# Bola de Drac

**boladedrac.jutge.org**

Jordi Petit  Enric Rodríguez  Salvador Roura

9 de novembre de 2012

## 1 Introducció

Hola a tothom! Aquest quadrimestre els alumnes d'EDA de la FIB i d'Algorísmia de la FME participaran en el joc **Bola de Drac**. Quina sort! Felicitats!

Les regles del joc són molt senzilles: els programes dels estudiants lluitaran els uns contra els altres durant diversos dies, a raó d'unes quantes rondes per dia. Al final de cada ronda s'eliminarà l'estudiant amb pitjors resultats. Quan més trigui un estudiant a ser eliminat, més alta serà la seva nota. Els millors 11 jugadors de la FIB i els millors 5 jugadors de la FME obtindran la nota màxima i es classificaran per a la Gran Final, en la qual es disputaran diverses rondes eliminatòries fins a trobar el campió de **Bola de Drac**.

La Gran Final se celebrarà el dimarts 18 de desembre de 14:00 a 15:00 a la Sala d'Actes de la FIB. Tothom hi serà benvingut.

Al final, només en pot quedar un!

## 2 Regles del joc

Aquest joc s'inspira en la mítica sèrie de dibuixos animats Bola de Drac (vegeu per exemple `http://ca.wikipedia.org/wiki/Bola_de_Drac`).

Cada jugador controla un personatge anomenat *goku*, el protagonista de la sèrie. Els gokus es mouen en un tauler rectangular que conté un laberint. L'objectiu d'un goku és agafar *boles de drac* i seguidament dipositar-les en alguna de les *càpsules Hoi Poi* que hi ha repartides per tauler. Cada goku només pot dur a sobre com a molt una bola de drac en cada moment. El nombre de boles de drac que hi ha al tauler és sempre el mateix, comptant tant les que duu algun goku com les que encara estan lliures. Quan una bola de drac és dipositada en alguna càpsula Hoi Poi, immediatament n'apareix una altra en una casella aleatòria del tauler.

Els gokus poden combatre entre ells, i d'aquesta manera prendre la bola de drac que duu un altre. Els combats tenen lloc quan dos gokus coincideixen en una casella, i el seu resultat depèn de la *força* de cadascun dels combatents. Una altra manera d'impedir que un altre goku dipositi la bola de drac que duu és llançant un raig *Kame Hame* que l'elimini.

A part de boles de drac i càpsules Hoi Poi, a les caselles del tauler poden haver-hi també *mongetes màgiques*, que permeten recuperar la força perduda, i *núvols Kínton*, sobre els quals els gokus es mouen el doble de ràpid i sense cansar-se.

Quan el joc acaba, és a dir, s'han realitzat totes les rondes, el jugador que ha dipositat més boles de drac guanya la partida. En cas d'empat, guanya aquell que conservi més força.

A continuació es donen les regles del joc amb més detall:

- Els següents paràmetres configuren una partida:
  - *nb_players* (): Nombre de jugadors.
  - *nb_rounds*(): Nombre de rondes de la partida.
  - *rows*(): Nombre de files del tauler.
  - *cols* (): Nombre de columnes del tauler.
  - *nb_capsules* (): Nombre de càpsules Hoi Poi.
  - *nb_balls* (): Nombre de boles de drac.
  - *nb_beans*(): Nombre de mongetes màgiques.
  - *nb_kintons* (): Nombre de núvols Kínton.
  - *max_strength*(): Força màxima dels gokus.
  - *res_strength* (): Força dels gokus després de ressuscitar.
  - *moving_penalty*(): Penalització de força per moure's.
  - *kamehame_penalty*(): Penalització de força per llançar un raig Kame Hame.
  - *combat_penalty* (): Penalització de força per combatre.
  - *goku_regen_time* (): Temps de regeneració per als gokus.
  - *bean_regen_time* (): Temps de regeneració per a les mongetes.
  - *kinton_regen_time* (): Temps de regeneració per als núvols Kínton.

  - *kinton_life_time* (): Temps de vida dels núvols Kínton.

