Pràctiques d'Estructura de Computadors

Nivell Bàsic

1 Introducció

La pràctica consisteix en implementar un joc de Memory, on es tracta de descobrir totes les parelles de valors d'un tauler. Tenim un tauler quadrat de 20 caselles (5x4), i a cada casella s'amaga un nombre enter entre 0 i 9, de manera cada valor es troba dues vegades al tauler. Aquesta estructura es troba emmagatzemada en una matriu anomenada gameCards (*int gameCards[5][4]*).

El jugador ha d'anar descobrint caselles, de dues en dues, i si descobreix dues caselles que contenen el mateix valor, la parella queda descoberta, mentre que si les dues caselles descobertes, no contenen el mateix valor, ambdues s'han de tornar a tancar.

La pràctica consta d'un programa en C, que us donem fet i **NO PODEU MODIFICAR**, un fitxer d'encapçalament, **que tampoc podeu modificar**, i un esquelet d'un programa en assemblador en el que heu d'implementar les subrutines que us indiquem. El programa en C, genera un menú en pantalla i permet accedir a les diferents opcions del menú. Les diferents opcions del menú s'han d'implementar amb les subrutines en assemblador que us indiquem.

El programa en assemblador conté l'estructura bàsica de les subrutines que heu d'implementar i algunes subrutines ja implementades per a gestionar l'entrada/sortida.

1.1 Objectius de la pràctica:

- Familiaritzar-se amb l'entorn de desenvolupament (Visual Studio).
- Aprendre els conceptes bàsics del llenguatge de baix nivell de l'arquitectura Intel x86.
- Observar els canvis que produeixen les instruccions sobre els registres, flags o registres d'estat i la memòria.
- Treballar en C i llenguatge Assembly Intel x86.

1.2 Material

- Microsoft Visual Studio 2019 Community.
- Documentació sobre l'entorn i configuració del projecte al Campus Virtual.
- Esquelet del Projecte de la pràctica que inclou el programa en C, el fitxer d'encapçalament .h i l'esquelet del programa assemblador (disponible al Campus Virtual).

1.3 Desenvolupament de la pràctica:

- La pràctica es realitzarà en 6 sessions.
- Hi ha una prova de validació individual un cop hagin acabat les sessions de laboratori.
- La pràctica està dividida en 3 nivells: bàsic, mig i avançat. Quan un grup acabi un nivell, haurà de presentar-lo al membre de l'equip docent responsable del seu

grup de pràctiques durant la sessió. Si el grup rep el vist i plau, haurà de lliurar el seu codi al Campus Virtual i se li obrirà el següent nivell. No es poden saltar nivells (per exemple, presentar directament el nivell mig sense haver presentat el bàsic), ni presentar més d'un nivell per sessió.

- La nota de laboratori depèn del nivell assolit i la qualitat/funcionalitat de la solució proposada. El nivell bàsic dona dret a una nota fins a 6. El mig permet sumar fins a un punt més i l'avançat un altre punt més com a màxim. En el Campus Virtual disposeu d'un document on s' explica amb més detall el mètode d'avaluació de l'assignatura.

- Un cop finalitzades totes les sessions de laboratori caldrà lliurar una memòria final de pràctiques en format pdf, seguint la plantilla que us proporcionarem al Campus Virtual.

2 Nivell Bàsic (2-3 sessions):

En aquest primer nivell s'han d'implementar unes funcionalitats bàsiques, és a dir, les 5 primeres opcions del menú. Aquests opcions son:

- **1. Show Cursor:** En aquesta opció, el cursor s'ha de posicionar a la posició indicada en el programa en C (fila 1, columna 2) (penseu que la primera posició de la matriu és la (0, 0)). Aquesta opció del menú crida a la subrutina *posCurScreen* que heu d'implementar.

- **2. Move Cursor:** En aquesta opció s'ha de moure el cursor fent servir les tecles 'i' (amunt), 'k' (avall), 'j' (esquerra) i 'l' (dreta). El programa espera que pitgeu una tecla, fa el moviment corresponent i espera una nova tecla abans de tornar al menú principal. Si la tecla pitjada per a fer el moviment no correspon a les tecles permeses, el programa ha de seguir esperant una tecla vàlida. Aquesta opció del menú crida a la subrutina *moveCursor* (que heu d'implementar) que ha de cridar a la subrutina *getMove* (que també heu d'implementar) i a la subrutina *posCurScreen* (que heu implementat a la primera opció del menú.

- **3. Move Cursor Continuous:** En aquest cas el moviment es va repetint fins que pitgem la tecla 's' per a sortir o ' ' per a obrir una casella. Cal controlar que el cursor no surti dels límits del tauler. Heu d'implementar la subrutina *movCursorContinuous* que cridarà a la subrutina *moveCursor* que ja heu implementat a la opció anterior.

- **4. Open Card:** En aquesta opció el cursor es va movent de la mateixa forma que en l'opció anterior, fins que pitgem la tecla 'espai'. Quan es pitja ' ' es mostra el contingut corresponent a la casella del tauler sobre la que tenim situat el cursor, fent servir la variable *carac* i cridant a la subrutina *printch*. Si pitgem la tecla 's', sortim sense mostrar el contingut de cap tecla. Per aquesta opció heu d'implementar la subrutina *openCard*. Aquesta subrutina cridarà a *movCursorContinuous* i quan pitgem ' ' cridarà a una nova subrutina *calcIndex* per a determinar la posició de la matriu i accedir i mostrar per pantalla el contingut d'aquesta casella.

