

Ing. Francisco Rene Ardón Guerra

ARQUITECTURA DE COMPUTADORES Y ENSAMBLADORES 1

Practica de Assembler

## **Calculadora de Assembler**

**Nombre:** Eruyn Andrés Morales Morales

**Carnet:** 202046116

**Carrera:** Ing. Sistemas

**Chiquimula, 21 de diciembre del 2,022**

## Introducción

El proyecto de clase consta de la programación de una calculadora que por medio de un menú evaluará las operaciones básicas:

Cómo lo son suma resta, multiplicación y división además de la potencia.

La calculadora consta de un menú a elección en la cual se evaluarán 2 números y se mostrará el resultado en la consola. El resultado nunca será de más de 2 dígitos es decir no veremos resultados (123) pero sí se podrán ingresar valores de 2 dígitos es decir  $12 \times 03 = 36$

El proyecto debe de ser enviado en un repositorio de GitHub el día 21 de diciembre del 2022 y será testeada en las páginas de emulador que hemos visto en clase.

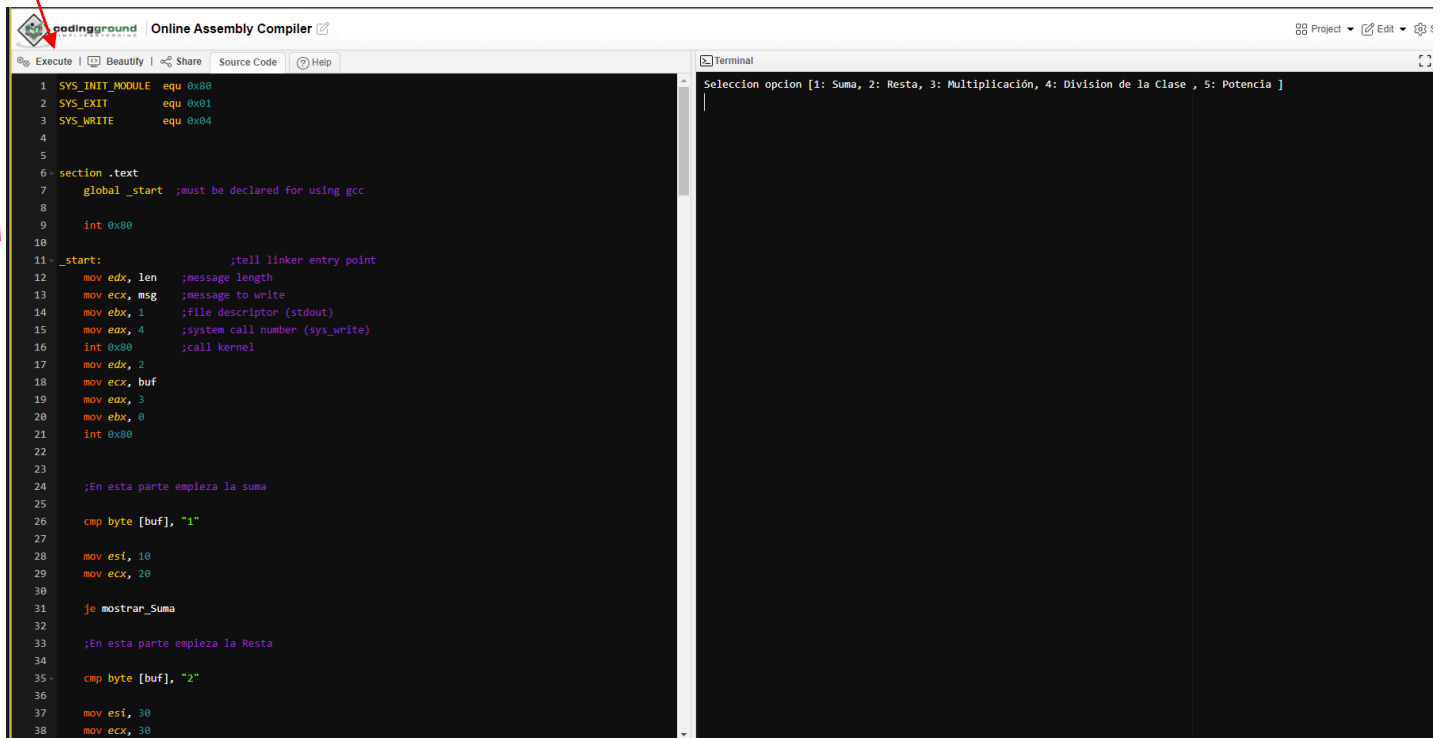
Todos los métodos de operaciones básicas ya se encuentran en este repositorio por lo que se deben de juntar las operaciones en un solo bloque de código integrado.

