Estructura de los Computadores RAÚL BELTRÁN MARCO 23900664F

Memoria

**Índice**

1. PRÁCTICA 7 (p.2)
2. PRÁCTICA 8 (p.-)
3. PRÁCTICA 9 (p.-)
4. Ejercicios optativos (p.-)

- Optativos Práctica 7 (p.-)

- Optativos Práctica 8 (p.-)

- Optativos Práctica 9 (p.-)

DATOS

🡪 Nombre: Raúl Beltrán Marco

🡪 Email: rbm61@alu.ua.es/raulbeltmarc@gmail.com

🡪 Grupo: lunes /03/19

Estructura de los Computadores RAÚL BELTRÁN MARCO 23900664F

PRÁCTICA 7

Vectores

Entregable 7\_1

**Calcula la suma de los valores positivos y negativos del vector, dirección del cual se pasa como parámetro en $a0 y la longitud en $a1. La función devuelve en $v0 la suma de los valores positivos y en $v1 la suma de los negativos.**

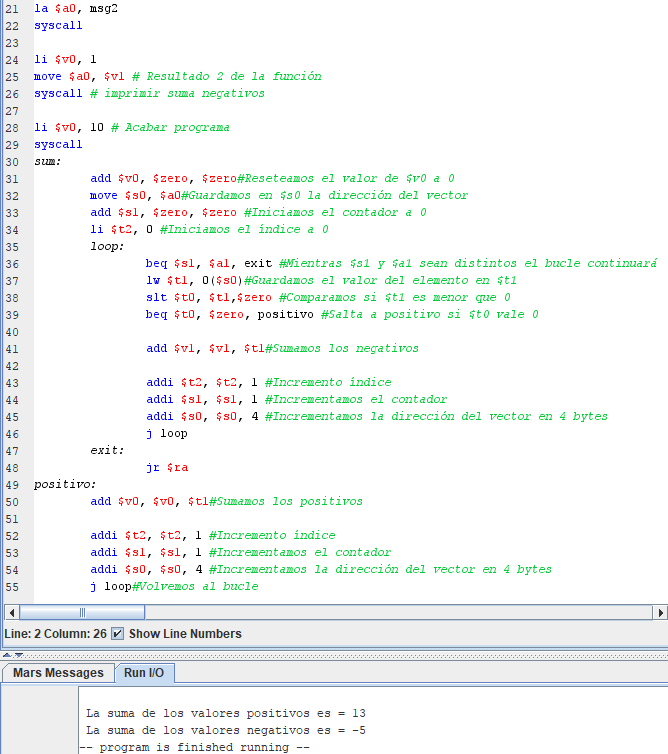
Para poder realizar este ejercicio hemos tenido que implementar la función sum en la cual hemos realizado los siguientes pasos:

**Primero** hemos inicializado a 0 el contador($s1) y el índice ($t2) los cuales usaremos para recorrer el bucle, además de **inicializar a 0 el valor de $v0** y guardar en $s0 la dirección del vector para evitar problemas.

**Dentro del bucle**, hacemos uso de la beq para comparar el valor de nuestro contador y la longitud del bucle($a1), después guardamos en $t1 el valor del vector leído con lw, para después usar slt para **comprobar si es positivo** el valor de $t1.

Si es **positivo** ($t0=0) entonces saltará a “positivo” donde se sumarán los valores positivos. Tanto al final de “loop” como de “positivo”, aumentaremos en 1 el contador y el índice además de aumentar en 4 la dirección del vector para leer el siguiente valor.

Una vez **fuera del bucle** usaremos jr $ra para volver al programa principal

****

Estructura de los Computadores RAÚL BELTRÁN MARCO 23900664F

Entregable 7\_2

Estructura de los Computadores RAÚL BELTRÁN MARCO 23900664F

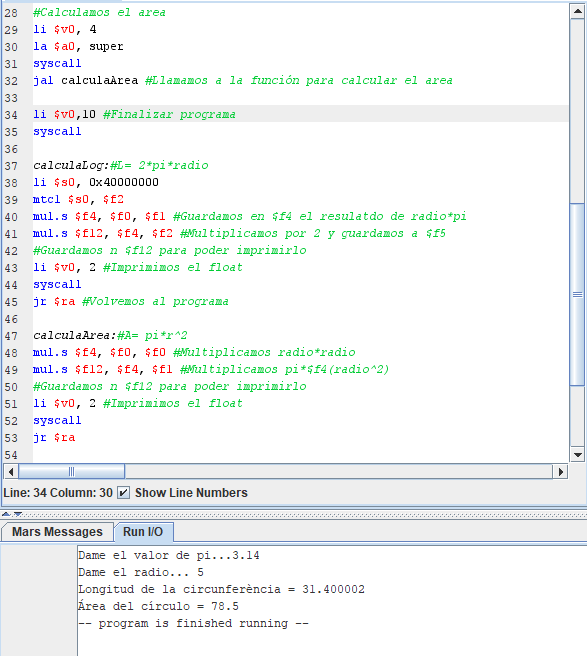
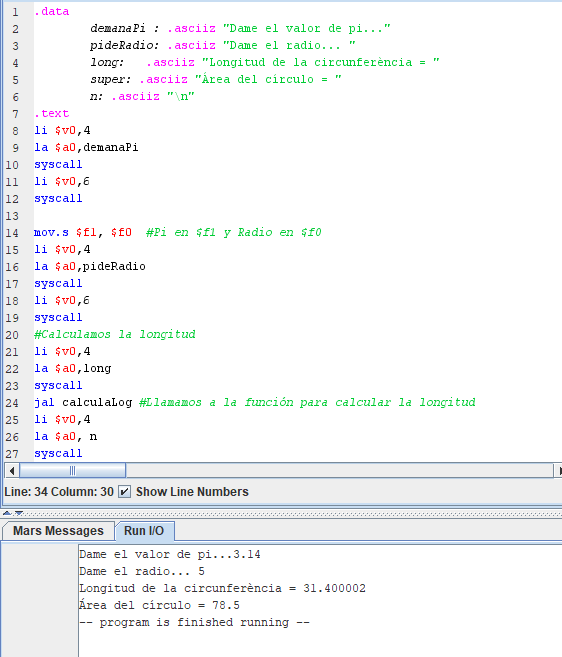
PRÁCTICA 8

