

# Universidad Autónoma de Baja California



*Facultad de Ingeniería, Arquitectura y Diseño*

*Ingeniería en Software*

*Organización de Computadoras*

Taller 10 GIT HUB

*Brayan Arturo Rocha Meneses*

**Ensenada Baja California Noviembre de 2024**



Assembly ▾




```
1 section .data
2 num1 db 5 ; Define el primer número, 5, almacenado en una variable de 1 byte
3 num2 db 11 ; Define el segundo número, 11, almacenado en una variable de 1 byte
4 result db 0 ; Variable para almacenar el resultado de la suma, inicialmente en 0
5 message db "Resultado: ", 0 ; Mensaje a mostrar antes del resultado, seguido de un
6 terminador null
7 section .bss
8 buffer resb 4 ; Reserva un buffer de 4 bytes en la sección .bss para almacenar datos
9 temporales
10 section .text
11 global _start ; Define la etiqueta _start como el punto de inicio del programa
12 ; Macro para imprimir una cadena
13 %macro PRINT_STRING 1
14 mov eax, 4 ; Llamada al sistema para escribir (syscall número 4 en Linux)
15 mov ebx, 1 ; Descriptor de archivo 1 (stdout) para la salida estándar
16 mov ecx, %1 ; La dirección de la cadena que se pasará como argumento a la macro
17 mov edx, 13 ; Longitud de la cadena que se va a imprimir
18 int 0x80 ; Llama a la interrupción 0x80 para ejecutar la llamada al sistema
19 %endmacro
20 ; Macro para imprimir un número
21 %macro PRINT_NUMBER 1
22 mov eax, %1 ; Carga el número que se desea imprimir en el registro EAX
23 add eax, '0' ; Convierte el valor numérico a su equivalente en ASCII
24 mov [buffer], eax ; Almacena el valor ASCII en el buffer
```



Assembly ▾




```
22 mov eax, %1 ; Carga el número que se desea imprimir en el registro EAX
23 add eax, '0' ; Convierte el valor numérico a su equivalente en ASCII
24 mov [buffer], eax ; Almacena el valor ASCII en el buffer
25 mov eax, 4 ; Llamada al sistema para escribir
26 mov ebx, 1 ; Descriptor de archivo 1 (stdout) para la salida estándar
27 mov ecx, buffer ; Dirección del buffer que contiene el número en formato ASCII
28 mov edx, 1 ; Longitud del dato a imprimir (1 byte)
29 int 0x80 ; Llama a la interrupción 0x80 para ejecutar la llamada al sistema
30 %endmacro
31 _start:
32 ; Realiza la suma de los valores en num1 y num2
33 mov al, [num1] ; Carga el valor de num1 en el registro AL
34 add al, [num2] ; Suma el valor de num2 al valor en AL
35 mov [result], al ; Almacena el resultado de la suma en la variable result
36 ; Imprime el mensaje de texto "Resultado: "
37 PRINT_STRING message ; Llama a la macro PRINT_STRING para imprimir el mensaje
38 ; Imprime el resultado de la suma
39 PRINT_NUMBER [result] ; Llama a la macro PRINT_NUMBER para imprimir el valor almacenado
40 en result
41 ; Salir del programa
42 mov eax, 1 ; Llamada al sistema para salir del programa (syscall número 1 en Linux)
43 mov ebx, 0 ; Código de salida 0 (sin errores)
44 int 0x80 ; Llama a la interrupción 0x80 para ejecutar la salida del programa
45
```

 Assembly ▾



```
1 section .data
2 message db "La suma de los valores es: ", 0 ; Mensaje inicial para mostrar
3 newline db 10, 0 ; Nueva línea para la salida
4 section .bss
5 buffer resb 4 ; Buffer para convertir números a caracteres
6 section .text
7 global _start
8 %macro DEFINE_VALUES 3
9 ; Define una "estructura" con tres valores
10 val1 db %1 ; Primer valor
11 val2 db %2 ; Segundo valor
12 val3 db %3 ; Tercer valor
13 %endmacro
14 %macro PRINT_STRING 1
15 ; Macro para imprimir una cadena de caracteres
16 mov eax, 4 ; Syscall número para 'write'
17 mov ebx, 1 ; File descriptor para stdout
18 mov ecx, %1 ; Dirección del mensaje
19 mov edx, 25 ; Longitud del mensaje
20 int 0x80 ; Ejecuta la syscall
21 %endmacro
22 %macro PRINT_NUMBER 1
23 ; Convierte un número en eax a caracteres ASCII y lo imprime
24 mov eax, %1 ; Carga el número a imprimir en eax
```

 Assembly ▾



```
24 mov eax, %1 ; Carga el número a imprimir en eax
25 mov ecx, buffer + 3 ; Apunta al final del buffer
26 mov ebx, 10 ; Divisor para obtener dígitos decimales
27 .next_digit:
28 xor edx, edx ; Limpia edx para la división
29 div ebx ; Divide eax entre 10, cociente en eax, residuo en edx
30 add dl, '0' ; Convierte el dígito a ASCII
31 dec ecx ; Mueve hacia atrás en el buffer
32 mov [ecx], dl ; Almacena el dígito en el buffer
33 test eax, eax ; Verifica si quedan dígitos
34 jnz .next_digit ; Si quedan dígitos, continúa
35 ; Calcula la longitud del número en el buffer
36 mov edx, buffer + 4 ; Posición final del buffer
37 sub edx, ecx ; Calcula la longitud real del número
38 ; Imprime el número
39 mov eax, 4 ; Syscall para write
40 mov ebx, 1 ; Salida estándar
41 mov ecx, ecx ; Dirección inicial en el buffer
42 int 0x80 ; Ejecuta la syscall
43 %endmacro
44 %macro PRINT_SUM 0
45 ; Realiza la suma de tres valores y la imprime
46 mov al, [val1] ; Carga el primer valor en AL
47 add al, [val2] ; Suma el segundo valor
```



Assembly ▾



```
41 mov ecx, ecx ; Dirección inicial en el buffer
42 int 0x80 ; Ejecuta la syscall
43 %endmacro
44 %macro PRINT_SUM 0
45 ; Realiza la suma de tres valores y la imprime
46 mov al, [val1] ; Carga el primer valor en AL
47 add al, [val2] ; Suma el segundo valor
48 add al, [val3] ; Suma el tercer valor
49 movzx eax, al ; Expande AL a EAX para asegurar un valor de 32 bits
50 ; Imprime el resultado de la suma
51 PRINT_NUMBER eax
52 PRINT_STRING newline
53 %endmacro
54 ; Definimos los tres valores con la macro DEFINE_VALUES
55 DEFINE_VALUES 3, 5, 7
56 _start:
57 ; Imprime el mensaje inicial
58 PRINT_STRING message
59 ; Imprime la suma de los valores
60 PRINT_SUM
61 ; Salir del programa
62 mov eax, 1 ; Syscall para 'exit'
63 mov ebx, 0 ; Código de salida
64 int 0x80 ; Ejecuta la syscall para salir del program
```