Nota importante en el repositorio de GitHub que deben entregar se deberá agregar un documento llamado README.md en este se agregará la información de cómo funciona el proyecto con imágenes de ejemplos, (esta documentación debe estar escrita en inglés.)

# Documentación

Para poder utilizar esta calculadora de Assembler se necesita hacer algunos pasos correctamente, entre ellos son:

**Primer paso:** Para poder iniciar la calculadora, primero debemos de copiar el código que les dejaré en GitHub como Proyecto.asm, luego de haberlo copiado en la página para compilarlo solamente hay que darle click donde dice **Execute** y el Código empezará a funcionar correctamente.



The screenshot shows the 'Online Assembly Compiler' interface. A red arrow points to the 'Execute' button in the top toolbar. Another red arrow points to the source code area, which contains the following assembly code:

```
1 SYS_INIT_MODULE equ 0x80
2 SYS_EXIT equ 0x01
3 SYS_WRITE equ 0x04
4
5
6 section .text
7     global _start ;must be declared for using gcc
8
9     int 0x80
10
11 _start:
12     mov edx, len ;message length
13     mov ecx, msg ;message to write
14     mov ebx, 1 ;file descriptor (stdout)
15     mov eax, 4 ;system call number (sys_write)
16     int 0x80 ;call kernel
17     mov edx, 2
18     mov ecx, buf
19     mov eax, 3
20     mov ebx, 0
21     int 0x80
22
23
24 ;En esta parte empieza la suma
25
26 cmp byte [buf], "1"
27
28 mov esi, 10
29 mov ecx, 20
30
31 je mostrar_Suma
32
33 ;En esta parte empieza la Resta
34
35 cmp byte [buf], "2"
36
37 mov esi, 30
38 mov ecx, 30
```

A third red arrow points to the terminal output, which displays the prompt: 'Selección opción [1: Suma, 2: Resta, 3: Multiplicación, 4: División de la Clase, 5: Potencia]'.

