Ants

Enric Rodríguez

30 d'abril de 2019

1 Regles del Joc

En aquest joc, cada jugador controla una colònia de formigues. A cada ronda d'una partida, els jugadors acumulen tants punts com el nombre de formigues que hi ha a la seva colònia al final de la ronda. El guanyador del joc és el jugador que, al final de la partida, té la màxima puntuació.

El joc es juga en un tauler rectangular de mida $BOARD\_ROWS \times BOARD\_COLS$. Les cel·les del tauler poden ser d'*aigua* o de *sòl*. Cada formiga ocupa una sola cel·la i no pot haver-hi dues formigues a la mateixa cel·la al mateix temps. Les formigues tenen por de l'aigua, i per tant només poden ocupar cel·les de sòl.

A les colònies hi ha diverses categories de formigues: *reines*, *soldats* i *treballadors*. Les formigues tenen una vida limitada, que es decrementa en una unitat després de cada ronda. Quan s'esgota la vida d'una formiga, la formiga mor. Inicialment, la vida d'una reina és de *QUEEN_LIFE* rondes, la vida d'un soldat és de *SOLDIER_LIFE* rondes, i la vida d'un treballador és de *WORKER_LIFE* rondes.

Cada colònia té una reina. Si la reina d'una colònia mor, llavors una altra formiga de la colònia (soldat o treballador), si n'hi ha, és escollida a l'atzar i es converteix en la nova reina. La vida d'aquesta nova reina és de *QUEEN_LIFE* rondes.

Una partida consisteix en *NUM_ROUNDS* rondes. En una ronda, les formigues es poden moure a una de les cel·les adjacents de sòl dins del tauler, en horitzontal o vertical (però no en diagonal). Les reines només poden moure's en rondes que són múltiples de *QUEEN_PERIOD*. Per exemple, si *QUEEN_PERIOD* és 2, les reines només es poden moure en les rondes 0, 2, 4, 6, etc. Els soldats i els treballadors es poden moure en qualsevol ronda.

Quan una formiga intenta moure's cap a una cel·la que ja ocupa una altra formiga (sigui de la mateixa colònia o no), la primera ataca a la última. El resultat de la lluita es determina de la manera següent:

- Si les dues formigues són de la mateixa categoria, totes dues moren.

- Les reines maten soldats i treballadors.
- Els soldats maten els treballadors.

Si la formiga atacant sobreviu, es realitza el moviment.

Independentment de la presència d'una formiga, algunes cel·les de sòl també poden contenir un bonus de menjar de diferents tipus: *pa*, *llavors* o *fulles*. Quan una reina es desplaça cap a una cel·la amb menjar, es menja el bonus. D'altra banda, quan un treballador es mou a una cel·la amb menjar, pot collir el menjar, carregar-lo durant algun temps i deixar-lo a una cel·la més enllà. Els bonus només es poden deixar a la posició actual del treballador. El treballador només pot portar un bonus a la vegada i només pot deixar-lo en cel·les que no contenen cap bonus. Quan un treballador que porta els aliments mor (perquè la seva vida està esgotada o com a resultat d'una lluita), el bonus es destrueix. Els soldats no poden menjar ni carregar bonus.

Una reina pot posar un ou (d'un soldat o d'un treballador) en una cel·la de sòl adjacent (horitzontalment o verticalment). Al final de la ronda l'ou esclata i una nova formiga de la colònia neix. Si el nounat és un soldat, la seva vida és de *SOLDIER_LIFE* rondes, i si és un treballador, de *WORKER_LIFE* rondes. No obstant això, si en el moment en què l'ou esclata la cel·la ja està ocupada per una altra formiga (sigui de la mateixa colònia o no), la nova formiga mor automàticament.