Coma flotante 1

Entregable 8\_1

**Completa el siguiente código de partida que pide el radio por teclado y tiene que calcular y mostrar en la consola la longitud de la circunferencia y el área del círculo.**

Para poder realizar este ejercicio entregable hemos tenido que implementar dos funciones llamadas calculaLog y calculaArea. Como podemos observar en la imagen de la derecha, ambas funciones se han implementado usando mul.s para multiplicar entre floats. Especial mención en calculaLog, debido que para implementar el valor de 2 para poder aplicar la fórmula de la longitud hemos usado la herramienta de Mips para obtener 0x40000000(podríamos a ver usador las funciones mtc1 y cvt.s.w como podremos ver en el entregable 8\_2) y poder pasarlo a flotante

****

Estructura de los Computadores RAÚL BELTRÁN MARCO 23900664F

Entregable 8\_2

**Haz el código que suma los elementos del vector y calcula el valor medio en coma flotante. Muestra el resultado por la consola**

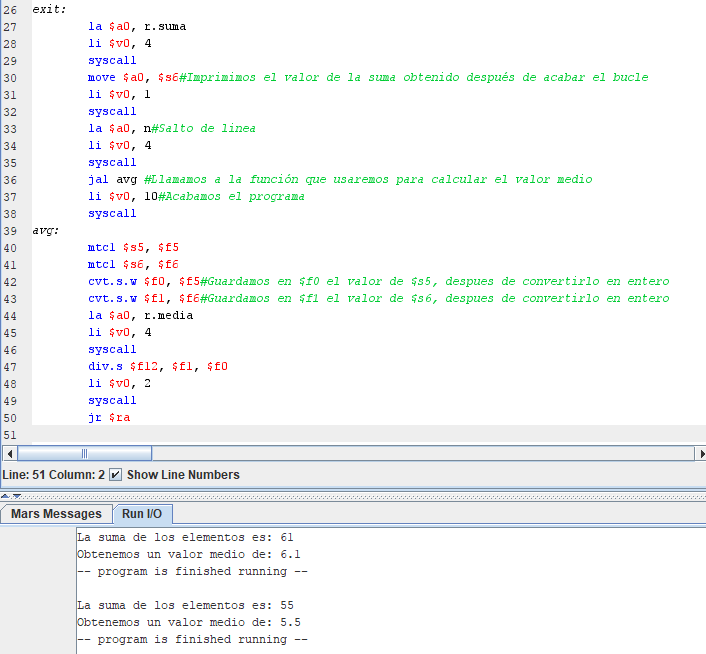
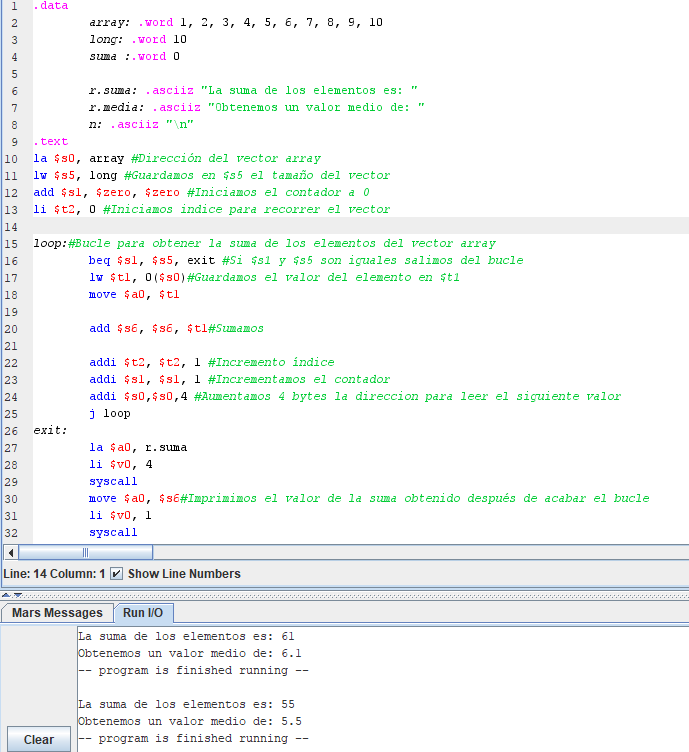
Para poder realizar este ejercicio, (a parte de los .asciiz para mostrar mensajes) hemos tenido que realizar varios pasos.