- **5. Open Card Continuous:** En aquesta opció el cursor es va movent de la mateixa forma que en l'opció anterior, i anem obrint les caselles que desitgem pitjant la tecla espai. El procés finalitza quan pitgem la tecla 's' per a sortir. Cal implementar una subrutina nova *openContinuous* que anirà cridant a *openCard*.

- **0. Exit:** Amb aquesta opció, que ja us donem implementada, es surt i finalitza l'execució del programa.

Com ja hem dit, la pràctica consta d'un programa en C que us donem fet i NO PODEU MODIFICAR, i un programa en assemblador que conté algunes subrutines ja implementades (per a posicionar el cursor, llegir caràcters de teclat i escriure caràcters per pantalla) i les capçaleres de les subrutines que heu d'implementar. No heu d'afegir altres variables ni subrutines.

2.1 Subrutines que cal implementar en assemblador per al Nivell Bàsic:

- **posCurScreen:**

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; Posicionar el cursor a la pantalla, dins el tauler en funció de les variables row (int) i col (char)
; a partir dels valors de les variables RowScreenIni i ColScreenIni.
; Primer cal restar 1 a row (fila) per a que quedi entre 0 i 4
; i convertir el char de la columna (A..D) a un número entre 0 i 3.
; Per calcular la posició del cursor a pantalla (rowScreen) i
; (colScreen) utilitzar aquestes fórmules:
;        rowScreen=rowScreenIni+(row*2)
;        colScreen=colScreenIni+(col*4)
; Per a posicionar el cursor a la pantalla cridar a la subrutina gotoxy
; que us donem implementada
;
; Variables utilitzades:
; row: fila per a accedir a la matriu gameCards
; col : columna per a accedir a la matriu gameCards
; rowScreen : fila on volem posicionar el cursor a la pantalla.
; colScreen : columna on volem posicionar el cursor a la pantalla.
; rowScreenIni : fila de la primera posició de la matriu a la pantalla.
; colScreenIni : columna de la primera posició de la matriu a la pantalla.
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

- **getMove:**

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; Llegir un caràcter de teclat cridant a la subrutina que us donem implementada getch.
; Verificar que el caràcter introduït es troba entre els caràcters 'i' i 'l',
; o bé correspon a les tecles espai ' ' o 's', i deixar-lo a la variable tecla.
; Si la tecla pitjada no correspon a cap de les tecles permeses,
; espera que pitgem una de les tecles permeses.
;
; Variables utilitzades:
; tecla : variable on s'emmagatzema el caràcter corresponent a la tecla pitjada
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

- **moveCursor:**

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; Cridar a la subrutina getMove per a llegir una tecla
; Actualitzar les variables (row) i (col) en funció de
; la tecla pitjada que tenim a la variable (tecla)
; (i: amunt, j:esquerra, k:avall, l:dreta).
; Comprovar que no sortim del tauler,
; (row) i (col) només poden
; prendre els valors [1..5] i [A..D], respectivament.
; Si al fer el moviment es surt del tauler, no fer el moviment.
; Posicionar el cursor a la nova posició del tauler cridant a la subrutina posCurScreen
;
; Variables utilitzades:
; tecla : caràcter llegit de teclat
; 'i': amunt, 'j':esquerra, 'k':avall, 'l':dreta
; row : fila del cursor a la matriu gameCards.
; col : columna del cursor a la matriu gameCards.
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

- **movCursorContinuous:**

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; Subrutina que implementa el moviment continu
; del cursor fins que pitgem 's' o espai ' '
; S'ha d'anar cridant a la subrutina moveCursor
;
; Variables utilitzades:
; tecla: variable on s'emmagatzema el caràcter llegit
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

- **calcIndex:**

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; Aquesta subrutina serveix per a poder accedir a les components de la matriu
; i poder obrir les caselles
; Calcular l'índex per a accedir a la matriu gameCards en assemblador.
; gameCards[row][col] en C, ´es [gameCards+indexMat] en assemblador.
; on indexMat = (row*4 + col (convertida a número))*4 .
;
; Variables utilitzades:
; row: fila per a accedir a la matriu gameCards
; col: columna per a accedir a la matriu gameCards
; indexMat: índex per a accedir a la matriu gameCards
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

- **OpenCard:**

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; S'ha de cridar a movCursorContinuous per a triar la casella desitjada.
; Un cop som a la casella desitjada premem al tecla ' ' (espai per a veure el contingut)
; Calcular la posició de la matriu corresponent a la
; posició que ocupa el cursor a la pantalla, cridant a la subrutina calcIndex.
; Mostrar el contingut de la casella corresponent a la posició del cursor al tauler.
; Considerar que el valor de la matriu és un  int (entre 0 i 9)
; que s'ha de "convertir" al codi ASCII corresponent.
;
; Variables utilitzades:
; row : fila per a accedir a la matriu gameCards
; col : columna per a accedir a la matriu gameCards
; indexMat : índex per a accedir a la matriu gameCards
; gameCards : matriu 5x4 on tenim els valors de les cartes.
; carac : caràcter per a escriure a pantalla.
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

- **openCardContinuous:**

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; S'ha d'anar cridant a openCard fins que pitgem la tecla 's'
;
; Variables utilitzades:
; tecla : variable on s'emmagatzema le caràcter llegit
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```