Una reina necessita tres tipus de nutrients a la seva reserva per produir ous: *carbohidrats*, *proteïnes* i *lípids*. Depenent de si la reina decideix posar un ou d'un soldat o d'un treballador, la quantitat necessària d'aquests nutrients és diferent. Les constants que defineixen aquestes quantitats es resumeixen a la taula següent:

	Soldat	**Treballador**
Carbohidrats	*SOLDIER_CARBO*	*WORKER_CARBO*
Proteïnes	*SOLDIER_PROTE*	*WORKER_PROTE*
Lípids	*SOLDIER_LIPID*	*WORKER_LIPID*

Quan una reina menja, els nutrients del menjar s'acumulen a la seva reserva. Els nutrients de cada tipus d'aliment són determinats per les constants de la taula següent:

	Pa	**Llavor**	**Fulla**
Carbohidrats	*BREAD_CARBO*	*SEED_CARBO*	*LEAF_CARBO*
Proteïnes	*BREAD_PROTE*	*SEED_PROTE*	*LEAF_PROTE*
Lípids	*BREAD_LIPID*	*SEED_LIPID*	*LEAF_LIPID*

A l'inici d'una partida, la colònia de cada jugador està formada per la seva reina, *NUM_INI_SOLDIERS* soldats i *NUM_INI_WORKERS* treballadors, situats en cel·les de sòl en una de les quatre cantonades del tauler (el jugador 0 a

la cantonada superior esquerra, el jugador 1 a la cantonada superior dreta, el jugador 2 a la cantonada inferior dreta, i el jugador 3 a la cantonada inferior esquerra). Inicialment la reina no té nutrients per posar ous, és a dir, la seva reserva està buida. De la mateixa manera, quan un soldat o un treballador es converteix en reina, no té nutrients. Hi ha una excepció, però: si la formiga era un treballador que portava menjar, es consumeix el bonus i els seus nutrients s'acumulen a la reserva de la nova reina; el mateix passa si la cel·la que ocupa conté un bonus. Si es produeixen les dues situacions, es consumeixen els dos bonus.

Per a cadascun dels quatre quadrants del tauler hi ha un rectangle de *BONUS_ROWS* $\times$ *BONUS_COLS* cel·les incloses al quadrant on poden aparèixer bonus de pa. També hi ha rectangles anàlegs per a llavors i fulles. Per a cadascun d'aquests rectangles, cada *BONUS_PERIOD* rondes (començant per la ronda 0) una cel·la de sòl del rectangle sense bonus i sense formigues es tria de forma aleatòria i hi apareix un nou bonus. Si no hi ha cap cel·la així, no es crea cap nou bonus. La ubicació exacta d'aquests rectangles es determina aleatòriament i és desconeguda pels jugadors.

Cada formiga té un nombre natural que l'identifica de manera única. A cada ronda, els jugadors utilitzen aquests identificadors per manar a les seves formigues accions, com a màxim una per a cada formiga:

- moure's a una cel·la adjacent de sòl;
- collir menjar;
- deixar menjar;
- posar un ou de soldat o de treballador.

Només se selecciona la primera acció ordenada a una formiga. La resta d'ordres per a aquesta formiga s'ignoren. A més, qualsevol programa d'un jugador que intenti donar més de 1000 ordres a la mateixa ronda s'avorta.

Al final de cada ronda, totes les ordres seleccionades s'ordenen aleatòriament i s'executen seguint aquest ordre. Si no es pot executar una ordre (p. ex., la formiga comandada ha mort com a conseqüència de l'execució de comandes anteriors), l'ordre s'omet. Després d'executar totes les ordres, els comptadors de vida es decrementen i moren aquelles formigues la vida de les quals s'ha esgotat. A continuació, els ous posats per les reines esclaten (fins i tot si la reina que va posar l'ou ha mort, per exemple, a causa d'una baralla), també en ordre aleatori. Llavors, per a cada colònia que ja no té reina, s'escull una nova reina a l'atzar entre els soldats i els treballadors restants, si n'hi ha cap, tal com s'explica anteriorment. A continuació, els bonus dels aliments (pa, llavors i fulles) es generen si el nombre actual de rondes és un múltiple de *BONUS_PERIOD*. Finalment, per a cada jugador, el nombre de formigues de la colònia s'afegeix a la puntuació.

1.1 Paràmetres del Joc

Un joc queda definit per un tauler i pel conjunt de paràmetres de la Figura 1.

Llevat que hi hagi un cas de força major, els valors indicats dels paràmetres són els que s'utilitzaran en totes les partides del joc.

2 El Visor

A la Figura 2 es mostra una captura de pantalla amb tots els elements del joc.

