**IES SABADELL Curs 2019/2020**

**CFGS DAM, DAM-vi**

M03.- UF2.-PROGRAMACIÓ ESTRUCTURADA

**Joc del Memory**

**1.- Plantejament**

Objectiu: ser capaços de desenvolupar un programa en llenguatge Java que permetrà a l’usuari jugar al popular joc del memory.

El projecte es planteja per fer-ho progressivament amb sessions guiades de classe.

A cada sessió es presentarà una qüestió relativa al projecte, es comentarà, valoraran possibles solucions, avantatges, inconvenients, ... i es prendran decisions.

La solució que s’anirà construint no té perquè ser la més eficient ni la millor possible, però serà prou didàctica i adequada al nivell del que s’explica a classe.

Les diferents versions del programa s’aniran penjant al curs moodle de la matèria

**2.- Enunciat**

En aquest exercici farem una versió del joc de les parelles **memory**.

Una constant global ens indicarà la mida del tauler, que serà de 4x4. En el programa s'utilitzaran dues matrius. La primera, que anomenarem *tauler*, mostrarà només les caselles descobertes i a les caselles cobertes contindrà interrogants. La segona matriu, *secret*, té totes les caselles descobertes.

Implementa les següents funcions:

***inicialitzaTauler***: rebrà una matriu de caràcters que representa el *tauler* de joc i la mida del tauler, i la inicialitzarà tota amb interrogants.

***mostraTauler***: rebrà el *tauler*  i la mida del tauler, i el mostrarà per pantalla. Cal numerar les files i columnes.

***posaPeces***: rebrà la matriu *secret* i la mida del tauler, i la inicialitzarà, posant les lletres de la A a la H dues vegades cadascuna, en posicions consecutives. Després de cridar aquesta funció, la matriu secret ha de quedar:

AABB

CCDD

EEFF

GGHH

***remenaPeces***: rep una matriu *secret* ja inicialitzada i la mida del tauler, i remena les peces. Per fer-ho, se segueix el següent algorisme:

Per cada casella de la matriu, es tria una casella aleatòria, i s'intercanvien la lletra de la casella que estem mirant amb la de la casella aleatòria.

Per exemple, la primera casella a mirar seria la 0,0. Sortegem una casella i surt la 1,2. Intercanviem el contingut de la casella 0,0 per la casella 1,2. Després seguiríem amb la casella 0,1 i així successivament.

***inicialitzaSecret***: rep una matriu *secret* i es limita a cridar les funcions *posaPeces* i *remenaPeces*.

***validarDada****:* rep la mida del tauler. Demana a l’usuari un valor enter corresponent a la fila/columna. Valida que sigui correcte i retorna el valor corresponent.

***validarCasella***: rep una matriu *tauler*, i una *fila* i *columna* corresponents a la casella que es vol destapar. Retorna un booleà. La funció s'assegura que la casella encara no està destapada (és a dir, que al tauler hi ha un interrogant). Si les comprovacions fallen, es retorna un false, en cas contrari tornarà true.

***torn***: rep el *tauler*, el *secret* i la mida del tauler; i retorna si s'ha trobat una parella o no. Cada torn consisteix en destapar dues peces. Per la primera, es demanarà la posició a l'usuari (utilitzant les dues funcions anteriors), quan sigui correcte, es copiarà el contingut de la casella des de *secret* cap a *tauler*, i es mostrarà el *tauler*. Per la segona tirada, es faran els mateixos tres passos. Després, es comprovarà si les dues caselles descobertes coincideixen. Si coincideixen, la funció retornarà cert. Si no coincideixen, es tornaran a posar interrogants a les dues posicions descobertes del *tauler*, i es tornarà fals.

***joc***: gestiona una partida sencera, i no rep ni retorna cap valor. Com a variables locals crearem les matrius del *tauler* i del *secret*, que inicialitzarem cridant les funcions corresponents. El joc continuarà fins que s'hagin trobat totes les parelles. Per cada torn, es dirà a quin dels dos jugadors toca tirar i es cridarà la funció *torn*. Si *torn* torna cert, vol dir que el jugador ha encertat una parella. En aquest cas, se li suma un punt, i se l'informa que pot tornar a tirar. Si no encerta, se li diu que ha fallat, i es canvia de jugador. Quan la partida hagi acabat, es mirarà quin dels dos jugadors té més punts, i es dirà qui ha guanyat.

