



**Universidad Autónoma de Baja California  
Facultad de Ingeniería Arquitectura y Diseño**

**Organización de Computadoras**

**Docente:**

**Crespo Regland Jonatan**

**Alumno:**

**Garcia Ceseña Edgar Rafael**

**GRUPO 932**

**Ingeniería en Software y Tecnologías Emergentes**

**TALLER 7**



1.

```
1 section .data
2 num1 db 5 ; Declaración de la variable num1 con valor 5
3 num2 db 11 ; Declaración de la variable num2 con valor 11
4 result db 0 ; Variable result inicializada en 0 para almacenar el resultado de la
5 suma
6 msg db 'Resultado: ', 0 ; Mensaje que se mostrará antes del resultado
7 section .bss
8 buffer resb 4 ; Reserva un espacio de 4 bytes en el buffer para almacenar el
9 carácter ASCII del resultado
10 section .text
11 global _start
12 _start:
13 ; Realizar la suma de num1 y num2
14 mov al, [num1] ; Cargar el valor de num1 en el registro AL
15 add al, [num2] ; Sumar el valor de num2 al valor en AL
16 mov [result], al ; Guardar el resultado de la suma en la variable result
17 ; Convertir el resultado numérico en su representación ASCII
18 movzx eax, byte [result] ; Cargar el valor de result en EAX y expandirlo a 32 bits
19 add eax, 48 ; Sumar 48 para convertir el número en su carácter ASCII
20 correspondiente ('0' = 48)
21 mov [buffer], al ; Almacenar el carácter ASCII en el buffer
22 ; Mostrar el mensaje "Resultado: "
23 mov eax, 4 ; Llamada al sistema para escribir (sys_write)
24 mov ebx, 1 ; File descriptor 1 (salida estándar)
25 mov ecx, msg ; Dirección del mensaje
26 mov edx, 11 ; Longitud del mensaje
27 int 0x80 ; Interrupción para ejecutar la llamada al sistema

28 ; Mostrar el resultado en ASCII
29
30 mov eax, 4 ; Llamada al sistema para escribir (sys_write)
31 mov ebx, 1 ; File descriptor 1 (salida estándar)
32 mov ecx, buffer ; Dirección del buffer que contiene el carácter ASCII del resultado
33 mov edx, 1 ; Longitud de 1 byte (un carácter)
34 int 0x80 ; Interrupción para ejecutar la llamada al sistema
35 ; Salir del programa
36 mov eax, 1 ; Llamada al sistema para salir (sys_exit)
37 xor ebx, ebx ; Código de salida 0
38 int 0x80 ; Interrupción para ejecutar la llamada al sistema
```

2.

```
1 section .data
2     num1 db 17           ; Valor que producirá A (65) al sumar con num2
3     num2 db 48           ; Ajuste de num2 para obtener los valores ASCII
4     msg db 'Resultado: ', 0
5
6 section .bss
7     buffer resb 4
8
9 section .text
10 global _start
11 _start:
12     ; Imprimir cada carácter modificando num1 para obtener el ASCII deseado
13
14     ; Obtener 'A'
15     mov al, [num1]       ; num1 = 17
16     add al, [num2]       ; num2 = 48, 17 + 48 = 65 ('A')
17     mov [buffer], al
18     call print_buffer
19
20     ; Obtener '\'
21     mov al, 44           ; num1 ajustado a 44 para '\'
22     add al, [num2]       ; 44 + 48 = 92 ('\')
23     mov [buffer], al
24     call print_buffer
25
26     ; Obtener '$'
27     mov al, -12          ; num1 ajustado a -12 para '$'
28     add al, [num2]       ; -12 + 48 = 36 ('$')
29     mov [buffer], al
30     call print_buffer
31
32     ; Obtener '&'
33     mov al, -10          ; num1 ajustado a -10 para '&'
34     add al, [num2]       ; -10 + 48 = 38 ('&')
35     mov [buffer], al
36     call print_buffer
37
38     ; Obtener '1'
39     mov al, 1            ; num1 ajustado a 1 para '1'
```

```

38 ; Obtener '1'
39 mov al, 1 ; num1 ajustado a 1 para '1'
40 add al, [num2] ; 1 + 48 = 49 ('1')
41 mov [buffer], al
42 call print_buffer
43
44 ; Salir del programa
45 mov eax, 1 ; sys_exit
46 xor ebx, ebx
47 int 0x80
48
49 print_buffer:
50 mov eax, 4 ; sys_write
51 mov ebx, 1 ; File descriptor (stdout)
52 mov ecx, buffer ; Dirección del buffer con el carácter ASCII
53 mov edx, 1 ; Longitud de 1 byte
54 int 0x80
55 ret ; Retorno para reutilizar la rutina
56

```

3.

```

1 section .data
2 msg db "Resultado: ", 0
3
4 section .bss
5 buffer resb 4
6
7 section .text
8 global _start
9 _start:
10 ; Cargar valor ASCII de '@' directamente
11 mov eax, 04 ; Carga inmediata del ASCII '@' (04)
12 mov [buffer], al ; Almacenar '@' en buffer
13
14 ; Imprimir el mensaje "Resultado: "
15 mov eax, 4
16 mov ebx, 1
17 mov ecx, msg
18 mov edx, 11
19 int 0x80
20
21 ; Imprimir '@'
22 mov eax, 4
23 mov ebx, 1
24 mov ecx, buffer
25 mov edx, 1
26 int 0x80
27
28 ; Salir del programa
29 mov eax, 1
30 xor ebx, ebx
31 int 0x80

```

4.

```
1 section .data
2     symbol dd 64          ; ASCII '@'
3     msg db 'Resultado: ', 0
4
5 section .bss
6     buffer resb 4
7
8 section .text
9 global _start
10 _start:
11     ; Cargar el valor de '@' indirectamente
12     mov eax, [symbol]      ; Acceso indirecto al valor ASCII '@' desde la dirección de symbol
13     mov [buffer], al       ; Almacenar '@' en buffer
14
15     ; Imprimir el mensaje "Resultado: "
16     mov eax, 4
17     mov ebx, 1
18     mov ecx, msg
19     mov edx, 11
20     int 0x80
21
22     ; Imprimir '@'
23     mov eax, 4
24     mov ebx, 1
25     mov ecx, buffer
26     mov edx, 1
27     int 0x80
28
29     ; Salir del programa
30     mov eax, 1
31     xor ebx, ebx
32     int 0x80
```