- Els diferents tipus de casella són:

  - *Empty*: Casella buida.
  - *Rock*: Una roca.
  - *Capsule*: Una càpsula Hoi Poi.
  - *Ball*: Una bola de drac.
  - *Kinton*: Un núvol Kínton.
  - *Bean*: Una mongeta màgica.

  El tauler sempre estarà tot rodejat de roques.

- Cada casella pot rebre la visita d'un (i només un) goku. Els gokus poden trobar-se en un d'aquests estats:

  - *Normal*: Un goku normal.
  - *On_Kinton*: Un goku muntat sobre un núvol Kínton.
  - *With_Ball*: Un goku amb una bola de drac.
  - *On_Kinton_With_Ball*: Un goku muntat sobre un núvol Kínton amb una bola de drac.

  Cada goku pertany a un jugador i pot estar viu o eliminat temporalment.

- Inicialment tots els gokus es troben al màxim nivell de força *max_strength*() i els marcadors de boles dipositades estan a zero.

- A cada ronda, cada jugador pot demanar una acció, és a dir, pot triar —independentment dels altres jugadors— què vol que faci el seu goku: moure's cap a una casella adjacent seguint una direcció, o llançar un raig Kame Hame cap a una direcció (però no les dues coses alhora!). Els gokus sense núvol Kínton només poden realitzar accions en les rondes senars. Per contra, un goku muntat sobre un núvol Kínton pot moure's o llançar rajos Kame Hame en totes les rondes.

  Les direccions possibles són cap amunt, cap avall, cap a la dreta o cap a l'esquerra. Els gokus no es poden moure a les caselles on hi ha roques.

  En cas de demanar diverses accions a un goku, només es considerarà la primera. Per altra banda, les accions il·legals seran ignorades. Així doncs, pot ser que no totes les accions demanades siguin finalment concedides.

- Per tal que un goku agafi una bola de drac, només cal que passi per una casella amb bola de drac. Si un goku amb bola de drac passa per una casella amb una altra bola de drac, no passa res: el goku continua amb la seva bola i l'altra bola queda a la casella.

- Per tal que un goku amb una bola de drac la dipositi, només cal que passi per una de les *nb_capsules* () caselles amb càpsula Hoi Poi. Si un goku sense bola de drac passa per una casella de càpsula Hoi Poi, no passa res.

El nombre de boles de drac al tauler és sempre *nb_balls* (), comptant tant les que duu algun goku com les que encara estan lliures. Quan una bola de drac és dipositada, immediatament n'apareix una altra en una casella aleatòria del tauler.

- Repartides pel tauler hi ha *nb_beans* () mongetes màgiques. Quan un goku passa per una casella que conté una mongeta màgica, la consumeix i així restitueix la seva força al valor màxim inicial.

  Una vegada consumida, la mongeta màgica torna a aparèixer a la mateixa casella a partir d'un mínim de *bean_regen_time* () rondes, quan no hi ha cap goku a la casella.

- Repartits pel tauler hi ha *nb_kintons* () núvols Kínton. Quan un goku passa per una casella que conté un núvol Kínton aleshores s'hi puja. Passades *kinton_life_time* () rondes, el núvol Kínton desapareix i el goku torna a ser normal.

  Si en passar per una casella de núvol Kínton el goku ja muntava un núvol Kínton, aleshores la vida del núvol s'incrementa en *kinton_life_time* () rondes.

  Una vegada consumit, el núvol Kínton torna a aparèixer a la mateixa casella a partir de *kinton_regen_time* () rondes, quan no hi ha cap goku a la casella.

- En una casella no pot haver-hi dos gokus alhora. Quan un goku es mou a una casella on ja hi ha un altre goku, ja sigui perquè aquest segon encara no s'ha mogut o bé perquè s'hi acaba de moure, té lloc un combat. En un combat entre els gokus $A$ i $B$, la probabilitat que $A$ en sigui el vencedor és $(1 + \mathit{força}(A))/(2 + \mathit{força}(A) + \mathit{força}(B))$. El vencedor del combat es queda a la casella i la seva força després de lluitar és $\max(0,\ \mathit{força} - \mathit{combat\_penalty}())$. El perdedor queda eliminat i sense força.

