ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

ESTRUCTURA DE COMPUTADORES

GRADO EN INGENIERÍA INFORMÁTICA

ACTIVIDADES PRÁCTICAS

ACTIVIDAD 4.2:
TRAZAS DE PROGRAMAS EN ENSAMBLADOR

CURSO 2021-22

INTRODUCCIÓN

En esta actividad se propone la realización de una serie de trazas de programas en ensamblador de **MIPS**, útiles de cara a la preparación del examen de prácticas de la asignatura.

Las trazas se realizarán mediante el simulador **MARS**, de Peter Sanderson y Kenneth Vollmar, de la Missouri State University, y se grabarán en el fichero **trazas2.xlsx**.

EJERCICIO 1: SUMA DE VECTORES

Se pretende realizar un programa en ensamblador que sume dos vectores de 16 números de 32 bits expresados en complemento a 2.

A continuación se muestra una posible solución en lenguaje C (obviando la entrada/salida):

```
void salida (void);
#define N 16
int x[N] = {1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16};
int y[N] = {100,200,300,400,500,600,700,800,900,1000,1100,1200,1300,1400,1500,
            1600};
int z[N];
register int i;
int main (void) {
   for (i = 0; i < N; i++)
      z[i] = x[i] + y[i];
   salida();
   return 0;
}
```

En el fichero **vectores_int_suma.asm** se muestra un esqueleto de la solución. Los ficheros **salida.asm**, **stdio.asm** y **stdioaux.asm** contienen código que permite presentar los resultados por consola, y no deben ser modificados.

Se pide:

a) Programar la solución completando el código del fichero.
b) Realizar una traza de ejecución del programa, escribiendo los valores de los registros y variables involucrados únicamente en los siguientes puntos:
   1. Justo antes de entrar en el bucle (una única vez).
   2. En la ramificación condicional correspondiente a la evaluación de la condición del bucle (una vez en cada iteración).
   3. En instrucción de actualización del índice del for (una vez en cada iteración). Incluir en esta anotación los valores de los elementos x[i], y[i] y z[i], es decir, los elementos de los tres vectores accedidos en la iteración.
   4. En la instrucción siguiente al bucle (una única vez).

Incluir la traza en el fichero **trazas2.xlsx**.

Juego de datos de entrada para realizar las trazas: los vectores del enunciado.

EJERCICIO 2: VALORES MÁXIMO Y MÍNIMO DE UN VECTOR DE ENTEROS

Se pretende realizar un programa de ensamblador que calcule los valores máximo y mínimo de un vector de enteros.

A continuación se muestra una posible solución en C:

```
void salida (void);

int vector[N] = {1,-3,55,8905,-111,90,543,-4321,0,43,99,-34,-55,4231,45,
-4532543}
int maximo, minimo;
register int i, aux;

int main (void) {
      maximo = vector[0];
      minimo = maximo;
      for (i = 1; i < N; i++) {
            aux = vector[i];
            if (aux < minimo)
                  minimo = aux;
            else if (aux > maximo)
                  maximo = aux;
      }
      salida();
      return 0;
}
```

En el fichero **vector_int_minmax.asm** se proporciona un esqueleto de la solución.

Se pide:

c) Programar la solución completando el código del fichero.

d) Realizar una traza de ejecución del programa, escribiendo los valores de los registros y variables involucrados únicamente en los siguientes puntos:

1. Justo antes de entrar en el bucle (una única vez).
2. En la ramificación condicional correspondiente a la evaluación de la condición del if (una vez en cada iteración).
3. En instrucción de actualización del índice del for (una vez en cada iteración).
4. En la instrucción siguiente al bucle (una única vez).

Incluir la traza en el fichero **trazas2.xlsx**.

Juego de datos de entrada para realizar las trazas: el vector del enunciado.

EJERCICIO 3: INTERPOLACIÓN LINEAL CON COMA FLOTANTE

Se pretende realizar una interpolación lineal del valor de un punto de una recta situado entre otros dos puntos dados, mediante la expresión siguiente:

$$f(x) = \frac{(x - x_1) \cdot (y_2 - y_1)}{(x_2 - x_1)} + y_1$$

donde (x_1, y_1) y (x_2, y_2) son las coordenadas cartesianas de 2 puntos, mientras que x es la abscisa de un punto intermedio situado entre los dos anteriores.