El **primer paso** ha sido guardar la dirección del vector array dado además de su tamaño haciendo uso de lw. Después hemos inicializado a 0 el contador de nuestro bucle “loop” y el índice que usaremos para recorrer el vector.

Una vez **dentro del bucle**, hacemos uso de beq, para comprobar que el contador($s1) no es igual al tamaño del vector($s5), mientras sean diferentes el bucle continuará. Mientras realizamos el bucle guardamos en $s6 el valor de la suma que se va acumulando, después aumentamos en 1 el valor del contador y del índice, además de, **importante**, aumentar en 4 bytes la dirección de array($s0) para poder leer la siguiente posición

Una vez **fuera del bucle** mostramos el resultado obtenido en $s6 y llamamos a la función avg, donde calculamos el valor medio.

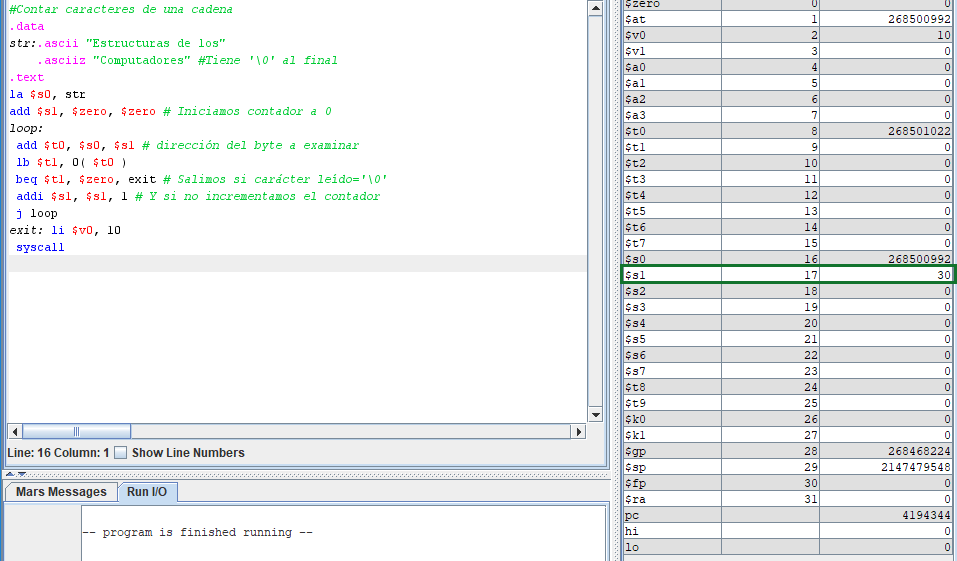
En la **función avg**, sencillamente implementaremos las reglas mtc1 y cvt.s.w para poder “mover” a Coproc 1(líneas 40-43) el valor de $s5 y $s6 para realizar la operación tipo float: div.s para dividir la suma total ($s6🡪$f1) entre el total de elementos ($s5🡪$f0) para luego mostrar el resultado obtenido en $f12 usando un syscall



Estructura de los Computadores RAÚL BELTRÁN MARCO 23900664F

Ejercicios optativos Práctica 7

**Ensambla y ejecuta el código. ¿Cuántos caracteres tiene la cadena?**



* Como podemos observar tiene 30 caracteres

**¿En qué dirección de la memoria se encuentra el carácter null)?**

* En 1e

**¿Cómo se actualiza el índice del vector?**

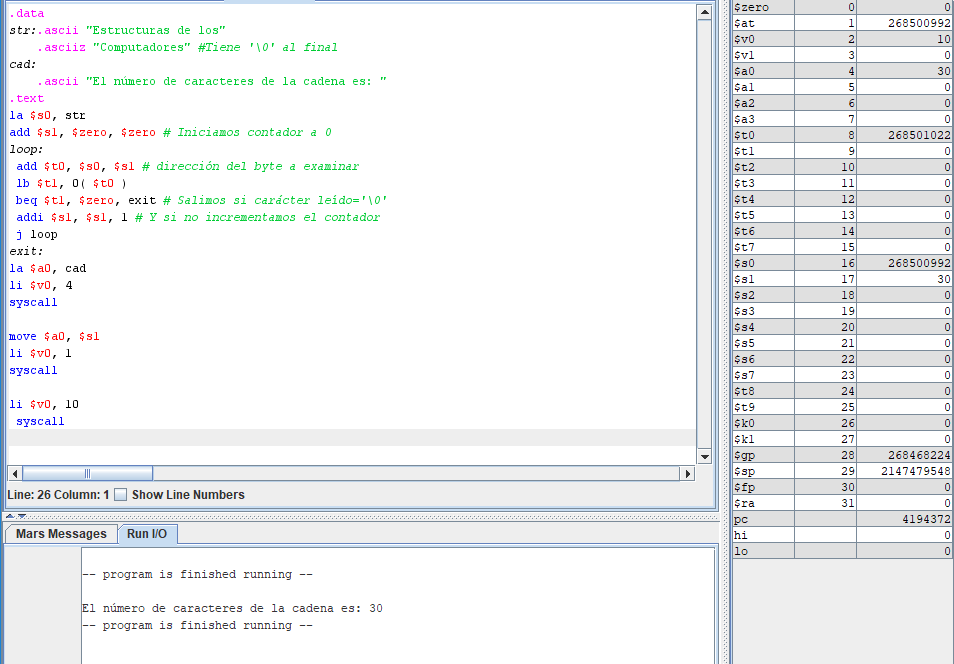
* En $s1 y usando la instrucción addi $s1, $s1, 1 para ir aumentando de 1 en 1