***main***: el *main* es limita a inicialitzar el generador de nombres aleatoris, i cridar a *joc*. Quan el joc acaba, pregunta si es vol fer altra partida o no.

***Exemple d'execució***

Benvingut al joc del memory!

0 ????

1 ????

2 ????

3 ????

0123

Torn del jugador 0:

Primera tirada...

Introdueix la fila: 1

Introdueix la columna: 2

0 ????

1 ??C?

2 ????

3 ????

0123

Segona tirada...

Introdueix la fila: 2

Introdueix la columna: 3

0 ????

1 ??C?

2 ???D

3 ????

0123

No coincideixen...

Torn del jugador 1:

Primera tirada...

Introdueix la fila: 0

Introdueix la columna: 0

0 B???

1 ????

2 ????

3 ????

0123

Segona tirada...

Introdueix la fila:

Al cap d'algunes tirades...

Torn del jugador 0:

Primera tirada...

Introdueix la fila: 1

Introdueix la columna: 2

0 ????

1 ??C?

2 ????

3 ????

0123

Segona tirada...

Introdueix la fila: 0

Introdueix la columna: 1

0 ?C??

1 ??C?

2 ????

3 ????

0123

Coincideixen! Pots tornar a tirar.

Torn del jugador 0:

Primera tirada...

Introdueix la fila:

Al cap d'algunes tirades més...

Torn del jugador 1:

Primera tirada...

Introdueix la fila: 3

Introdueix la columna: 3

0 BCBE

1 EFCG

2 GD?D

3 FAAH

0123

Segona tirada...

Introdueix la fila: 2

Introdueix la columna: 2

0 BCBE

1 EFCG

2 GDHD

3 FAAH

0123

Ha guanyat el jugador 0 amb 5 punts!

Vols fer una altra partida (s/n)?

**3.- Sobre el Programa Solució**

La solució està inspirada en el programa fet per un alumne (Carles Bonache).

El programa s’ha plantejat amb la intenció de respectar al màxim les indicacions de l’enunciat, amb molt pocs arrengaments estètics. Hi ha algun canvi respecte el nom de les funcions. És possible fer el programa respectant completament l’enunciat.

S’ha encapsulat tot el necessari per jugar al Memory en una variable de tipus estructura/registre. D’aquesta manera es facilita molt la programació de funcions i pas de paràmetres, així com la possible ampliació de funcionalitats (escalabilitat).

El programa s’ha plantejar aplicant les tècniques del disseny descendent i programació per capes, intentant “amagar” la manera com implementem i treballem amb el joc del memory. **El principal objectiu d’aquest treball, per sobre de si som capaços de fer un programa com aquest amb les nostres forces, és comprendre el mecanismes i motivacions de la programació estructura i disseny descendent.**

Aquesta manera d’encarar el projecte recorda, en certa manera, la programació orientada objectes.

Es pot fer un esquema de com ha quedat el programa:

















* El Nivell 1 és és el de més alt nivell i és bàsicament el punt d’entrada al programa
* El Nivell 2 **implementa** **la lògica** del programa, (en una base de dades vindria a ser el model conceptual).

Aquest nivell pot tenir descomposicions d’alguns procesos “complicats”, fins a fer-los més asequibles.

Finalment, ha de descompondre el programa en petites funcions, força asequibles, que fan tasques molt específiques i que conformen el següent nivell:

* El Nivell 3 el componen funcions que tenen a veure amb l’implementació més bàsica de les dades i procesos del programa. Fixeu-vos que al nostre programa les funcions de nivell 3 són les que toquen l’estructura interna dels taulers (és a dir, accedeixen a les caselles). Això no és del tot cert a la funció fer\_jugada, però … oi que no costaria gaire que ho fos?

**4.- Ampliacions**

El programa, tal qual s’ha plantejat, és força limitat.

Es proposen les següents ampliacions:

1. Poder jugar contra l’ordinador.
   1. Amb joc “aleatori” per part de l’ordinador
   2. Amb joc “intel·ligent” per part de l’ordinador.
2. Poder marcar d’un color diferent els encerts de cada jugador sobre el tauler de joc.
3. Poder definir la dimensió del tauler de joc al començar. Aquesta dimensió ha de ser un número parell, per tal que el nº de caselles sigui parell.
4. Altres ...

Per implementar aquestes ampliacions, potser cal tocar l’estructura de dades.