Una solución parcial al problema anterior escrita en lenguaje C es la siguiente (obviando la parte de entrada/salida por consola):

```
float x,y,x1,y1,x2,y2;
...
y = ((x-x1)*(y2-y1)) / (x2-x1) + y1;
...
```

A continuación se muestra una posible solución en lenguaje ensamblador, que deberá programarse en el fichero **interpolación_lineal_float.asm** (se obvian la parte de reserva de espacio para datos en memoria y la entrada/salida por consola):

```
# y = ((x-x1)*(y2-y1)) / (x2-x1) + y1;
        l.s     $f20,x          # Carga de x en $f20
        l.s     $f21,x1         # Carga de x1 en $f21
        l.s     $f22,y1         # Carga de y1 en $f22
        l.s     $f23,x2         # Carga de x2 en $f23
        l.s     $f24,y2         # Carga de y2 en $f24
        sub.s   $f2,$f20,$f21   # f2 = x - x1
        sub.s   $f3,$f24,$f22   # f3 = y2 - y1
        mul.s   $f2,$f2,$f3     # f2 = f2*f3
        sub.s   $f3,$f23,$f21   # f3 = x2 - x1
        div.s   $f2,$f2,$f3     # f2 = f2 / f3
        add.s   $f2,$f2,$f22    # f2 = f20 + f22
        s.s     $f2,y           # Almacenamiento del resultado en y
```

Se pide realizar varias trazas del programa pedido. obviando la parte de E/S por consola. Incluir las trazas en el fichero **trazas2.xlsx**.

Juegos de datos de entrada para realizar las trazas:

Datos	Juego 1	Juego 2
x1	2.0	-3.5
y1	0.0	6.3
x2	6.0	3.5
y2	8.0	14.8
x	4.0	0.0

EJERCICIO 4: ESTRUCTURA DE SELECCIÓN CON DATOS EN COMA FLOTANTE

Se pretende realizar el cálculo expresado mediante el siguiente fragmento de código:

```
float x, y, z;
   ...
   if (x >=y) x = x+z;
   else       x = x-z;
   ...
```

A continuación se muestra una posible solución parcial en lenguaje ensamblador:

```
# if (x >=y)
if:   l.s    $f20,x
      l.s    $f22,y
      c.le.s 1,$f22,$f20
      bc1t   1,then
# parte else: x = x-z;
else: l.s    $f24,z
      sub.s  $f20,$f20,$f24
      s.s    $f20,x
      b      end
# parte then: x = x+z;
then: l.s    $f24,z
      add.s  $f20,$f20,$f24
      s.s    $f20,x
end:
```

Se pide:

a) Programar la solución en un fichero llamado **if_else_float.asm (hay que crear el fichero, no existe, y hay que reservar en él espacio para las variables residentes en memoria)**.
b) Realizar varias trazas de ejecución del programa hasta llegar a la etiqueta **end**, anotando los valores de los registros y variables involucrados en **todas** las instrucciones (descontando la E/S y la secuencia de terminación del programa). Anotar también el valor del flag 1.

Guardar las trazas realizadas en el fichero **trazas2.xlsx**.

Juegos de datos de entrada para realizar las trazas:

Datos	Juego 1	Juego 2
x	2.0	-3.5
y	0.0	6.3
z	6.0	3.5

EJERCICIO 5: RAÍZ CUADRADA CON DATOS EN COMA FLOTANTE

Se pretende realizar la raíz cuadrada de un número Z según el método de Newton-Raphson, utilizando la siguiente fórmula de recurrencia:

$$x_{i+1} = 0{,}5 \cdot \left(x_i + \frac{Z}{x_i}\right)$$

El algoritmo en C sería el siguiente:

```
void entrada (void);
void salida (void);
float z, raiz_z, epsilon;
register float xi, xi_1;
int main (void) {
   entrada();
   xi_1 = 1; /* Primera aproximacion a la solucion */
   do {
      xi = xi_1;
      xi_1 = 0.5*(xi-z/xi);
   } while (fabs(xi_1-xi) >= epsilon);
   raiz_z = xi_1; /* Valor final del resultado */
   salida();
   return 0;
}
```

El fichero **raíz_cuadrada.asm** contiene un esqueleto de la solución.

Se pide:

a) Programar la solución.

b) Realizar varias trazas de ejecución del programa, escribiendo los valores de los registros, las variables y del flag de coma flotante involucrados en los siguientes puntos:
   1. Justo antes de entrar en el bucle (una única vez).
   2. En la ramificación condicional correspondiente a la evaluación de la condición del bucle **do-while** (una vez en cada iteración).
   3. En la instrucción siguiente a la anterior (una vez en cada iteración).
   4. En la instrucción siguiente al bucle (una única vez).

Se pide realizar varias trazas del programa pedido. obviando la parte de E/S por consola. Incluir las trazas en el fichero **trazas2.xlsx**.

Juegos de datos de entrada para realizar las trazas:

Datos	Juego 1	Juego 2
Z	9.0	3.0
epsilon	1e-5	1e-6