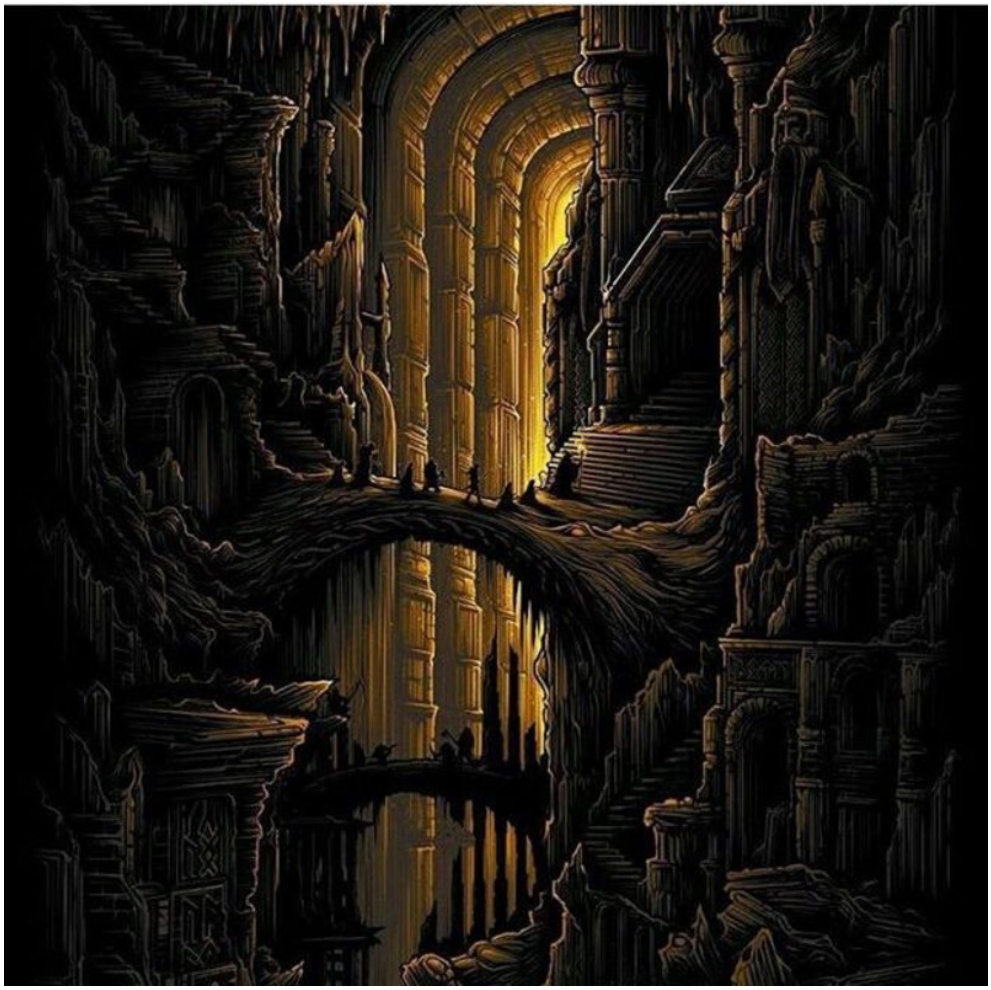
Moria

Salvador Roura

21 d'abril de 2022



1 Normes del joc

Es tracta d'un joc per a quatre jugadors, identificat amb números de 0 a 3. Cada jugador té el control d'un clan de nans ajudats per alguns mags. L'objectiu del joc és dominar l'antic regne de Mòria.

Aquest joc també inclou algunes unitats de Sauron: orcs, trolls i un Balrog. Les unitats de Sauron corresponen al jugador -1.

El joc té una durada de 200 rondes, numerades de l'1 al 200. Cada unitat de cada jugador, les unitats de Sauron incloses, pot moure's com a màxim una vegada per ronda.

El tauler té unes dimensions de 60×60. Les unitats no es poden moure fora d'ell. Les 2 files superiors, les 2 files inferiors, les 2 columnes de més a l'esquerra i les dues columnes de més a la dreta són cel·les fora de Mòria i per tant, són les úniques cel·les de tipus Exterior.

La resta de cel·les, de tipus Cova, Abisme, Granit i Roca, pertanyen a Mòria. Les cel·les de tipus Cova són transitables per qualsevol unitat. Els Abismes són cel·les només transitables per orcs i Balrog. El Granit i les cel·les de tipus Roca només són transitables per Balrog. Tot i això, les cel·les de tipus Roca són prou toves per ser excavades pels nans i convertides en Coves.