**¿El programa funcionaría si la cadena solo constara del carácter null?**

* Sí que funcionaria y $s1 valdría 0

Estructura de los Computadores RAÚL BELTRÁN MARCO 23900664F

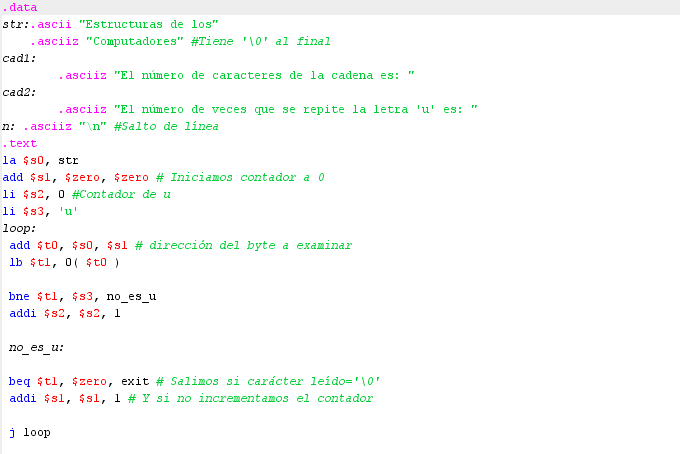
**Modifica el código para que muestre por pantalla el mensaje “El número de caracteres de la cadena es:” y a continuación el resultado.**

****

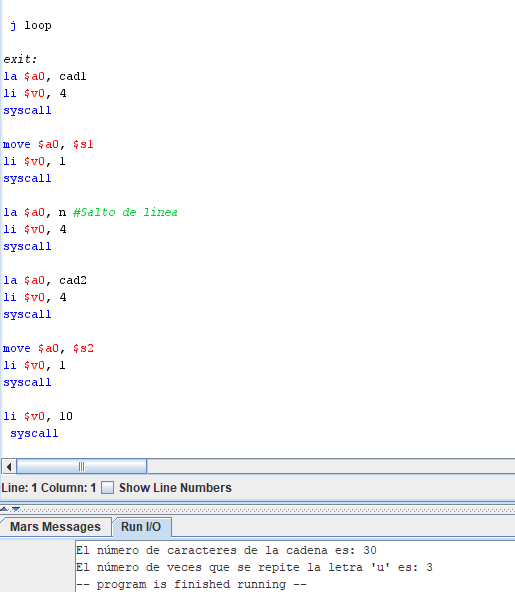
Estructura de los Computadores RAÚL BELTRÁN MARCO 23900664F

**Modifica el código para que calcule el número de veces que se repite la vocal “u”.**

Parte del .data y el bucle

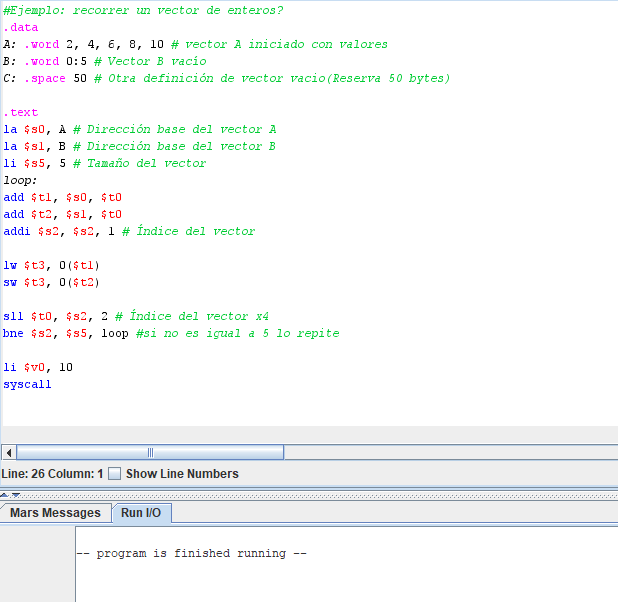
****

Parte del código que se activa cuando acaba el bucle (exit)



Estructura de los Computadores RAÚL BELTRÁN MARCO 23900664F

**Ensambla y ejecuta el programa. Comprueba que el vector A se copia en el vector B.**



**¿En qué dirección empieza el vector B?**

* En value (+14)