  Si el goku vençut o la casella tenen bola de drac, aquesta passa a mans del vencedor. Les boles de drac sobrants (per exemple, si el vencedor ja té bola de drac) reapareixen en una posició aleatòria del tauler.

  En cas que el goku vençut estigui muntat sobre un núvol Kínton, aquest passa a mans del vencedor. Si el vencedor ja està muntat sobre un núvol Kínton, aleshores a aquest se li afegeix la vida del núvol del vençut, que desapareix.

- Un raig Kame Hame elimina tot goku que hi hagi en línia recta des de la posició on hi ha el goku llançador fins a la propera casella de roca, en la direcció indicada. La força dels gokus eliminats queda a 0. Les seves boles de drac reapareixen aleatòriament en una altra casella del tauler. Els seus núvols Kínton desapareixen.

- Les accions que realitzen els gokus poden consumir força. Aquest desgast depèn del goku i de l'acció:

- Llançar un raig Kame Hame consumeix *kamehame_penalty*() unitats de força. Si un goku no disposa almenys d'aquesta quantitat de força, no pot llançar un raig Kame Hame.
- Si un goku està muntat sobre un núvol Kínton, no perd força quan es mou. Si no té núvol Kínton:
  * quan la ronda és congruent amb 3 mòdul 4, el goku només es pot moure si té almenys *moving_penalty*() unitats de força, i perd aquesta quantitat de força després del moviment;
  * quan la ronda no és congruent amb 3 mòdul 4, no hi ha penalització per moviment. Això implica que un goku sense força i sense núvol Kínton només es pot moure en una de cada quatre rondes!

El desgast de força és independent de si el goku porta bola de drac o no.

- Les accions demanades pels jugadors a cada ronda s'executen de la forma següent. Inicialment es determina un ordre aleatori dels jugadors. Aleshores, seguint aquest ordre, s'executen primer els rajos Kame Hame, i després es realitzen els moviments (i els combats que en resultin).

  Més concretament, el llançament de Kame Hames funciona de la forma següent. A cada ronda només es pot llançar un sol Kame Hame. Seguint l'ordre dels jugadors, suposem que el primer jugador que ha demanat un Kame Hame és $A$. Si $\mathit{força}(A) < \mathit{kamehame\_penalty}()$, o és ronda parell i $A$ no té núvol Kínton, aleshores s'ignora la petició de $A$, tal com s'ha explicat anteriorment. Altrament $A$ té una probabilitat

  $$\frac{1 + \mathit{força}(A) - \mathit{kamehame\_penalty}()}{1 + \mathit{max\_strength}() - \mathit{kamehame\_penalty}()}$$

  de que se li concedeixi el llançament. Es repeteix el procés per cada jugador seguint l'ordre, fins que es concedeixi un Kame Hame o no hi hagi més jugadors. Una vegada s'ha concedit un Kame Hame, les peticions de Kame Hame dels jugadors següents són ignorades.

- Al final de cada ronda, s'actualitzen els comptadors de regeneració i de vida. Això permet que els núvols Kínton i les mongetes màgiques tornin a aparèixer a les seves caselles, i que els núvols Kínton que han esgotat la seva vida desapareguin.

  Els gokus eliminats també poden ressuscitar. Després de com a mínim *goku_regen_time*() rondes, un goku eliminat torna a aparèixer en estat normal en una casella aleatòria amb el nivell *res_strength* () de força . Els gokus sempre ressusciten en posicions buides i sense altres gokus al voltant.

- Si un jugador comet un error d'execució o esgota el seu temps de càlcul (3 segons per toda la partida), el seu goku queda congelat, no juga més.