- A la cantonada superior esquerra 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. A la cantonada superior dreta hi ha un botó per tancar el visor. La ronda actual també es mostra a la part superior del 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 reserva de la reina (si un jugador no té reina, tots els valors són 0). A les partides jugades a Jutge.org, també es mostra el percentatge de temps de CPU que s'ha consumit fins ara (si està esgotat, s'indica amb un 'out').
- A la columna de la dreta, es mostra la puntuació de cada jugador en ordre, sent la puntuació més alta la que hi ha a la part superior.
- Les cel·les de sòl estan pintades de color marró clar, mentre que les cel·les d'aigua es pinten de color blau cel.
- Les reines, soldats i treballadors es representen com a la Figura 3.
- Els bonus es representen com a la Figura 4.
- Els treballadors que porten un bonus tenen un cercle al seu voltant, de diferent color segons el tipus de bonus: taronja pel pa, verd per les fulles, i gris per les llavors. A la captura de pantalla, hi ha quatre treballadors portant pa, un portant fulles, i encara un altre portant llavors.

3 Programar el joc

El primer que heu de fer és descarregar el codi font. Inclou un programa de C++ que executa les partides i un visor HTML per veure-les en un format ani-

Paràmetre	Valor	Descripció
NUM_PLAYERS	4	Nombre de jugadors del joc.
NUM_ROUNDS	250	Nombre de rondes que dura la partida.
BOARD_ROWS	25	Nombre de files del tauler.
BOARD_COLS	25	Nombre de columnes del tauler.
QUEEN_PERIOD	2	Les reines es poden moure cada *QUEEN_PERIOD* rondes, començant a la ronda 0.
SOLDIER_CARBO	3	Unitats de carbohidrats necessàries per a un ou d'un soldat.
SOLDIER_PROTE	3	Unitats de proteïnes necessàries per a un ou d'un soldat.
SOLDIER_LIPID	3	Unitats de lípids necessàries per a un ou d'un soldat.
WORKER_CARBO	1	Unitats de carbohidrats necessàries per a un ou d'un treballador.
WORKER_PROTE	1	Unitats de proteïnes necessàries per a un ou d'un treballador.
WORKER_LIPID	1	Unitats de lípids necessàries a un ou d'un treballador.
BREAD_CARBO	2	Unitats de carbohidrats contingudes en una bonus de pa.
BREAD_PROTE	0	Unitats de proteïnes contingudes en una bonus de pa.
BREAD_LIPID	1	Unitats de lípids contingudes en una bonus de pa.
SEED_CARBO	0	Unitats de carbohidrats contingudes en una bonus de llavor.
SEED_PROTE	1	Unitats de proteïnes contingudes en una bonus de llavor.
SEED_LIPID	2	Unitats de lípids contingudes en una bonus de llavor.
LEAF_CARBO	1	Unitats de carbohidrats contingudes en una bonus de fulla.
LEAF_PROTE	2	Unitats de proteïnes contingudes en una bonus de fulla.
LEAF_LIPID	0	Unitats de lípids contingudes en una bonus de fulla.
NUM_INI_SOLDIERS	3	Nombre de soldats inicials.
NUM_INI_WORKERS	11	Nombre de treballadors inicials.
BONUS_ROWS	3	Nombre de files d'un rectangle on apareix menjar.
BONUS_COLS	3	Nombre de columnes d'un rectangle on apareix menjar.
BONUS_PERIOD	25	Els bonus de menjar apareixen cada *BONUS_PERIOD* rondes, començant a la ronda 0.
WORKER_LIFE	75	Rondes després de les quals un treballador mor.
SOLDIER_LIFE	150	Rondes després de les quals un soldat mor.
QUEEN_LIFE	300	Rondes després de les quals una reina mor.

Figura 1: Paràmetres del joc.