**¿Porque no se pueden acabar los vectores con el carácter null igual que se hace con las cadenas de caracteres?**

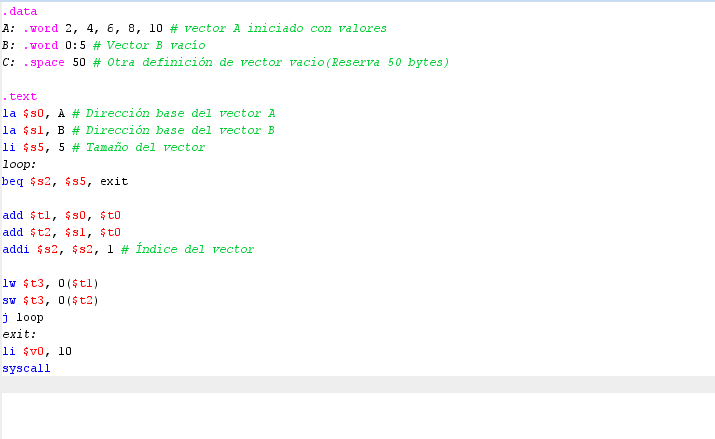
* Los vectores no son caracteres, no tienen que ser bytes

Estructura de los Computadores RAÚL BELTRÁN MARCO 23900664F

**En el programa se recorren los vectores actualizando el índice con la instrucción sll. ¿De qué otra manera se podrían recorrer los vectores?**

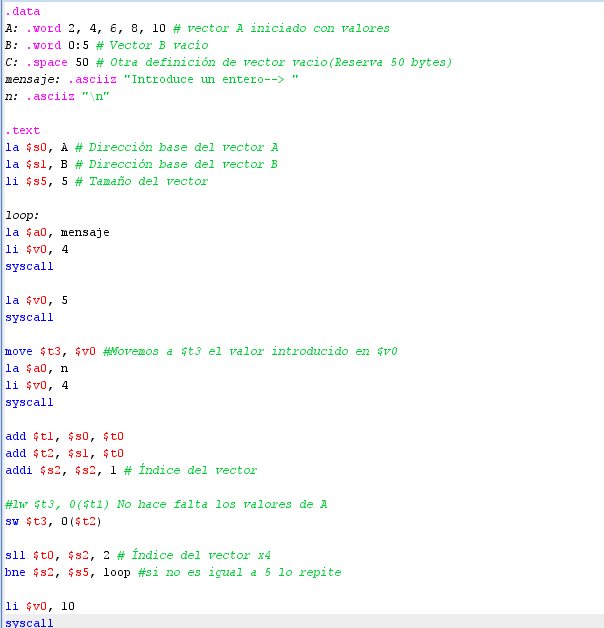
* Sumando de 4 en 4

**Se ha utilizado un bucle del tipo do-while, modifica el programa por que el bucle sea de tipo for-while.**

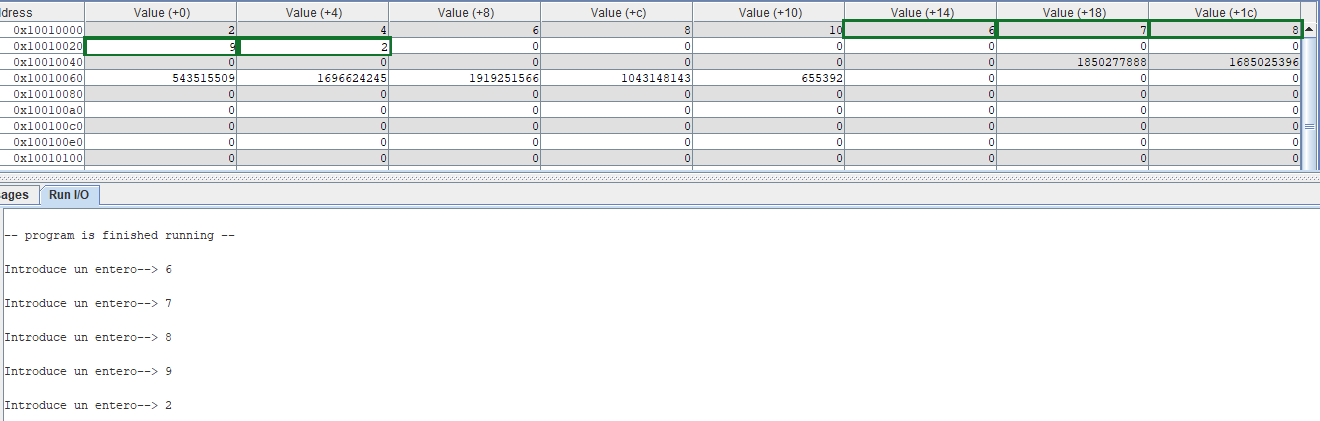
****

Estructura de los Computadores RAÚL BELTRÁN MARCO 23900664F

**Modifica el programa para que el vector B se rellene con enteros leídos del teclado. Previamente se tiene que mostrar un mensaje por consola que pida los elementos.**