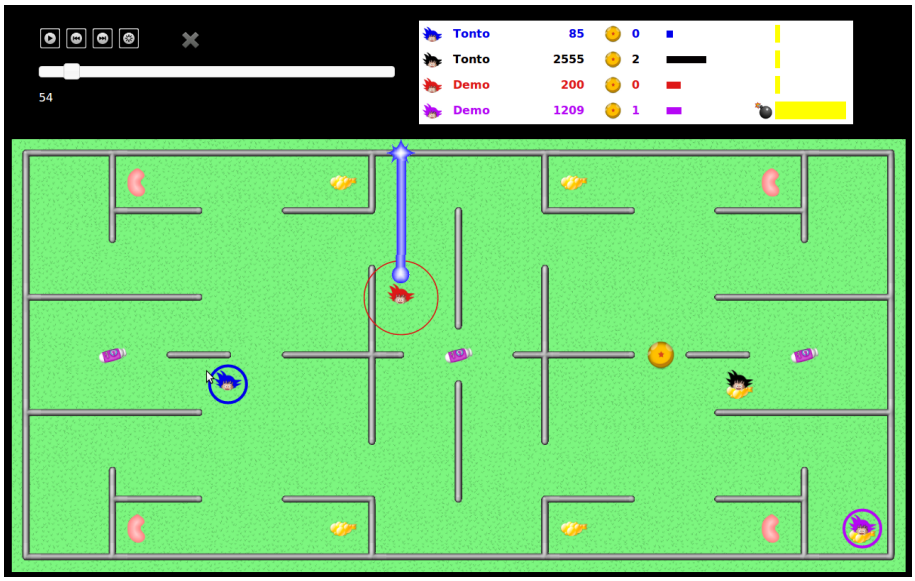
- En acabar totes les rondes de la partida, el rànking dels jugadors es calcula segons la seva puntuació, que segueix la fórmula:

$$\text{puntuació} = \text{boles} \cdot (\mathit{max\_strength}() + 1) + \text{força}.$$

  D'aquesta manera, guanya qui més boles de drac hagi dipositat, i en cas d'emptat, el que tingui més força. El guanyador pot tenir el goku congelat, això està permès.

# 3 El Visor

A continuació es mostra una captura de pantalla del visor on apareixen pràcticament tots els elements del joc:




- A la part superior esquerra apareixen botons que permeten aturar, anar al principi i al final de la partida, desactivar el mode animació, o tancar el visor. Una barra horitzontal indica visualment en quin punt de la partida es troba la ronda actual. El nombre a sota mostra exactament la ronda actual.

- A la part superior dreta apareix el marcador. Cada jugador apareix amb el seu nom i goku en el color que li correspon. Quan un jugador és eliminat, el seu goku desapareix del marcador i va reapareixent gradualment fins que en el moment de ressuscitar recupera la seva mida normal. El marcador indica, per cada jugador, la seva puntuació, i mostra també el

nombre de boles de drac capturades. La força de cada goku es representa mitjançant una barra horitzontal. Finalment, una barra de color groc mostra el temps de CPU consumit. Quan s'excedeix el temps límit, apareix la icona d'una bomba i el jugador queda congelat.

- Els gokus envoltats per un cercle petit i gruixut porten una bola de drac. A la captura de pantalla, és el cas dels gokus lila (que està muntat sobre un núvol Kínton) i blau (que no té núvol Kínton).
- Els gokus envoltats per un cercle gran i fi acaben de ressuscitar. A la captura de pantalla, és el cas del goku vermell.
- A la captura de pantalla, el goku vermell està llançant un raig Kame Hame.
- Els altres elements que apareixen a la captura de pantalla són els següents:


Bola de drac


Núvol Kínton


Mongeta màgica


Càpsula Hoi Poi

# 4 Programació

En el web `https://boladedrac.jutge.org` trobareu el material necessari per programar el joc en C++. En concret, hi teniu tots els fitxers de suport, exemples de jugadors, i un visor de partides amb els quals podreu desenvolupar i provar els vostres jugadors.

Necessitareu `g++`, `make` i un navegador recent (Chrome, Firefox). El codi del joc és portable a qualsevol sistema Linux, Mac o Windows, suposant que hi instal·leu les eines adequades. Els ordinadors de la FIB i de la FME ja tenen aquest software instal·lat. Si voleu treballar amb altres màquines:

- Amb Debian o Ubuntu, amb una instrucció del tipus

  ```
  sudo apt-get install build-essential chromium-browser
  ```

  tindreu tot el que us cal.
- Amb Mac, en tindreu prou amb instal·lar XCode des de l'App Store.
- Amb Windows, us bastarà instal·lar per exemple MinGW (`http://mingw.org`).