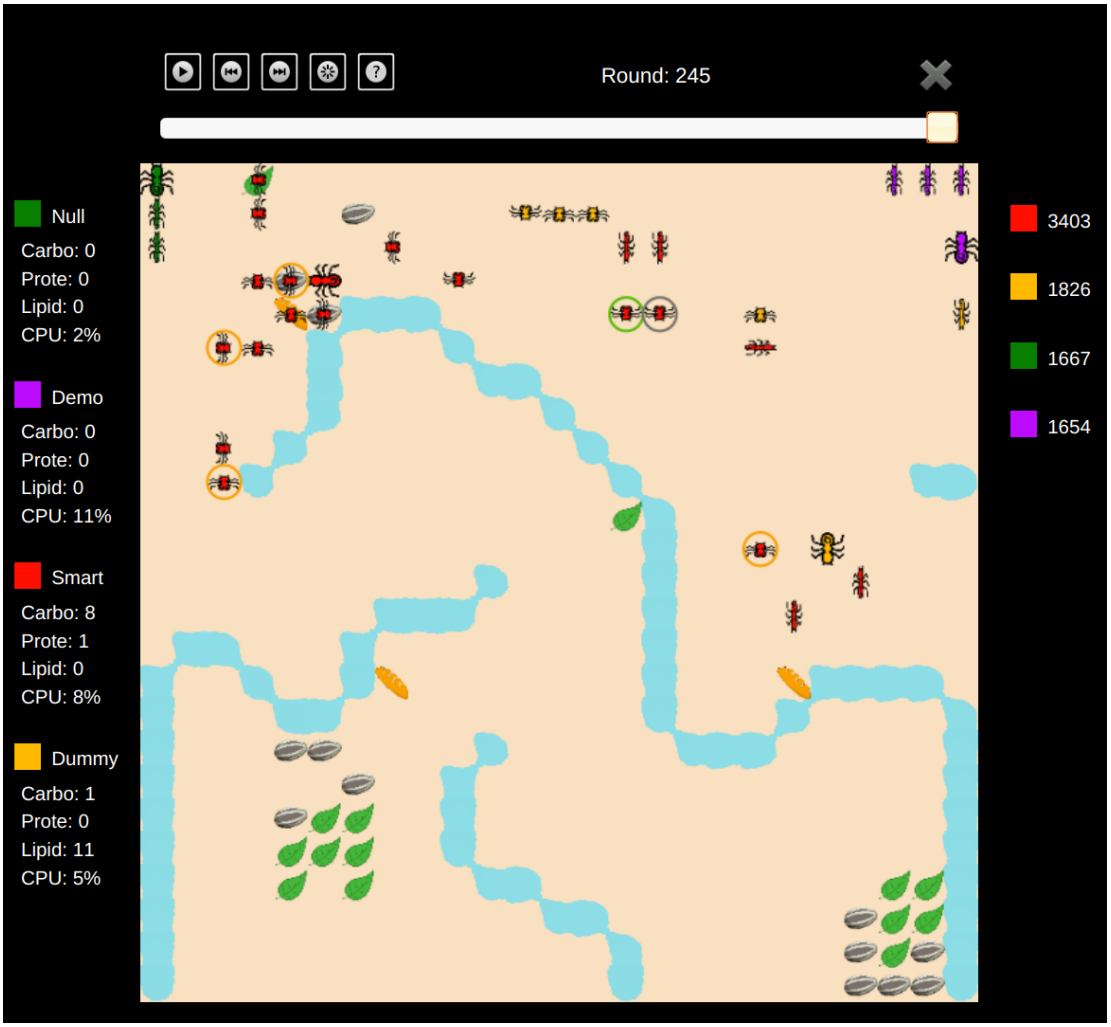

Figura 2: Captura de pantalla del joc.

Figura 3: Representació d'una reina, un soldat i un treballador.

Figura 4: Representació d'un bonus de pa, fulla i llavor.

mat raonable. A més, també es proporciona un jugador "Null" i un jugador "Demo" per facilitar la implementació del vostre propi jugador.

3.1 Executar la primera partida

Aquí explicarem com executar el joc sota Linux, però el mateix hauria de funcionar sota Windows, Mac, FreeBSD, OpenSolaris, ... Només cal una versió recent de `g++`, `make`, i un navegador modern com Mozilla Firefox o Google Chrome.