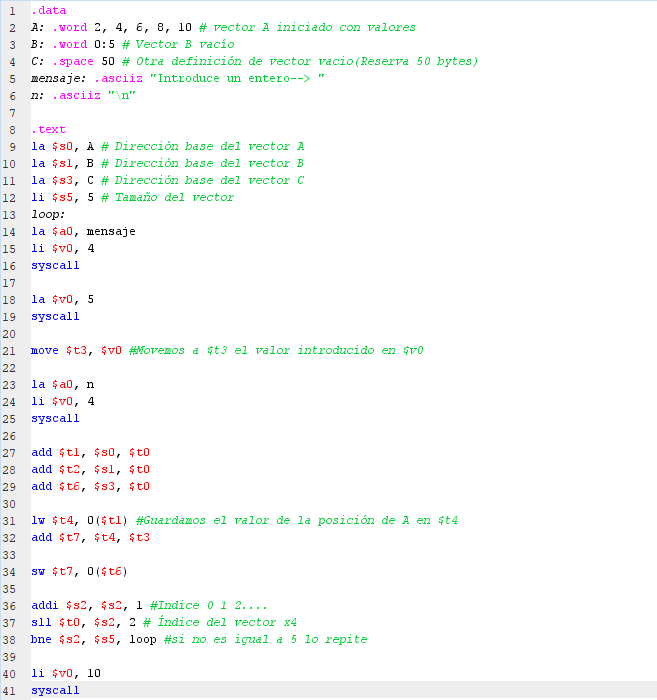


Como podemos observar se guardan bien los datos:

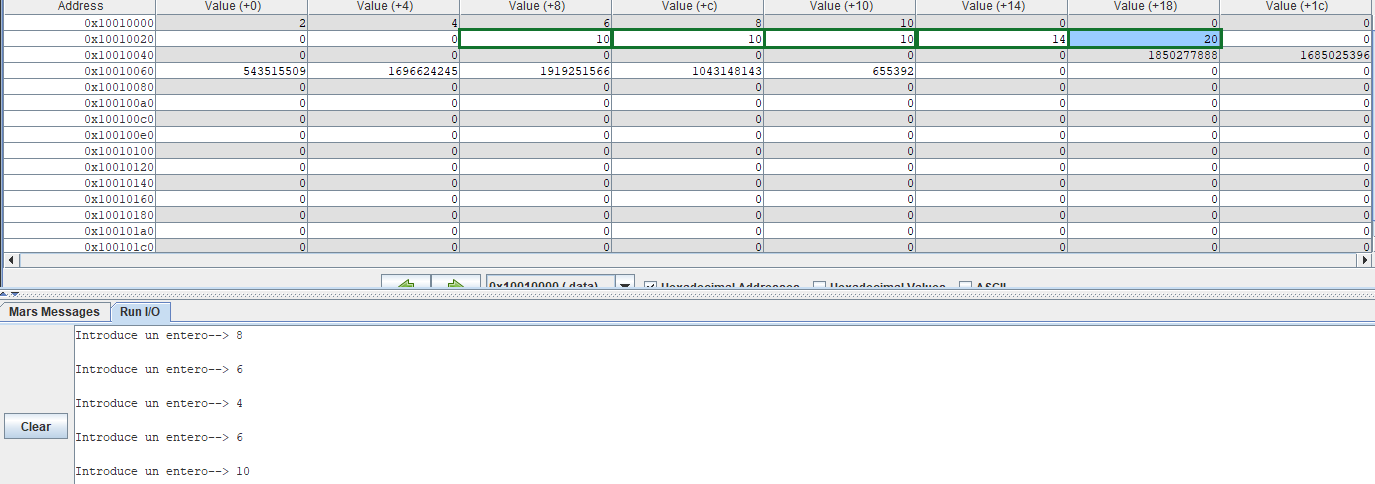


Estructura de los Computadores RAÚL BELTRÁN MARCO 23900664F

**Completa el programa para que se rellene el vector C con la suma de los elementos del vector A y del B (C[y]=A[y]+B[y]).?**

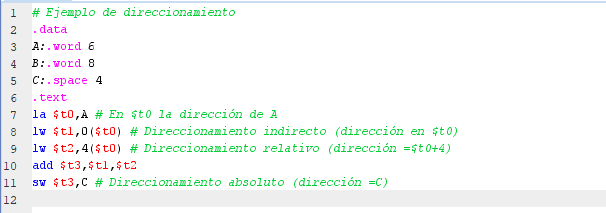
****

Como podemos observar hemos obtenido los resultados esperados



Estructura de los Computadores RAÚL BELTRÁN MARCO 23900664F

**En el siguiente ejemplo es muestra la utilización de los distintos modos de direccionamiento:**



**¿Cuántas pseudoinstrucciones contiene el código?**

* Hay dos pseudoinstrucciones la y sw

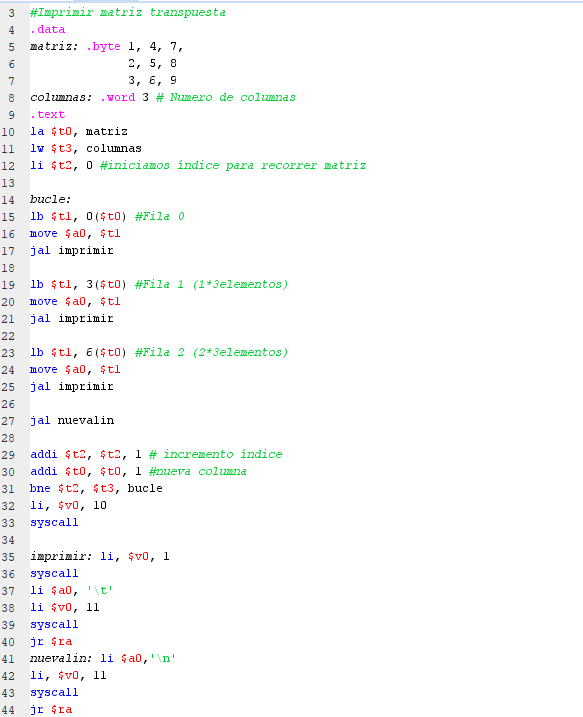
**Ensambla el código y observa la traducción de las pseudoinstrucciones en instrucciones del MIPS. ¿En qué instrucciones se ha traducido sw $t3, C?**