Inicialment, la majoria de cel·les de Mòria són del tipus Roca i Cova. Per defecte, 80 cel·les de tipus Cova tenen un tresor. Els nans hauran d'excavar per assolir aquests tresors. També hi ha algunes cel·les de Granit. Els Abismes només apareixen, amb una probabilitat petita, després que les cel·les de tipus Roca siguin excavades per nans.

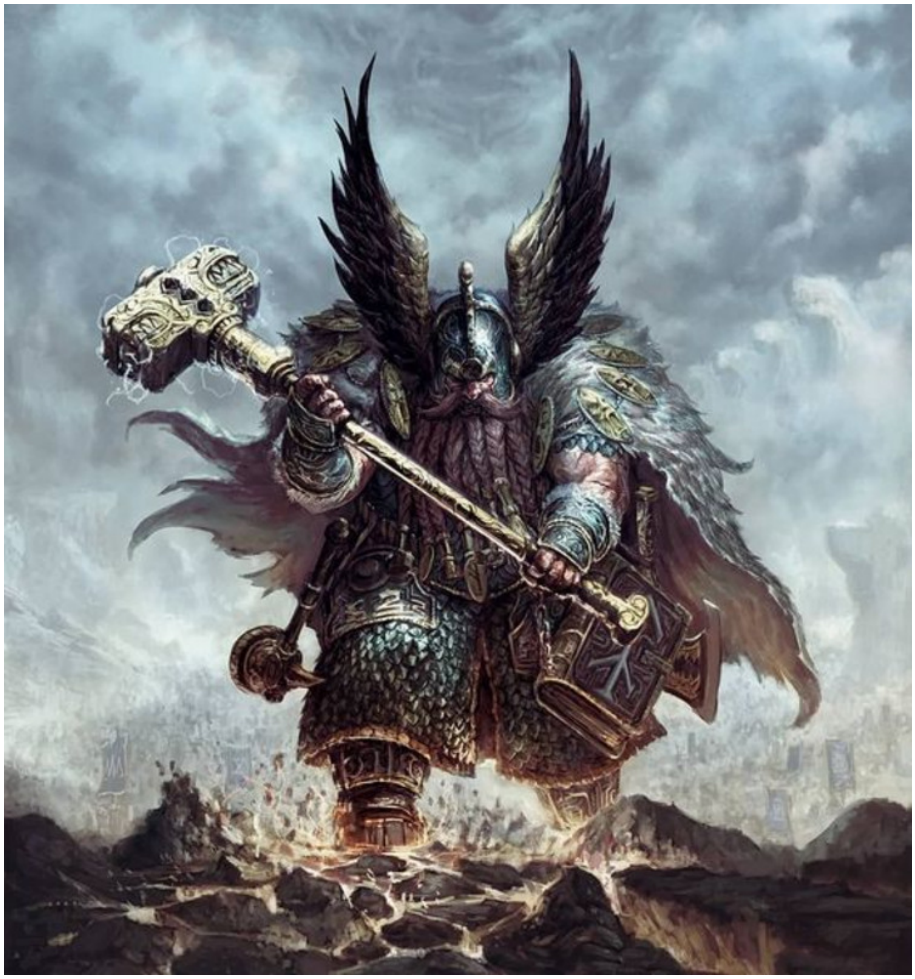
Cada cel·la pot tenir com a màxim una unitat. Quan una unitat intenta moure's a una cel·la que ja ocupava una altra unitat, això significa un atac. Una unitat que ataca no es mou durant aquesta ronda, encara que la unitat atacada sigui morta. Això inclou "atacar" una cel·la de tipus Roca (vegeu més avall).

Els nans i els mags no poden atacar unitats del mateix clan. (Això seria un moviment il·legal.) Els orcs i trolls mai ataquen altres unitats de Sauron. El Balrog no ataca: només mata tot el que l'envolta.

Els nans són els personatges principals d'aquest joc. Al principi, cada clan en té 20. Els nans neixen amb 100 punts de salut. Quan ataquen, la unitat enemiga contigua perd entre 20 i 40 punts de salut. Els nans es poden moure horitzontalment, verticalment i en diagonal a les cel·les Exteriors i les de tipus Cova.

Un nan pot excavar una cel·la de tipus Roca atacant-la. Una cel·la de tipus Roca excavada 5 vegades (per un o més nans) es converteix en un Abisme amb probabilitat 4%, i en una cel·la de tipus Cova en cas contrari.