## 4.1 Com fer un jugador

Per fer un jugador, copieu primer el fitxer `Null.cc` a un fitxer `XXX.cc`, on `XXX` és un identificador de la vostra elecció. Trieu un identificador no ofensiu, que no hagi estat triat ja per un altre estudiant, i compost per, com a molt, 12 lletres, dígits i caràcters de subratllat; per exemple, `SonGoku`.

A continuació, al fitxer `SonGoku.cc` (o com l'hagueu anomenat) que acabeu de crear, heu de canviar la línia 11 per posar-hi el nom del vostre jugador (a l'exemple, **#define** PLAYER_NAME SonGoku.

Finalment, heu d'implementar el vostre jugador tot completant la classe PLAYER_NAME que hereda les operacions de consulta del tauler (classe *Board*) i de creació d'accions (classe *Action*) a través de la classe base *Player*. El mètode *play* () es crida a cada ronda per transmetre-us l'estat actual del tauler i recollir les accions del vostre jugador. A la vostra classe hi podeu afegir camps (variables) per recordar l'estat d'una ronda a l'altra, mètodes (o funcions) per descompondre el vostre programa, etc.

Fixeu-vos que la vostra classe ha de contenir una funció anomenada *factory* () que no heu de modificar, i que després de la classe també hi ha una crida per registrar el vostre jugador que tampoc heu de modificar.

Podeu prendre com a referència el jugador `Demo.cc` que s'adjunta amb el material de la pràctica.

## 4.2 Com executar i veure partides localment

1. Copieu el fitxer `Tonto-SISTEMA.o` a `Tonto.o`, segons el sistema que tingueu (Mac, Linux/Windows de 32/64 bits).

2. Editeu el fitxer `Makefile`, i al lloc indicat de la capçalera hi afegiu els jugadors que voleu incorporar amb l'extensió `.o`.

3. Executeu `make` per compilar tots els fitxers que calguin i crear l'executable `BolaDeDrac`.

4. Per disputar una partida amb els paràmetres de `demo.cnf`, executeu una comanda com ara

   ```
   ./BolaDeDrac Tonto SonGoku Demo Demo -i demo.cnf -o game.bdd
   ```

   Aquí, assumint que a `demo.cnf` el nombre de jugadors establert sigui 4, el primer jugador serà `Tonto`, el segon `SonGoku`, i els altres dos `Demo`. El resultat de la partida quedarà a `game.bdd`.

5. Visualitzeu la partida executant `./viewer.sh game.bdd` (en Linux/MAC) o `./viewer.bat game.bdd` (en Windows). L'script buscarà un navegador i l'obrirà per mostrar la partida.

Podeu obtenir la llista completa de paràmetres del programa `BolaDeDrac` fent `./BolaDeDrac --help`. En particular, `./BolaDeDrac --list` us llistarà els noms dels jugadors inclosos.

Si us cal, podeu netejar el vostre directori de fitxers executables i fitxers objectes amb la comanda `make clean`.

### 4.3 Restriccions

Els codis del jugadors que volgueu enviar al web del joc han de complir certes limitacions:

- Tot el codi del vostre jugador s'ha de trobar en un sol fitxer i seguir les convencions donades.
- No podeu fer servir variables globals (utilitzeu camps).
- Només podeu usar les llibreries estàndard de C++, com ara `vector`, `map`, etc.
- No podeu obrir fitxers, ni fer altres crides al sistema operatiu (threads, forks, etc.).
- Si us cal, podeu utilitzar **cout** i **cerr**, però no **cin**. Compte, escriure missatges consumeix temps!
- En execució local, el sistema no controla jugadors que abortin, que triguin massa, o que interfereixin amb els contraris. En l'execució al servidor, sí.

### 4.4 Estructures de dades

Per saber com consultar el tauler, feu un cop d'ull a les operacions públiques de la class *Board* del fitxer *Board.hh*. Per saber com sol·licitar les accions, mireu les operacions públiques de la class *Action* del fitxer *Action.hh*. Fixeu-vos que només podeu utilitzar les operacions públiques d'aquestes classes.

