 

Fonaments de Programació II Pràctica 3: “El joc del picross”

**Curs 2021-22**

**Estudiants:** Matías Ariel Larrosa Babio,

Pau Revertè,

Pablo

**Professor/a:** Ramon Castells

**Data de lliurament:** x/x/2022

**Contingut**

[Decisions de disseny 3](#_bookmark0)

[Procediments base 3](#_bookmark1)

[Procediments extres 5](#_bookmark2)

[Algorisme 8](#_bookmark3)

[Joc de proves 16](#_bookmark4)

[Procediments base: 16](#_bookmark5)

[Procediments extres: 17](#_bookmark6)

[Procediments en profunditat: 18](#_bookmark7)

[Conclusions 22](#_bookmark8)

[Checklist 23](#_bookmark9)

# Decisions de disseny

A l’hora de dissenyar el programa el problema principal és crear un registre útil per guardar tota la informació necessària del joc. El registre que hem decidit crear és casella\_t i té 3 camps on és guarda tota la informació de la imatge.

**typedef struct{**

**char** valor**;**

**bool** revelat**;**

**bool** flag**;**

**}**casella\_t**;**

**Valor** és l’estat en la qual una casella es troba. Guardarà el caràcter ‘0’ quan en aquella casella estigui buida i, un ‘1’ quan en aquella no ho estigui (■). Aquest valor només es modificarà al inicialitzar el tauler.

**Revelat** és un valor booleà que determina si aquella casella es visible o no per a l’usuari. Al inicialitzar el tauler totes les posicions estaran ocultes (false) i, l’usuari podrà revelar aquella posició seleccionant aquella casella.

**Flag** és un valor booleà que determina si aquella casella esta marcada per l’usuari.Funciona semblant a una bandera del buscamines. L’usuari podrà marcar una casella com un flag per tenir una guia visual per resoldré el tauler a més, podrà introduir un valor especial per triar automàticament totes les caselles que no estiguin marcades.

Així doncs, amb el registre definit podem dissenyar els procediments de la pràctica.

## Procediments base

En aquest apartat s’explicaran els procediments mínims demanats; carregarDades, calcularCantonades, selecció, printTaulerJoc.

**bool** carregarDades(**char** \*filename, **int** \*m, **int** \*n, **int** \*e, **casella\_t** joc[][MAXROWCOL]);

Carrega el contingut del fitxer amb el nom **filename** (\*.txt) al tauler passat “joc”.

El format que accepta el nostre programa es un fitxer de text on la primera línia conte 3 enters corresponents a; les files del tauler, les columnes del tauler, i els errors màxims que te l’usuari al jugar en aquest tauler. La resta del fitxer conté files \* columnes 1s i 0s que correspon al tauler.

Si hi ha menys informació de la que deuria haver-hi o si no s’ha pogut obrir el fitxer el mètode retornarà fals.

**void** calcularCantonades(**int** m, **int** n, **casella\_t** joc[][MAXROWCOL], **int** maxCombinations, **char** cantonades[][maxCombinations]);

Calcula els valors de les files i les columnes que corresponen a la disposició del tauler en aquella fila o columna.

El mètode demana el tauler (dimensions mxn) i una taula on es guardaran aquests valors que tindra m+n files imaxCombinations[[1]](#footnote-1).

Les dades de cada cantonada es guardaran per files on les m primeres files ([0, m-1]) correspondran a les cantonades de les files i les n ([m, n-1]) següents correspondran a les de les columnes. Cada fila de la taula tindrà un valor centinella al final quan la disposició d’aquella cantonada hagi acabat.

Per calcular la disposició de les cantonades es recorre el tauler per files i columnes, contant quants cops apareix un 1 i guardant aquest nombre quan es troba un 0.

**bool** seleccio(**int** m, **int** n, **casella\_t** joc[][MAXROWCOL], **int** i, **int** j, **bool** flag);

Selecciona una posició del tauler del joc, decidint també si es vol revelar la posició o si es vol posar un flag. Per fer-hi aquesta funció necessitem principalment la dimensió de la taula tant les files com les columnes, el tauler del joc (joc[][MAXROWCOL]).  
Aquest procés és un booleà per aquest motiu, retornarà si la posició és correcta (true) o no (false). Primerament, comença revisant que la posició seleccionada es trobi dintre de les mesures de la taula, que no es posen nombres negatius i que la posició seleccionada no es trobi ja revelada. Un cop revisat això donarà la resposta a posicio\_correcta, permeten així o no si entra dintre del if que utilitza el mètode revelat si l'usuari revela la posició o el .flag si vol posar-hi una flag.

**bool** afegirExtensio (**char**\* fitxer, **char**\* extensio);

Afegeix una extensió al final de la cadena de text eliminant el caràcter '\n' del final d'aquest. Per fer aquest procés necessitem la cadena de text a modificar i la cadena de text que conté l'extensió a afegir.

Principalment, inicialitzem una variable que serà la que determinarà la posició a la cadena de text començant en 0. Un cop inicialitzada la variable farem un bucle fins que saltLinea es trobi en la posició del '\n', cal tenir en compte que repetirà el bucle avançant mentre sigui diferent de '\n' i saltLinea sigui més petit que CADMAX. Amb el saltLinea a la posició on es descobreix el '\n' sumarem 5 posicions per poder posar les extensions ('.txt\0' o '.pbm\0'), en cas d'exercir-se de la CADMAX retornarà false, ja que no s'haurà posat l'extensió de forma correcta. En cas de poder haver pogut afegir l'extensió entrarà a un bucle el qual escriurà l'extensió al fitxer.