Per executar la vostra primera partida, seguiu els passos següents:

1. Obriu la consola i feu `cd` al directori on heu extret el codi font.
2. Executeu

   ```
   make all
   ```

   per construir el joc i tots els jugadors. Noteu que `Makefile` identifica com a jugador qualsevol fitxer `AI*.cc`.
3. Això crea un fitxer executable anomenat `Game`. Aquest executable us permet executar el joc usant una comanda com:

   ```
   ./Game Demo Demo Demo Demo -s 30 -i default.cnf -o default.out
   ```

   En aquest cas, això comença una partida amb llavor aleatòria 30 de quatre clons del jugador "Demo", sobre el tauler definit a `default.cnf` (els paràmetres per defecte). La sortida d'aquesta partida es redirigeix al fitxer `default.out`.
4. Per veure la partida, obriu el fitxer `viewer.html` del directori `Viewer` amb el navegador i carregueu el fitxer `default.out`. O alternativament useu el script `viewer.sh`, e.g. `viewer.sh default.out`.

   Nota: Per aconseguir una visió més gran de la partida, poseu el navegador en mode pantalla completa (pitgeu la tecla F11).

Useu

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

per veure la llista de paràmetres que podeu usar. És particularment útil

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

per veure tots els noms de jugadors registrats.

Si cal, recordeu que podeu executar

```
make clean
```

per esborrar tots els fitxers executables i objectes i començar de nou.

3.2 Afegir el vostre jugador

Per crear un nou jugador amb, per exemple, nom `MyPlayer`, copieu `AINull.cc` (un jugador buit que es proporciona com a plantilla) a un nou fitxer `AIMyPlayer.cc`. Aleshores, editeu el nou fitxer i canvieu el

#define PLAYER_NAME Null

per

#define PLAYER_NAME MyPlayer

El nom escollit per al vostre jugador ha de ser únic, no ofensiu i de com a molt 12 caràcters de longitud. Aquest nom s'usarà per definir una nova classe *PLAYER_NAME*, a la que ens referirem com a la vostra classe jugador d'ara en endavant. El nom serà mostrat a la web i durant les partides.

Ara ja podeu començar a implementar el vostre mètode *play* (). Aquest mètode serà cridat a cada ronda i és on el vostre jugador ha de decidir què fer, i fer-ho. Naturalment, podeu definir mètodes i variables auxiliars dins de la vostra classe jugador, però el punt d'entrada del vostre codi serà sempre el mètode *play* ().

Des de la vostra classe jugador també podeu cridar funcions per accedir a l'estat del joc, tal com es defineix a la classe *State* a `State.hh`, i per ordenar a les vostres unitats, tal com es defineix a la classe *Action* a `Action.hh`. Aquestes funcions estan disponibles al vostre codi usant herència múltiple. La documentació sobre les funcions disponibles es pot trobar en els fitxers `.hh` abans esmentats. També podeu examinar el codi del jugador "Demo" a `AIDemo.cc` com a exemple de com usar aquestes funcions. Finalment, també pot ser profitós fer un cop d'ull als fitxers `Structs.hh` per estructures de dades útils, `Random.hh` per utilitats per generar nombres aleatoris, `Settings.hh` per consultar els paràmetres del joc i `Player.hh` pel mètode *me*().

Noteu que no heu d'editar el mètode *factory* () de la vostra classe jugador, ni tampoc la darrera línia que afegeix el vostre jugador a la llista de jugadors registrats.

3.3 Jugar contra el jugador "Dummy"

Per provar la vostra estratègia contra el jugador "Dummy", us proporcionem fitxers objecte `AIDummy.o` d'aquest jugador. D'aquesta manera no teniu accés al codi font, però podreu afegir-lo com a jugador i competir contra ell localment.

Per afegir el jugador "Dummy" a la llista de jugadors registrats, heu d'editar el fitxer `Makefile` i donar a la variable `DUMMY_OBJ` el valor apropiat. Recordeu que els fitxers objecte contenen instruccions binàries per a màquines específiques, de manera que no podem proporcionar un únic fitxer genèric. Si us cal un fitxer objecte per a la vostra arquitectura, contacteu-nos i intentarem proporcionar-vos-el.

També podeu demanar als vostres amics els seus fitxers objecte i afegir-los al `Makefile` donant valor a la variable `EXTRA_OBJ`.