## 5 El web del joc

El web de Bola de Drac es troba a `https://boladedrac.jutge.org` i és on podreu trobar aquestes instruccions i el codi necessari per començar a jugar. Haureu de fer servir aquest web per fer els lliuraments. També el podreu usar per jugar partides on-line contra els programes d'altres estudiants. I tothom podrà veure els estudiants eliminats en temps real.

És normal que el vostre navegador es queixi en accedir al web en mode de connexió segura perquè el certificat no està comprovat per un tercer.

## 5.1 Lliuraments

Podeu fer més d'un programa amb estratègies diferents. Sempre que es vulgui es podrà pujar al web versions actualitzades dels fitxers amb els vostres jugadors, per exemple `SonGoku.cc`. De forma automàtica, es farà el procés de verificació següent:

- Es compilarà el jugador enviat. Si no compila, l'enviament es rebutjarà.
- Es jugarà una partida amb quatre còpies del jugador enviat. Si algun dels programes aborta o triga massa temps a respondre, l'enviament es rebutjarà.
- Es jugaran quatre partides contra tres programes tontos (un programa tonto és un jugador programat pels organitzadors amb una estratègia senzilla). El programa enviat serà rebutjat si no guanya les quatre partides.

En el cas que un programa sigui rebutjat, se n'indicarà el motiu, i es mostrarà la raó o la partida causant del rebuig. En cas contrari, el programa s'acceptarà al web del joc, podrà disputar partides on-line, i el seu nom quedarà reservat: ningú podrà tornar a enviar un altre programa amb el mateix nom. En qualsevol cas, els vostres codis no seran mai mostrats als altres estudiants.

## 5.2 Partides d'entrenament

El web del joc permet celebrar partides on-line entre els programes acceptats. Aquestes partides es consideren "d'entrenament" perquè els seus resultats, que només són visibles als autors dels programes, no afecten la classificació.

Per fer-ho, es té el concepte de participants *amics*, i es distingeix entre programes d'ús *privat* i d'ús *públic*. Per defecte, els programes enviats són d'ús privat: només l'autor i els seus amics poden jugar partides on-line amb aquells programes. En concret, un programa privat no pot participar en una partida si el seu autor no és amic de tots els autors dels programes de la partida.

Es pot evitar aquesta restricció fent que un jugador sigui d'ús públic. Fent això, es podrà enfrontar contra tots els altres jugadors d'ús públic. Quan un jugador es fa d'ús públic, no es permet tornar-lo a fer d'ús privat.

Mitjançant el web es pot modificar l'ús dels jugadors, demanar que es juguin partides on-line, i administrar la llista dels amics. Perquè dos participants es considerin amics és necessari que ambdós s'incloguin mútuament a les seves llistes.

## 5.3 Partides oficials

Quan comencin les eliminatòries, el web anirà organitzant rondes amb els programes representants dels estudiants supervivents. Els resultats d'aquestes partides i les pròpies partides seran visibles per a tothom.

A cada ronda, s'eliminarà l'autor del programa amb pitjors resultats, seguint el procés següent: s'agrupen els programes de quatre en quatre a l'atzar, afegint programes tontos de reserva si convé per arribar a un múltiple de 4. Per a cada grup de 4, es juga una partida. En cada partida, se salva qualsevol programa que obtingui millor puntuació que dos dels seus tres rivals a la partida. Es repeteix el procés (agrupar de 4 en 4, afegir programes tontos, etc.) amb els programes no salvats, fins que únicament quedin 4 o menys programes. Aleshores, es disputen partides entre aquests programes (afegint de nou, si convé, programes tontos de reserva) fins que un d'ells quedi per darrere dels seus rivals, sense comptar els de reserva. Aleshores, l'autor del programa perdedor queda eliminat.

Cada estudiant podrà triar quin dels seus programes acceptats l'ha de representar, i podrà canviar aquesta decisió tantes vegades com vulgui, en particular si envia versions millorades dels seus programes. Tanmateix, mentre es disputi una ronda no es podrà canviar el programa: el canvi es farà efectiu en començar la següent ronda (si no ha quedat eliminat). Tampoc no es podrà canviar de programa una vegada arribats a la Gran Final.

### 5.4 Novetats i incidències

Qualsevol novetat o incidència que els organitzadors del joc vulguin donar a conèixer es publicarà en el web del joc.