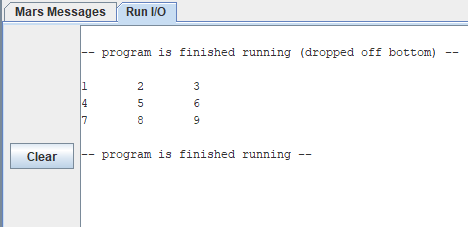
* En lui y sw

**¿Qué registro auxiliar se ha utilizado?**

* El $at

Estructura de los Computadores RAÚL BELTRÁN MARCO 23900664F





**Identifica los distintos modos de direccionamiento en memoria utilizados: direccionamiento absoluto, direccionamiento indirecto y direccionamiento**

Absoluto: lw $t3, columnas

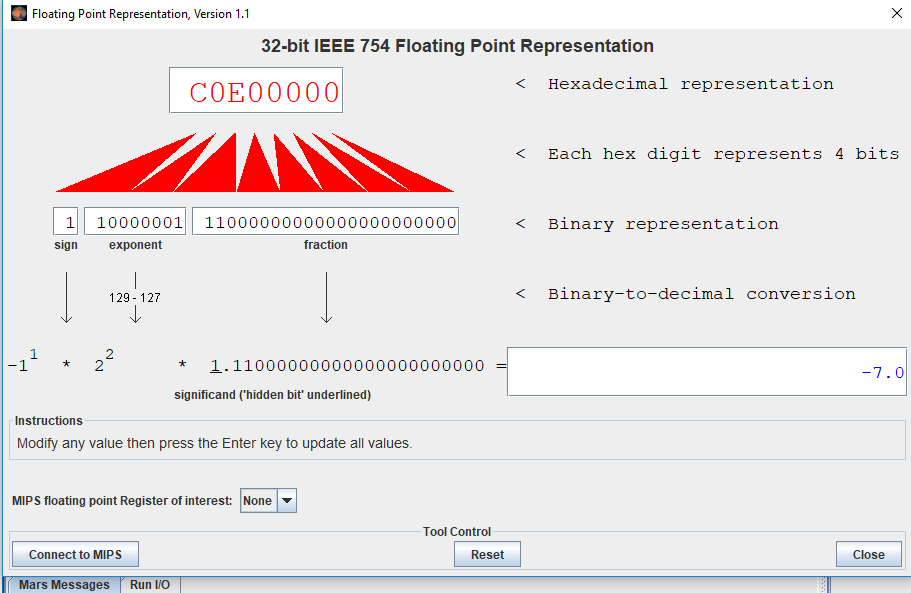
Indirecto: lb $t1, 0($t0)

Relativo: lb $t1, 3($t0) y lb $t1, 6($t0)

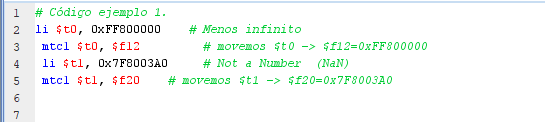
Estructura de los Computadores RAÚL BELTRÁN MARCO 23900664F

Ejercicios optativos Práctica 8

**Comprobad que el número decimal de C0E00000 es -7.0 y que la representación en 32 bits de simple precisión de 4.25 es 40880000**

****

**EJEMPLO1**

****

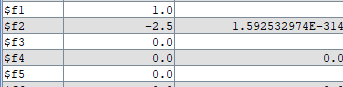
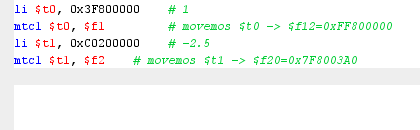
**Haciendo uso de la herramienta de representación en coma flotante del MARS,**

**Comprueba que realmente en $f12 hay un y en $f20 un NaN según el estándar IEEE 754.**

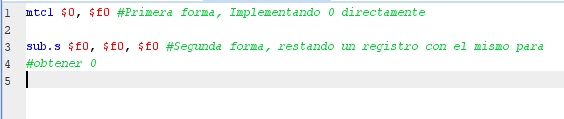
* 0xff800000:
* 0x7f8003a0: NaN

Estructura de los Computadores RAÚL BELTRÁN MARCO 23900664F

**Haz que el contenido de $f1 sea el valor 1 en coma flotante y el de $f2 el valor 2.5 en coma flotante.**

****

**Di una manera de escribir un 0.0 en $f0 con sólo una instrucción máquina**

****

**¿Por qué no hay registros HI y LO para guardar el resultado de la multiplicación y división en coma flotante del mismo modo que con los números enteros?**

Porque las multiplicaciones realizadas en IE^3 conservan el formato IE^3, en cambio cuando se multiplica sin este formato devuelve un resultado que es el doble de tamaño

