps de CPU.
- Qualsevol enviament al Jutge ha de ser un intent honest de jugar. Qualsevol intent de fer trampes de qualsevol manera serà durament penalitzat.
- Un cop hagueu enviat un jugador al Jutge que hagi derrotat al Dummy, podeu fer més enviaments però haureu de canviar el nom del jugador. És a dir, un cop un jugador ha vençut al Dummy, el seu nom queda bloquejat i no es pot reutilitzar.

4 Consells

- **NO DONEU O DEMANEU EL VOSTRE CODI A NINGÚ**. Ni tan sols una versió antiga. Ni fins i tot al vostre millor amic. Ni tans sols d'estudiants d'anys anteriors. Utilitzem detectors de plagi per comparar els vostres programes, també contra enviaments de jocs d'anys anteriors. No obstant, podeu compartir arxius objecte.

  Qualsevol plagi implicarà **una nota de 0 en l'assignatura** (no només del Joc) de tots els estudiants involucrats. Es podran també prendre mesures disciplinàries addicionals. Si els estudiants A i B es veuen implicats en un plagi, les mesures s'aplicaran als dos, independentment de qui va crear el codi original. No es farà cap excepció sota cap circumstància.
- Abans de competir amb els companys, concentreu-vos en derrotar al Dummy.
- Llegiu les capçaleres de les classes que aneu a utilitzar. No cal que mireu les parts privades o la implementació.
- Comenceu amb estratègies simples, fàcils d'implementar i depurar, ja que és exactament el que necessitareu al principi.
- Definiu mètodes auxiliars senzills (però útils) i *assegureu-vos que funcionin correctament*.
- Intenteu mantenir el vostre codi net. Això farà més fàcil canviar-lo i afegir noves estratègies.
- Com sempre, compileu i proveu el vostre codi sovint. És *molt* més fàcil rastrejar un error quan només heu canviat poques línies de codi.
- Utilitzeu **cerr** per produir informació de depuració i afegiu `asserts` per assegurar-vos que el vostre codi fa el que hauria de fer.

- Quan depureu un jugador, elimineu els **cerr**s que tingueu en el codi d'altres jugadors, per tal de veure només els missatges que desitgeu.
- Podeu utilitzar comandes com el `grep` de Linux per tal de filtrar la sortida produïda per `Game`.
- Activeu l'opció `DEBUG` al Makefile, que us permetrà obtenir traces útils quan el vostre programa avorta. També hi ha una opció `PROFILE` que podeu utilitzar per optimitzar codi.
- Si l'ús de **cerr** no és suficient per depurar el vostre codi, apreneu com utilitzar `valgrind`, `gdb` o qualsevol altra eina de depuració.
- Podeu analitzar els arxius produïts per `Game`, que descriuen com evoluciona el tauler a cada ronda.
- Conserveu una còpia de les versions antigues del vostre jugador. Feu-lo lluitar contra les seves versions anteriors per quantificar les millores.
- Assegureu-vos que el vostre programa sigui prou ràpid. El temps de CPU que es permet utilitzar és bastant curt.
- Intenteu esbrinar les estratègies dels altres jugadors observant diverses partides. D'aquesta manera, podeu intentar reaccionar als seus moviments, o fins i tot imiteu-los o milloreu-los amb el vostre propi codi.
- No espereu fins al darrer minut per enviar el jugador. Quan hi ha molts enviaments al mateix temps, el servidor triga més en executar les partides i podria ser ja massa tard!
- Podeu enviar noves versions del vostre programa en qualsevol moment.
- Recordeu: mantingueu el codi senzill, compileu-lo sovint i proveu-lo sovint, o us en penedireu.