Un Abisme és un forat sense fons no transitable per nans, mags i trolls. Qualsevol nan o mag que es mogui sobre un Abisme (que no contingui un orc) cau a l'Abisme i mor immediatament. Els trolls mai intentaran moure's cap un Abisme. Els orcs entren a Mòria només a través dels Abismes i després poden passar sobre ells. Al Balrog no li importen els Abismes d'aquest joc.



Quan un nan es mou a una cel·la de tipus Cova (que no conté cap unitat), aquesta cel·la és conquistada pel seu clan. Si la cova té un tresor, el nan l'agafa. En conseqüència, el comptador de tresors acumulats pel clan del nan augmenta en una unitat.

En qualsevol moment, si c és el nombre de cel·les actualment conquerides per un clan, i t el nombre total de tresors que ja acumula el clan, el nombre actual de punts d'aquest clan és de $c + 10t$. El clan amb més punts després de disputar-se les rondes guanya la partida.

Al començament del joc, cada clan té 5 mags. Igual que els nans, els mags poden moure's a les cel·les Exteriors i a les Coves. Tot i això, els mags d'aquest joc són febles i lents: neixen només amb 50 unitats de salut, no poden atacar a cap altra unitat, no poden conquerir cap cel·la, no poden recollir tresors, i només es poden moure horitzontalment o verticalment, no en diagonal. Però els mags tenen una propietat interessant: curen totalment qualsevol unitat aliada que ocupi les posicions horitzontals o verticals adjacents a ells al final de cada ronda.

Descrivim ara les unitats de Sauron. Primer, cal tenir en compte que les decisions preses per orcs, trolls i Balrog no depenen dels clans dels nans i mags propers. Des del punt de vista de les unitats de Sauron, tots els nans i els mags mereixen morir en la mateixa mesura.

Inicialment, el tauler no té ni Abismes ni orcs. Els Abismes poden aparèixer quan els nans excaven cel·les de tipus Roca. Després, cada Abisme "genera" un orc amb probabilitat 2% a cada ronda. Hi ha una limitació: el nombre total d'orcs del tauler mai no superarà els 20.

Els orcs neixen amb 75 punts de salut. Un atac d'un orc redueix la salut d'una unitat enemiga contigua entre 15 i 30 punts. Els orcs es poden moure horitzontalment, verticalment i en diagonal a coves i abismes. Un orc mai no surt de Mòria. Després que un orc mori, un altre orc amb el mateix identificador i 75 punts de salut pot entrar a Mòria a través d'un abisme.

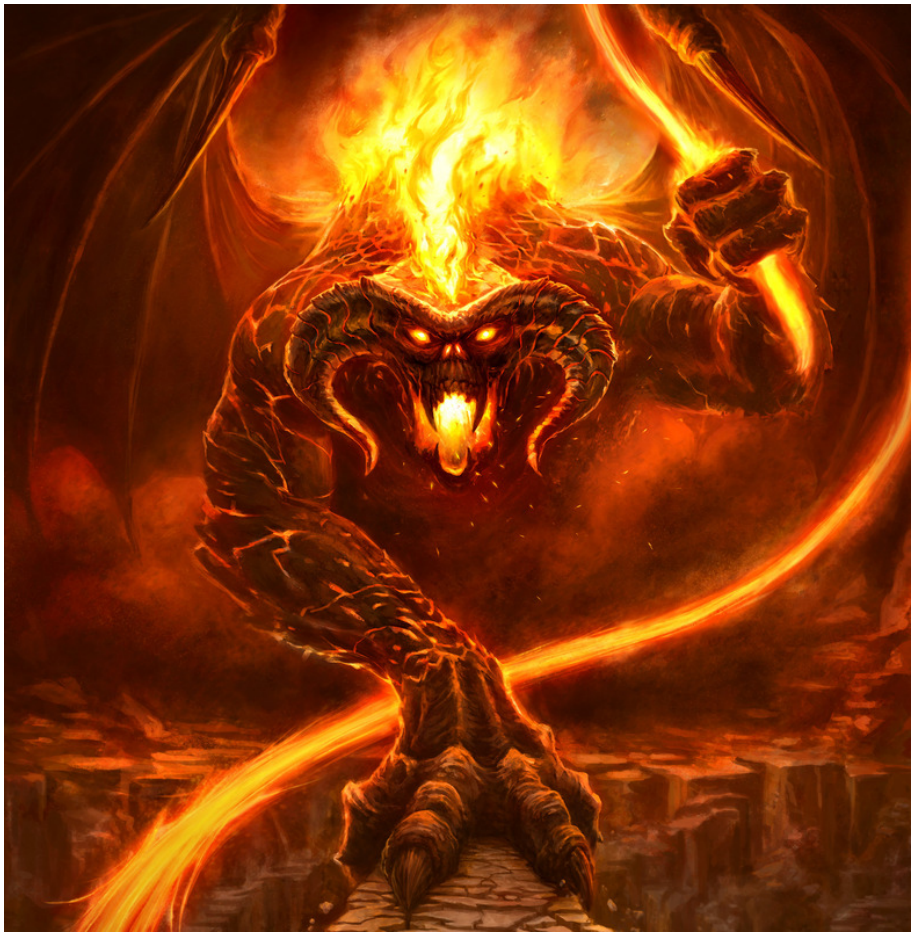
En poques paraules, un orc actua així: si té un nan o mag adjacent, l'orc l'ataca. En cas contrari, l'orc s'acosta al nan o a mag més proper dins de Mòria.