Finalment fitxer[] a la posició i + saltLinea es col·locarà un '\0' per assegurar que existeixi aquest, retornant així un true a la funció d'afegirExtensió.

**bool** combinarPath (**char** \*fitxer, **char** \*path, **char** \*pathFinal);

Ajuntar el path relatiu i el nom d'un fitxer al path final. Per fer-hi aquesta operació és necessari el nom del fitxer a crear, el path del directori relatiu i el pathFinal que és el path relatiu del fitxer. En aquest proces es retornarà true si s'han pogut ajuntar el path i el fitxer, si no, retornarà false. El primer procés que fem servir és el de posar el path al pathFinal, per posteriorment fer un bucle per a introducir el fitxer dintre del pathFinal. Cal tenir en compte que no s'ha de sobreescriure per aquest motiu se suma el indexfinal que anteriorment determinava el final del path. Finalment, un cop ja es troba tot dintre del pathFinal, es revisa que sigui més petit que CADMAX, si és més petit estarà tot dintre i retornarà true a la funció, si no és així no entrarà al if i retornarà un false.

**void** restaurarJoc (**int** m, **int** n, casella\_t joc[][MAXROWCOL], **int** \*pixelsRestants);

Restaura tot el tauler del joc. Aquest procés el fa mitjançant principalment dos bucles que són els encarregats de recórrer la taula joc[i][j] de .flag, .revelat i .valor, el que fa és posar a la posició d'aquell moment false, és a dir, un 0 a les flags i a revelat, ja que se està restaurant el joc d'aquesta forma. També cal recordar que hi ha un if, que és l'encarregat d'incrementar els píxels restants fins al seu màxim, això ho fa si en la posició actual troba un 1, ja que seria la posició d'un quadrat.

**void** taulerAleatori (**int** m, **int** n, casella\_t joc[][MAXROWCOL], **int** \*pixelsRestants);

Emplenar de jugades un tauler de mesures mxn de forma aleatòria. És necessari principalment la dimensió de la taula, el tauler de joc [][MAXROWCOL] i els píxels restants que serà la quantitat de posicions correctes que hi haurà. Aquest procediment començarà amb la densitat que serà fent un procés mitjançant aquesta fórmula ((rand() % 4 + 1) \* 20) que proporcionarà que el tauler estigui entre un 20 i un 80 % de ple, això ho fem per evitar un joc el qual sigui completament buit o completament ple. Un cop fet això farem dos bucles que determinin la posició a .valor, .revelat i .flag. Cal crear una variable que és 'a' que compararà un número aleatori entre el 1 i el 100 amb la densitat, en cas de ser-hi menor o igual serà true, si no, seria false, si és true incrementarà el valor de pixelsRestants, si no serà false i no entrarà al if. Després s'escriurà el valor 'a' a la posició joc[i][j].valor, mentre també es posa en false aquesta mateixa posició al .revelat i al .flag per posteriorment començar el joc amb un tauler totalment aleatori i nou.

|  |  |  |  |
| --- | --- | --- | --- |
| **Condicions Inicials** | **Resultat esperat** | **Resultat Real** | **Check** |
| S’introdueix el numero 0. | Sortir del programa. |  | ✔ |
| S’introdueix el numero 4. | Entrar a les instruccions del joc. |  | ✔ |
| S’introdueix el numero 3 varies vegades sobreescrivint els taulers. | Generar un tauler aleatori sobreescrivint el tauler anterior per comprovar si son aleatoris. |  | ✔ |
| S’introdueix el numero 1 i carregar el tauler joc sense la extenció .txt. Posteriorment introduir el numero 2 al inici i jugar el tauler carregat. | Ha de carregar el fitxer joc.txt que es 5x5 amb 3 vides. |  | ✔ |
| S’introdueix el numero 5 al inici i s’introdueix el necessari que et demani el programa. | S’ha de generar un fitxer amb el nom que s’hagi posat i amb la mesura i les vides determinades |  | ✔ |
| S’introdueix el número 6 al inici i es determina el nom de la imatge .pbm que es desarà. | S’ha de generar un fitxer .pbm aleatori amb el nom determinat anteriorment. |  | ✔ |
| Jugar un joc carregat i seleccionar una columna i posteriorment una casella correcta. | Ha de accedir a la columna exacta que expressa l’usuari i a la posició que determina aquest també i mostrar un quadrat. |  | ✔ |
| Jugar un joc carregat i seleccionar una columna per posteriorment seleccionar una casella incorrecta | Ha de accedir a la columna i posició que determina l’usuari i mostrar una creu i restar una vida. |  | ✔ |
| Jugar un joc i arribar al màxim d’errors permès. | Ha de prohibir que continuï el joc i retornar al menú inicial. |  | ✔ |
| Completar el tauler amb èxit. | Ha de finalitzar el joc felicitant el jugador i ha de retornar al menú inicial. |  | ✔ |
| Guardar una partida i tornar a entrar sense sobreescriure. | Ha de guardar la partida a la consola, es a dir, si es tanca el programa es perdrà. Al no sobreescriure-la ha de estar tot com s’ha deixat. |  | ✔ |

1. En un tauler de mxn la cantonada que podria tenir mes valors serà primer, la que sigui més llarga i una disposició de 10101... Tenint en compte això maxCombinations = (max(m, n) + max(m, n) % 2)/2 + 1. Per exemple: M, N = 5 -> 10101 -> (5+1)/2 || M, N = 4 ->1010 -> 4/2. Afegim un 1 per saber quan acaba aquella cantonada guardant un valor sentinella com a final. [↑](#footnote-ref-1)