Estructura de los Computadores RAÚL BELTRÁN MARCO 23900664F

**EJEMPLO2**

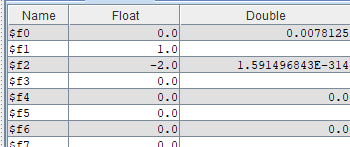
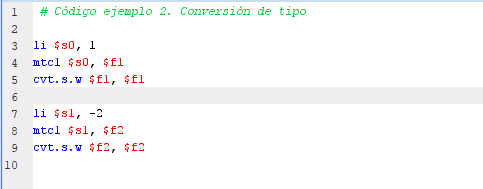
**Haciendo uso de la herramienta de representación en coma flotante del MARS, comprueba que realmente en $f0 después de la conversión hay un -8.**

* 0xc1000000: -8.0

**¿Qué valor consideraría la máquina que habría en $f0 si no hiciésemos la conversión con la instrucción cvt.s.w? Aprovecha que puedes ver contenidos en decimal y en hexadecimal.**

* Aparecería un NaN

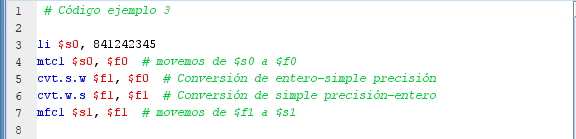
**Haz que el contenido de $f1 sea el valor 1 en coma flotante y el de $f2 el valor -2 en coma flotante utilizando las instrucciones de conversión de tipo.**

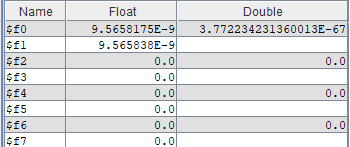
****

Estructura de los Computadores RAÚL BELTRÁN MARCO 23900664F

**EJEMPLO3**

**Ensambla el código y comprueba el resultado.**

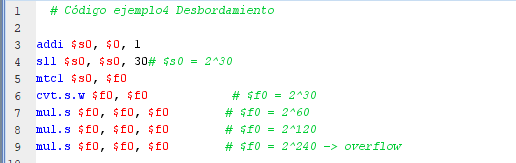
****

****

**¿Cuál es la razón por la que al finalizar el programa los contenidos de $s0 y $s1 son distintos?**

**🡪** Debido al redondeo que aparece cuando hacemos la conversión

**EJEMPLO4**



**¿Qué valor representa en el formato IEEE 754 el contenido final de $f0?**

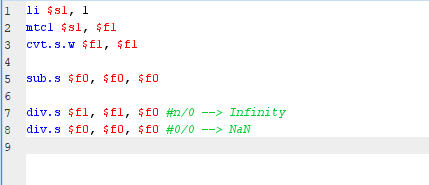
* Infinity

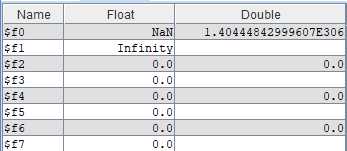
Estructura de los Computadores RAÚL BELTRÁN MARCO 23900664F

**EJEMPLO4**

**Observa que no ha ocurrido ninguna excepción a la ejecutar el código. ¿Ocurre lo mismo al dividir por cero? ¿Y al hacer 0/0?**

**Añade instrucciones al código anterior y haz la prueba.**

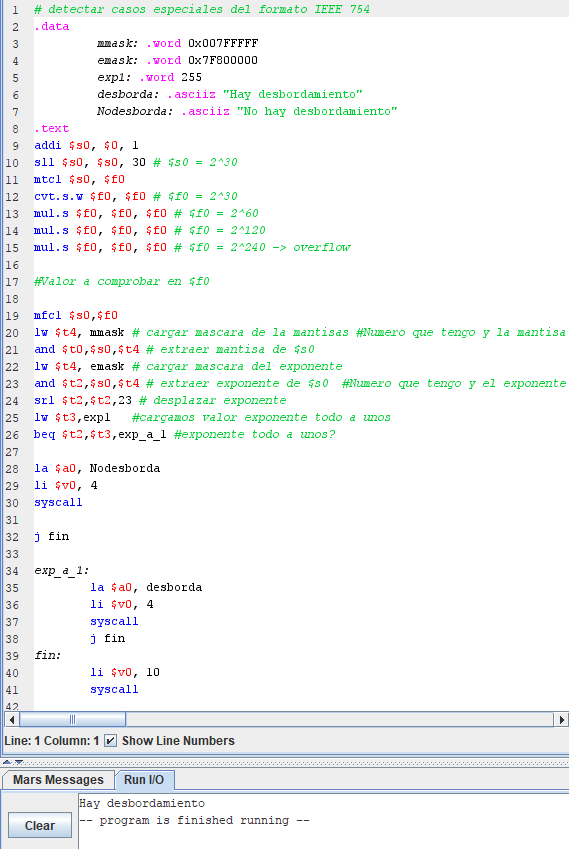
****

****

Estructura de los Computadores RAÚL BELTRÁN MARCO 23900664F

**EJEMPLO5**

**Completa el código para que muestre en consola un mensaje de error por desbordamiento. Comprueba que funciona correctamente.**

****