1. Parte 1: Ensamblador MIPS y simulador MARS

Antes de ejecutar el código

1. Estudiar las directivas e instrucciones que aparecen en el código. ¿Cuantas directivas tiene el código?

   Son 10 directivas en total.

2. ¿Cuáles son los valores de Stack Pointer ($sp), el Global Pointer ($gp) y el Program Counter ($pc) ?

MIPS Register	gp (hexadecima)	pc (hexadecimal)	sp (hexadecimal)
Value	0x10008000	0x00400000	0x7fffeffc

3. Rellenar la tabla que se muestra a continuación.

Data Segment		**Text Segment**	
Name	**Address**	**Name**	**Address**
Valor	0x10010000	main	0x00400000
Carac	0x10010014		
Id	0x10010020		

4. ¿Numero de instrucciones escritas del programa y numero de instrucciones generadas por MARS?

   - Numero de instrucciones escritas del programa: 13.
   - Numero de instrucciones generadas por MARS: 22.

5. Identificar la instrucción número 24 del programa y diga en cuantas instrucciones de lenguaje de máquina se transformo

   La instrucción 24, se transforma en 3 instrucciones en lenguaje de maquina

6. ¿Cuántos bytes de memoria ocupan los datos del programa?

   El programa ocupa 36 bytes de memoria.

7. ¿Cuál es el contenido de la dirección de memoria 0x10010018?

Carácter	P		I		M		,	
Hexadecimal	5	0	4	9	4	d	2	c

   EL MIPS por defecto esta en Little Endian.

8. Identifica la instrucción 31 de su programa. Escriba los contenidos de la columnas Address, Code y Basic

Address	Code	Basic
0x0040002c	0x7296a802	mul $21, $20 , $22

La conversión de la columna code en Binario:

HEXADE	7				2				9				6				a				8				0				2			
BINARY	0	1	1	1	0	0	1	0	1	0	0	1	0	1	1	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	1	0

Después de ejecutar el código

1. Llene la tabla mostrando el cambio de los registros.

REGISTER		VALUE	
NAME	NUMBER	HEXADECIMAL	DECIMAL
$at	1	0x1001000c	268501004
$v0	2	0x0000000a	10
$s0	16	0x00000008	8
$s1	17	0x00000009	9
$s2	18	0x0000000a	10
$s3	19	0x0000000b	11
$s4	20	0x00000003	3
$s5	21	0x0000000c	12
$s6	22	0x00000004	4
pc		0x00400058	4194392
lo		0x0000000c	12

2. La memoria de datos ha sufrido algún cambio?. Muestre el estado del conjunto de direcciones de memoria.

Memory Address	Contenido
0x10010000	0x00000008
0x10010004	0x00000009
0x10010008	0x0000000a
0x1001000c	0x0000000b
0x10010010	0x000a0b1a
0x10010014	0x616c6f48
0x10010018	0x50494d2c
0x1001001c	0x00000053
0x10010020	0x00000001

3. **Describe lo que hace el programa:** Se crea un arreglo de 4 elementos, luego de creado el programador extrae cada elemento del arreglo y lo guarda en registro distintos.

4. En qué dirección se detiene el arreglo: 0x00400054

Modificación del codigo

1. Reemplace en el código las instrucciones 28,31 y 34 (mul $s5,$s4,$s6) por sll $s5,$s4,2 y vuelva a ensamblar el programa y vea que pasa. Ejecute de nuevo el programa. Que ha cambiado?.

   En el programa cambia el registro de 0x00000000 a 0x0000000c, mientras que al usar el shift el registro no lo cambia.

2. Identificar alguna instrucción que sobre o no cumpla alguna función. Identifíquela y diga que hace?

La instrucción **addi $s6,$zero,4**, puesto que con el shift no es necesario para el registro con el valor 4.

2. Parte 2: Servicios de entrada y salida

Ensamble y Ejecute

1. Fíjese en la ventana de mensajes, la pestaña Run I/O y escriba las salidas que se van obteniendo.

   Al correr el programa con la opción "popup dialog for input syscalls", Primero te imprime el siguiente mensaje "Esto es una cadena", después te imprime el número "99"; lo siguiente es que aparece una ventana de dialogo para que se ingrese un integer y luego otra ventana para ingresar una string de máximo 8 caracteres.

2. Diga cómo cambia la memoria de datos durante la ejecución

Memory Address	**Contenido antes ejecutar**	**Contenido después de ejecutar**
0x10010000	0x6f747345	0x616c6f48
0x10010004	0x20736520	0x2073000a
0x10010008	0x20616e75	0x20616e75
0x1001000c	0x65646163	0x65646163
0x10010010	0x000a616e	0x000a616e
0x10010014	0x00000063	0x00000063
0x10010018	0x0000000b	0x0000000b
0x1001001c	0x00000016	0x00000016
0x10010020	0x00000021	0x00000021
0x10010024	0x0000002c	0x0000002c
0x10010028	0x00000037	0x00000037
0x10010034	0x00000058	0x00000058
0x10010038	0x00000063	0x00000063

3. Modifique el código y diga cómo cambia la memoria de datos.

Memory Address	**Contenido antes ejecutar**	**Contenido después de ejecutar**
0x10010000	0x6f747345	0x6720656d
0x10010004	0x20736520	0x61747375
0x10010008	0x20616e75	0x6d656420
0x1001000c	0x65646163	0x61697361
0x10010010	0x000a616e	0x62206f64
0x10010014	0x00000063	0x616c6961
0x10010018	0x0000000b	0x00742072
0x1001001c	0x00000016	0x00000016
0x10010020	0x00000021	0x00000021
0x10010024	0x0000002c	0x0000002c
0x10010028	0x00000037	0x00000037
0x10010034	0x00000058	0x00000058
0x10010038	0x00000063	0x00000063