3.4 Restriccions en l'enviament d'un jugador

Quan creieu que el vostre jugador és prou fort com per entrar a la competició, podeu enviar-lo a Jutge.org (`https://www.jutge.org`). Com que s'executarà en un entorn segur per evitar trampes, cal aplicar algunes restriccions al vostre codi:

- Tot el vostre codi font ha d'estar en un sol fitxer (com `AIMyPlayer.cc`).
- No podeu usar variables globals (en lloc d'això, useu atributs de la vostra classe).
- Només se us permet usar llibreries estàndard com ara `iostream`, `vector`, `map`, `set`, `queue`, `algorithm`, `cmath`, ... En molts casos, no cal ni tan sols que incloeu la corresponent llibreria.
- No podeu obrir fitxers ni fer altres crides a sistema (threads, forks, ...).
- A les partides al servidor de Jutge.org, el vostre temps de CPU i la memòria disponible estaran limitats.
- El vostre programa no hauria d'escriure a **cout** ni llegir de **cin**. Podeu escriure informació de depuració a **cerr**, però recordeu que fer això en el codi que pugeu al servidor pot malgastar part del vostre temps limitat de CPU.
- Qualsevol enviament a Jutge.org ha de ser un intent honest de jugar el joc. Qualsevol intent de fer trampa de qualsevol manera serà severament penalitzat.

4 Consells

- Llegiu només les capçaleres de les classes del codi font proporcionat. No us preocupeu per les parts privades o la implementació.
- Comenceu amb estratègies simples, fàcils de codificar i depurar, ja que això és exactament el que necessiteu al principi.
- Definiu mètodes auxiliars bàsics i assegureu-vos que funcionin correctament.
- Intenteu mantenir el vostre codi net. Llavors serà més fàcil canviar-lo i afegir noves estratègies.
- Com de costum, compileu i proveu el vostre codi sovint. És *molt* més fàcil de rastrejar un error quan només heu canviat poques línies de codi.
- Utilitzeu `cerrs` per a escriure informació de depuració i afegiu `asserts` per assegurar-vos que el codi està fent el que hauria de fer. Recordeu que heu de treure (o comentar) els `cerrs` abans de pujar el vostre codi a Jutge.org, perquè fan que l'execució sigui més lenta.
- En depurar un jugador, elimineu els `cerrs` que tingueu al codi dels altres jugadors, de manera que només veieu els missatges que voleu.
- Utilitzant comandes com `grep` a Linux, podeu filtrar la sortida que produeix `Game`.
- Activeu l'opció `DEBUG` al `Makefile`, que us permetrà obtenir traces útils quan el vostre programa avorti. Hi ha també una opció `PROFILE` que podeu utilitzar per optimitzar el codi.
- Si usar `cerrs` no és suficient per depurar el vostre codi, apreneu com utilitzar `valgrind`, `gdb`, `ddd` o qualsevol altra eina de depuració. Són molt útils!
- Podeu analitzar els fitxers que el programa produeix com a sortida, que descriuen com evoluciona el joc després de cada ronda.
- Conserveu una còpia de les versions antigues del vostre jugador. Quan una nova versió estigui llesta, feu-la lluitar contra les anteriors per mesurar-ne la millora.
- 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`).
- Abans de competir amb els vostres companys de classe, centreu-vos en la qualificació i derrotar el jugador "Dummy".

- Assegureu-vos que el vostre programa sigui prou ràpid: el temps de CPU que se us permet és bastant curt.
- Intenteu esbrinar les estratègies dels vostres competidors a base de veure partides. D'aquesta manera podeu intentar defensar-vos contra elles o bé fins i tot millorar-les en el vostre propi jugador.
- **NO DONEU EL VOSTRE CODI A NINGÚ.** Ni tan sols una versió vella. Utilitzem detectors de plagi per comparar tots els parells d'enviaments (inclosos programes de competicions anteriors). Tanmateix, podeu compartir els fitxers compilats .o.
- No espereu fins a l'últim moment per enviar el vostre jugador. Quan hi ha molts enviaments al mateix temps el servidor triga més temps per executar les partides, i podria ser massa tard.
- Podeu enviar noves versions del vostre programa en qualsevol moment.
- I una altra vegada: mantingueu el codi senzill, compileu sovint i proveu sovint. O us penedireu.