Els trolls no són especialment dolents. Són simplement estúpids, però molt forts. Els trolls neixen amb 500 punts de salut. Quan ataquen, la unitat adjacent perd entre 50 i 150 punts de salut. Els trolls es mouen horitzontalment, verticalment i en diagonal a cel·les Exteriors i de tipus Cova. Els trolls mai no abandonaran Mòria a propòsit (un cop a Mòria, s'hi queden), però reneixen fora d'ella. El joc sempre té 4 trolls.

Grosso modo, un troll es comporta així: si el troll té un nan o mag adjacent, l'ataca. En cas contrari, si el troll es troba fora de Mòria, tracta d'entrar-hi apropant-se a la cel·la de tipus Cova més propera. En cas contrari, el troll es trasllada a una cel·la de tipus Cova adjacent escollida aleatòriament.

A més del seu comportament general, els orcs i trolls sempre intenten evitar el Balrog, però no són gaire intel·ligents.

Gandalf no és aquí, de manera que Balrog és immortal. Sempre s'acosta al nan o mag més proper dins Mòria. En qualsevol moment, totes les unitats adjacents en horitzontal, vertical i en diagonal al Balrog (inclou orcs i trolls) immediatament moren. El Balrog no es pot moure en diagonal ni sortir de Mòria, però pot visitar totes les cel·les de dins.



Cada ronda es pot donar més d'una ordre a la mateixa unitat, tot i que només es seleccionarà la primera ordre (si n'hi ha). Tot programa que intenti donar més de 1000 comandes durant la mateixa ronda s'abortarà.

Cada ronda, els moviments seleccionats dels quatre jugadors s'executaran amb ordre aleatori, però respectant l'ordre relatiu entre les unitats d'un mateix clan. Per exemple, si diversos nans caminen en fila, i no hi ha altres unitats al voltant que interfereixin, tots poden avançar un pas endavant sense col·lisions entre ells, ordenant moviments des de l'inici de la fila fins al final.

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. Un altre exemple: suposem que un mag i tres nans (anomenem-los X, Y i Z) del mateix clan intenten moure's en aquest ordre a un Abisme ocupat per un orc. Primer, el mag no es mourà perquè la cel·la de destinació està ocupada. Llavors, X atacarà l'orc. Suposem que l'orc queda ferit, però no mor. Aleshores, Y també atacarà l'orc. Assumim que l'orc mor com a resultat de l'atac. Finalment, Z intentarà moure's cap a un Abisme, i morirà tragicòmicament.

Després d'haver executat tots els moviments seleccionats dels quatre jugadors, és el torn de Sauron: orcs, trolls i Balrog es mouran, per aquest ordre.

Si una unitat de Sauron té diversos moviments igualment atractius, en triarà un a l'atzar. Per exemple, si un orc o un troll té diversos nans i mags adjacents, només en triarà un d'ells a l'atzar i l'atacarà. De la mateixa manera, si el Balrog té diversos nans i mags a la mateixa distància dins de Mòria, escollirà un d'ells a l'atzar i s'hi aproparà.

Després d'executar tots els moviments d'una ronda seleccionats, els nans, els mags i els trolls morts reneixen. Per defecte, això es fa a les cel·les Exteriors, amb els seus respectius punts màxims de salut. En rars casos en què no es puguin trobar cel·les prou segures, poden renéixer a qualsevol cel·la de tipus Cova sense tresor. Si és possible, totes aquestes unitats es generen a l'inici de la partida en cel·les de tipus Cova sense tresor.