### 5.5 Resultats

El web del joc mostrarà a tothom i en temps real els resultats de cada ronda.

### 5.6 Fòrum

El fòrum del joc per als estudiants matriculats d'EDA es troba al Racó.

## 6 Dates importants

- El joc es fa públic el dimarts 6 de novembre.
- El web del joc rebrà partides a partir del dimarts 6 de novembre a les 11:00.
- Per participar a les rondes eliminatòries, caldrà tenir un programa acceptat abans del divendres 23 de novembre a les 15:00.
- Les eliminatòries seran des del dilluns 26 de novembre fins al divendres 14 de desembre.
- La Gran Final se celebrarà el dimarts 18 de desembre de 14:00 a 15:00 a la Sala d'Actes de la FIB.

# 7 Normativa

El mètode d'avaluació del joc es troba publicat a la Guia Docent d'EDA/Algorísmia. El lliurament de programes per al joc implica acceptar que, si es comprova que hi ha hagut frau en la seva realització, es restaran els punts obtinguts (enlloc de sumar-los) als de l'assignatura.

Els programes enviats es compararan periòdicament entre ells i també amb els enviats en cursos anteriors per detectar possibles còpies.

Davant de qualsevol imprevist, els professors prendran les mesures que considerin més oportunes.

# 8 Consells

Us recomanem seguir els consells següents:

- Estudieu la part pública de les classes lliurades. No us preocupeu per les parts privades ni la implementació.
- Sembla que el visor funciona millor amb Chrome que amb Firefox. Però compte: per raons de seguretat, per veure partides locals cal obrir Chrome amb la opció `--allow-file-access-from-files`, que permet llegir fitxers del vostre disc.
- Comenceu amb estratègies molt senzilles, que siguin fàcils de programar i de depurar, que és exactament allò que necessiteu al principi.
- Programeu procediments senzills i útils, i *assegureu-vos que funcionin correctament*.
- No us arrisqueu a ser eliminats abans de començar. Aconseguiu que el web us accepti un jugador tan ràpidament com us sigui possible. Un jugador acceptat representa tenir assegurada part de la nota de la pràctica!
- Conserveu diverses versions antigues (però que funcionin correctament) dels vostres jugadors, i no les toqueu per res.
- Compileu sovint, testejeu sovint. És *molt* més fàcil trobar i corregir els errors quan s'han canviat unes poques línies de codi que quan s'acumulen molts canvis de cop.
- Feu servir `cout`'s o `cerr`'s per posar xivatos, i `assert`'s per comprovar que el codi fa el que toca. Però millor que comenteu el xivatos abans d'enviar el jugador al web, ja que alenteixen els programes.
- Per testejar un jugador nou, enfronteu-lo contra uns altres jugadors que no treguin cap missatge. Així només apareixeran per pantalla els missatges del nou jugador.

- Un error de segmentació en el vostre codi aturarà el simulador, i no podreu veure què ha provocat l'error del jugador. Apreneu a usar el `valgrind` si no sou capaços de corregir els errors amb `assert`'s i xivatos.
- Poseu les opcions de *debug* al Makefile per ajudar-vos a trobar possibles errors.
- Feu competir els vostres jugadors contra rivals diferents, i estudieu les partides. Encara que no podreu veure els codis dels altres estudiants, si sou capaços de deduir les seves tàctiques i us semblen útils, podeu mirar d'imitar-les, defensar-vos-en o millorar-les.
- No deixeu el vostre codi a ningú, ni tan sols d'una versió antiga. Un cop feta la competició s'analitzaran els programes de cada ronda amb programes detectors de plagi (JPlag, per exemple). També es comparan els jugadors d'aquest curs amb els jugadors de cursos anteriors.
- Per evitar disgustos, feu servir el web del joc per jugar partides amb els vostres amics. Però no confieu massa en els vostres amics...
- Tingueu en compte que podeu lliurar nous jugadors en qualsevol moment (excepte durant la fase final).
- No espereu fins al darrer moment per enviar els vostres programes. Si tothom ho fa, la cua del Jutge no dóna l'abast.
- Insistim: Feu codis senzills. Compileu sovint, testejeu sovint. O afronteu-ne les conseqüències.