INGENIERÍATÉCNICA EN INFORMÁTICA DE GESTIÓN. CURSO 2001-2002
LABORATORIO DE ESTRUCTURA Y TECNOLOGÍA DE COMPUTADORES

Sesión 12
Implementación de bucles

Una vez introducidas, en el capítulo anterior, el conjunto de instrucciones y pseudoinstrucciones del MIPS R2000 que permiten implementar estructuras condicionales como *Si-entonces, Si-entonces-sino,* típicas de un lenguaje de alto nivel, en esta sesión veremos cómo implementar con el lenguaje ensamblador del procesador anterior estructuras de alto nivel repetitivas como *Mientras* y *Para*.

12.1. Estructura de control repetitiva *Mientras*

Crea un fichero con el siguiente fragmento de código que implementa una estructura de control repetitiva *Mientras*:

```
            .data
cadena:     .asciiz "hola"
            .align  2
n:          .space  4
            .text
main:       la    $t0,cadena    #carga dir. cadena en $t0
            andi  $t2,$t2, 0    #$t2=0
mientras:   lb    $t1,0($t0)    #almacenar byte en $t1
            beq   $t1,$0,finmientras #si $t1=0 saltar
                                #a finmientras
            addi  $t2,$t2, 1    #$t2=$t2+1
            addi  $t0,$t0, 1    #$t0=$t0+1
            j     mientras      #saltar a mientras
finmientras: sw   $t2,n($0)     #almacenar $t2 en n
```

A continuación se muestra la descripción algorítmica de este programa en ensamblador:

```
PROGRAMA MIENTRAS-1
VARIABLES
     VECTOR DE CARACTERES: cadena(4)='hola';
     ENTERO: n, i;
INICIO
     i=0;
     n=0;
     Mientras (cadena[i]!=0) hacer
          n=n+1;
          i=i+1;
     FinMientras
FIN
```

Borra los valores de la memoria y carga el fichero que contiene el programa en ensamblador en el simulador.

Cuestión 12.1: Ejecuta paso a paso el programa anterior y comprueba detenidamente la función de cada una de las instrucciones que constituyen el programa ensamblador.

Cuestión 12.2: ¿Qué valor se almacena en `n` después de ejecutar el programa?

Cuestión 12.3: Implementa en ensamblador el siguiente programa descrito en lenguaje algorítmico

```
PROGRAMA MIENTRAS-2
VARIABLES
     VECTOR DE CARACTERES:
          tira1(6)="holaaa"; tira2(5)="adios";
     ENTERO: n, i;
INICIO
     i=0;
     n=0;
     Mientras ((tira1[i])!=0) and (tira2[i]!=0)) hacer
          n=n+1;
          i=i+1;
     FinMientras
FIN
```

12.2. Estructura de control repetitiva *Para*

Crea un fichero con el siguiente fragmento de código que implementa una estructura de control repetitiva *Para*. Ésta es una estructura de control *Mientras* donde el bucle se repite un número de veces conocido a priori. Para implementarla se necesita pues un contador (inicializado a 0 ó al máximo número de veces menos uno que queremos que se repita el bucle) que llevará la cuenta de las veces que éste se repite. Este contador se deberá incrementar o decrementar dentro del bucle. La condición de salida del bucle siempre será comprobar si este contador alcanza la cota superior o inferior (0) (según el contador cuente de forma ascendente o descendente, respectivamente).

```
            .data
     vector: .word    6,7,8,9,10,-1
     res:    .space   4
            .text
     main:   la   $t2,vector     #$t2=dirección de vector
          and  $t3,$0,$0      #$t3=0
          and  $t0,$0,$0      #$t0=0
          addi $t1,$0,6       #$t1=6
para:     bge  $t0,$t1,finpara #si $t0>=$t1 saltar finpara
          lw   $t4,0($t2)     #carga elemento vector en $t4
          add  $t3,$t4,$t3    #suma los elementos del vector
          addi $t2,$t2,4      #$t2=$t2+4
          addi $t0,$t0,1      #$t0=$t0+1
          j    para           #saltar a bucle
finpara:  sw   $t3,res($0)    #almacenar $t3 en res
```

La descripción algorítmica de este programa en ensamblador se puede transcribir como el bucle repetitivo *Mientras* que se muestra a continuación:

```
PROGRAMA PARA-1
VARIABLES
     VECTOR DE ENTEROS: v(6)=(6,7,8,9,10,-1);
     ENTERO: res, i;
INICIO
     res=0;
     i=0;
     Mientras (i<=5) hacer
          res=res+v[i];
          i=i+1;
     FinMientras
FIN
```

Una descripción algorítmica equivalente y más sencilla consiste en utilizar una estructura de control *Para*:

```
PROGRAMA PARA-1
VARIABLES
     VECTOR DE ENTEROS: v(6)=(6,7,8,9,10,-1);
     ENTERO: res, i;
CODIGO
     res=0;
     Para i=0 hasta 5 hacer
          res=res+v[i];
     FinPara
FIN
```

Borra los valores de la memoria, carga el fichero que contiene el programa en ensamblador en el simulador.

Cuestión 12.4: Ejecuta paso a paso el programa y comprueba detenidamente la función que tiene cada una de las instrucciones que componen el programa en ensamblador.

Cuestión 12.5: ¿Qué valor se almacena en `res` después de ejecutar el código anterior?

Cuestión 12.6: Implementa en ensamblador el siguiente programa descrito en lenguaje algorítmico:

```
PROGRAMA PARA-2
VARIABLES
     VECTOR DE ENTEROS:  v1(8)=(6,7,8,9,10,-1,34,23);
                         v2(8);
INICIO
     Para i=0 hasta 7 hacer
          v2[i]=v1[i]+1;
     FinPara
FIN
```

Problemas propuestos

1. Implementa un programa en ensamblador que, dado un vector de enteros, obtenga como resultado cuántos elementos de éste son iguales a cero. El resultado se almacenará sobre la variable "total". El programa deberá inicializar los elementos del vector en memoria, así como una variable que almacenará el número de elementos que tiene el vector y reservará espacio para la variable resultado.

2. Implementa un programa en ensamblador en el que, dado un vector de enteros "V", obtenga cuántos elementos de este vector están dentro del rango determinado por dos variables "rango1" y "rango2". El programa deberá inicializar los elementos del vector en memoria, una variable que almacenará el número de elementos que tiene ese vector y dos variables donde se almacenarán los rangos. También deberá reservar espacio para la variable resultante.

3. Implementa un programa en ensamblador en el que, dada una cadena de caracteres, contabilice cuántas veces se repite un determinado carácter en la misma. El programa deberá inicializar la cadena en memoria, y ésta deberá finalizar con el carácter nulo. También deberá reservar espacio para la variable resultado.

4. Modifica el ejercicio anterior para encontrar dentro de la cadena una subcadena de longitud menor (p.e. 2 caracteres de longitud). El resultado será la posición donde se encuentre la subcadena o 0 si no se encuentra.