Un nan o mag mort per un altre clan serà "capturat", de manera que la unitat renascuda pertanyerà al clan atacant. Un nan o mag que ha estat mort per una unitat de Sauron o que ha caigut en un abisme s'assignarà a un clan diferent escollit a l'atzar.

Al final de cada ronda, les unitats adjacents horitzontalment o vertical als mags aliats recarregaran completament els seus punts de salut.

Com a resultat de totes les normes anteriors, el nombre total de nans, mags i trolls roman constant durant tot el joc: 80, 20 i 4, respectivament.

Si necessiteu (pseudo)-nombres aleatoris, heu d'utilitzar dos mètodes proporcionats pel joc: `random(l, u)`, que retorna un nombre enter aleatori dins `[l..u]`, i (amb menys freqüència) `random_permutation(n)`, que retorna un `vector<int>` amb una permutació aleatòria de `[0..n-1]`.

Notem que les direccions vàlides són `Bottom`, `BR`, `Right`, `RT`, `Top`, `TL`, `Left`, `LB` i `None`, corresponent als enters de 0 a 8. Aquesta definició circular pot ser usada per simplificar el vostre jugador. Vegeu el jugador `Demo` com a exemple.

2 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 jocs 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.

2.1 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 commanda com la següent:

   ```
   ./Game Demo Demo Demo Demo -s 30 -i default.cnf -o 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 i carregueu el fitxer `default.res`.

Utilitzeu

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

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

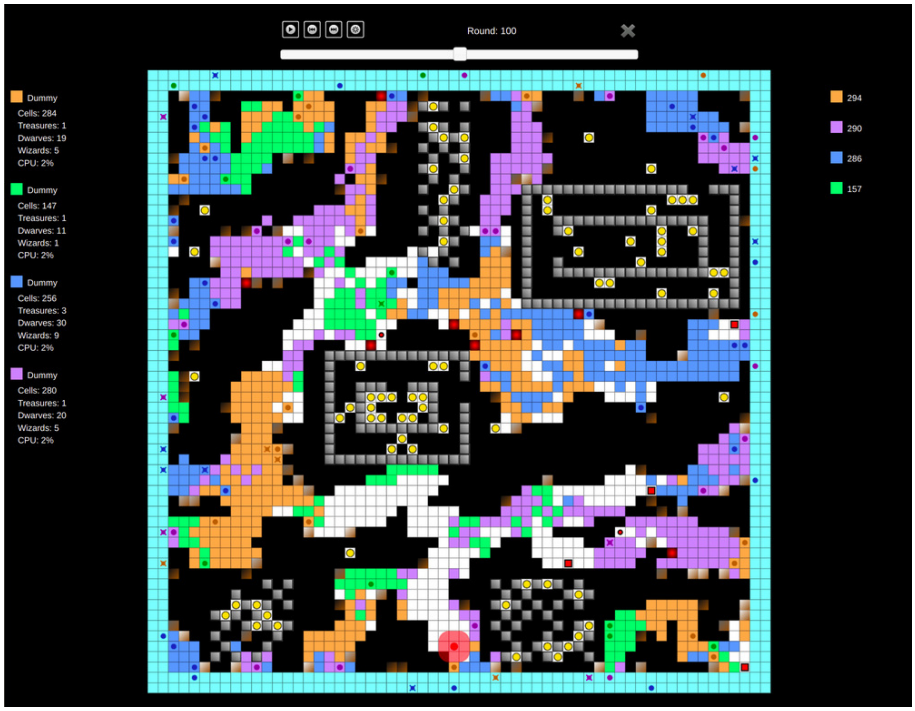
```
./Game --list
```

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.



2.2 Afegir el vostre jugador

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

#define PLAYER_NAME Null

a

#define PLAYER_NAME Gimli

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 classe del vostre jugador també podeu cridar a funcions per accedir a l'estat del joc. Aquestes funcions es posen a la vostra disposició mitjançant herència, però no ho expliqueu als professors d'enginyeria de software perquè potser no els agradarà. La documentació sobre les funcions disponibles es pot trobar en el fitxer addicional `api.pdf`.

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.

2.3 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 AIGimli.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.

3 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. No oblideu eliminar-los abans de pujar el codi. En cas de no fer-ho, el vostre jugador morirà.
- 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.
- Quan un jugador s'envia al servidor Jutge.org o durant la competició, les partides es duen a terme amb diferents llavors aleatòries. De forma

que quan us entreneu localment, executeu partides també amb diferents llavors aleatòries (amb l'opció `-s` de `Game`).

- 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.