Pràctiques d'Estructura de Computadors

Nivell Mig

1 Introducció

En aquest nivell mig heu d'implementar el nucli del joc del Memory aprofitant bona part de la feina feta en el nivell bàsic.

En concret, heu d'implementar la part que permet obrir dues posicions, comprovar si son igual i si no ho son, tornar-les a tancar.

Per això us proporcionem un nou programa en C que incorpora dues opcions més al menú i introdueix algunes variables noves. Recordeu que aquest programa en C **NO EL PODEU MODIFICAR**. També us donem un nou esquelet .asm, en el que haureu de copiar les subrutines que heu implementat en el nivell bàsic, i un nou fitxer d'encapçalament, de manera que haureu de crear un nou projecte igual que el vàreu crear pel nivell bàsic.

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 del nivell mig de la pràctica que inclou el programa en C, el fitxer d'encapçalament .h i l'esquelet del programa assemblador (disponible al Campus Virtual).

2 Nivell Mig (2-3 sessions):

En aquest segon nivell s'han d'implementar les opcions del menú relatives a les parelles. Aquests opcions son:

- **6. Open Pair:** Aquesta opció del menú crida a la subrutina *openPair* que heu d'implementar en l'esquelet del programa .asm. Aquesta opció del menú ha d'obrir dues caselles del taulell i si son iguals deixar-les obertes, però si no son iguals ha de tancar-les. Heu de tenir en compte que no podeu obrir una casella que ja està oberta, per això hem afegit una matriu de char anomenada *Board* de les mateixes dimensions que *gameCards* que us pot servir per a portar el control. D'altra banda heu de pensar que quan obriu la primera casella, i després la segona, heu de guardar el valor, i la posició per a poder tancar-la en cas necessari. Hem afegit variable per aquest fi. Per últim, hem afegit un indicador a la part de dalt del tauler (posició 3,30) que indica

si estem obrint la primera carta o la segona de la parella fent servir la variable *Num_Card*, i un indicador a la posició (3,41) que senyala quin jugador està jugant mitjançant la variable Player.

- **7. Open Pair Continuous:** Aquest opció del menú permet anar jugant de manera continua fins que pitgem la tecla 's' per a sortir. De manera que podem anar descobrint parelles, tancant les que no siguin parella, de forma continua. Si pitgem 's', ja sigui quan hem d'obrir la primera com la segona casella, hem de tornar al menú principal. Aquesta opció crida a la subrutina *openPairContinuous*, que heu d'implementar, i que cridarà a la subrutina *openPair* que heu implementat per a l'opció anterior.

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 Mig:

- **openPair:**

  ;;
  ; Posicionar el cursor a la posició 3,30 de la pantalla cridant a la subturina gotoxy
  ; Mostrar el valor de la variable *Num*_Card (1 o 2)
  ; Posicionar el cursor a la posició 3,41 de la pantalla cridant a la subrutina gotoxy
  ; Mostrar el valor de la variable Player (1 o 2)
  ; Posicionar el cursor al taulell de joc i moure'l de forma continua fins que pitgem 's' o ' '
  ; Quan pitgem ' ', obrim la casella (comprovar que no està oberta i marcar-la com oberta)
  ; Tornar a moure el cursor de forma continua fins que pitgem 's' o ' '
  ; Quan pitgem ' ', obrim la casella (comprovar que no està oberta i marcar-la com oberta)
  ; Comprovar si els valors de les dues caselles coincideixen. Si coincideixen, posar un 1 a HitPair.
  ; Si no coincideixen, tancar les dues caselles i desmarcar-les com a obertes.
  ;
  ; **Variables utilitzades:**
  ; Num_Card: Variable que indica si estem obrint la primera o la segona casella de la parella.
  ; carac : caràcter a mostrar per pantalla
  ; row : fila del cursor a la matriu gameCards o Board.
  ; col : columna del cursor a la matriu gameCards o Board.
  ; rowScreen: Fila de la pantalla on volem posicionar el cursor.
  ; colScreen: Columna de la pantalla on volem posicionar el cursor.
  ; indexMat: Índex per accedir a la posició de la matriu.
  ; gameCards: Matriu amb els valors de les caselles del tauler.
  ; Board: Matriu que indica si la casella està oberta o no.
  ; firstVal, firstRow, firstCol: Dades relatives a la primera casella de la parella.
  ; secondVal, secondRow, secondCol: Dades relatives a la segona casella de la parella.
  ; Player: Indica el jugador al que li correspon el torn.
  ; HitPair: Variable que indica si s'ha fet una parella (0 No parell – 1 Parella)
  ; tecla: Codi ascii de la tecla pitjada.
  ;;

- **openPairsContinuous:**

  ;;
  ; Aquesta subrutina ha d'anar cridant a la subrutina anterior *OpenPair*,
  ; fins que pitgem la tecla 's'
  ;
  ; **Variables utilitzades:**
  ; tecla: Codi ascii de la tecla pitjada.

;;;