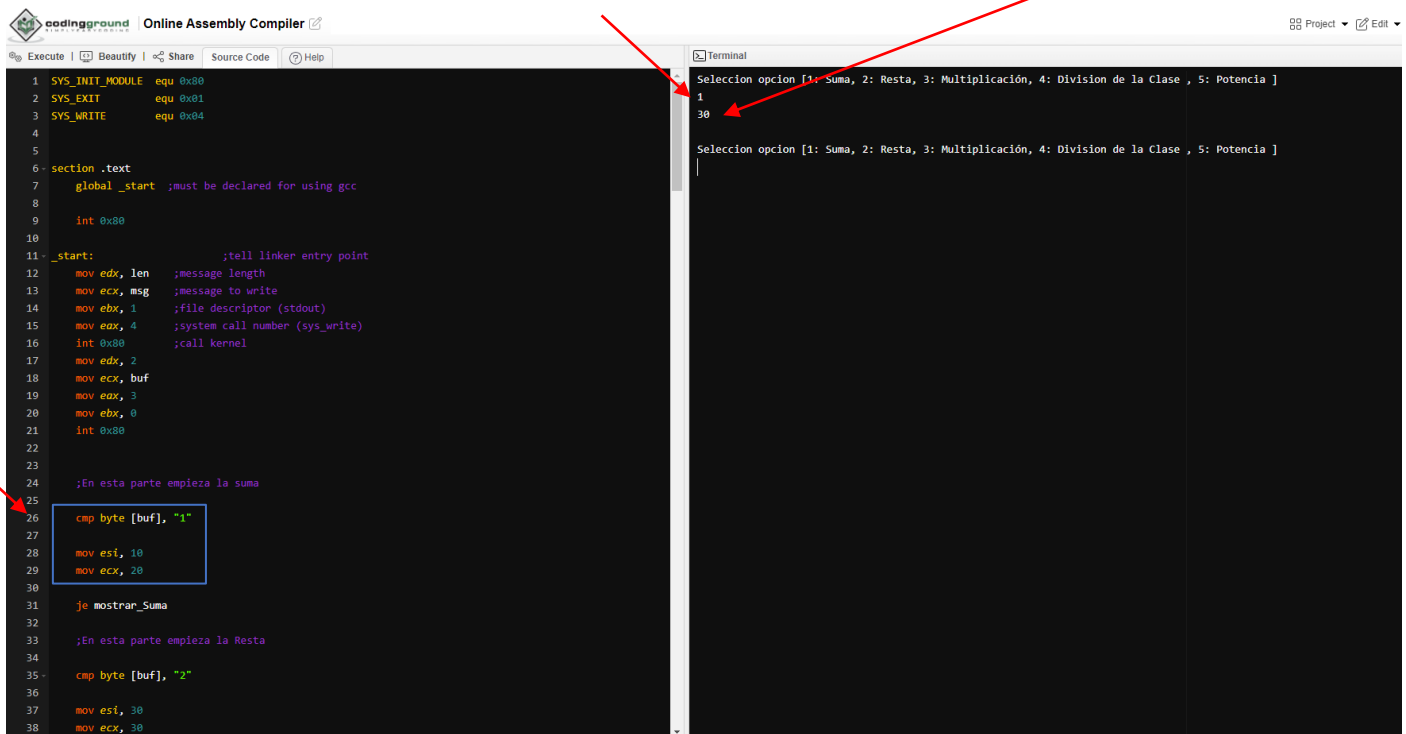
**Nota:** Si en dado caso la Página se queda cargando o no responde es necesario actualizar la página y volver a copiarlo para que vuelva a funcionar, ya que la página sugiere actualizarse para poder correr los cambios.

**Segundo Paso:** En este paso ya al terminar de compilarlo procedemos a ver la función de nuestro programa que es una Calculadora; para poder operar como primero es la suma en la parte derecha colocamos **La Opción No.1**, y el programa dará el Resultado de la suma.

## Ejemplo:

En este caso la Suma es de  $10 + 20 = 30$ .

Para poder cambiar los valores solamente los cambiamos en la parte izquierda en el recuadro azul en la parte de los registros.



The screenshot shows the Online Assembly Compiler interface. On the left, the assembly code is displayed with line numbers 1 through 38. A blue box highlights the code at lines 26-29: `cmp byte [buf], "1"`, `mov esi, 10`, and `mov ecx, 20`. A red arrow points to this box. On the right, the terminal window shows the output of the program. It displays the prompt "Selección opción [1: Suma, 2: Resta, 3: Multiplicación, 4: División de la Clase, 5: Potencia ]" and the user input "1". Below this, it shows the result "30". A red arrow points to the "30" in the terminal output.

```
1 SYS_INIT_MODULE equ 0x00
2 SYS_EXIT equ 0x01
3 SYS_WRITE equ 0x04
4
5
6 section .text
7 global _start ;must be declared for using gcc
8
9 int 0x00
10
11 _start: ;tell linker entry point
12 mov edx, len ;message length
13 mov ecx, msg ;message to write
14 mov ebx, 1 ;file descriptor (stdout)
15 mov eax, 4 ;system call number (sys_write)
16 int 0x80 ;call kernel
17 mov edx, 2
18 mov ecx, buf
19 mov eax, 3
20 mov ebx, 0
21 int 0x80
22
23
24 ;En esta parte empieza la suma
25
26 cmp byte [buf], "1"
27
28 mov esi, 10
29 mov ecx, 20
30
31 je mostrar_Suma
32
33 ;En esta parte empieza la Resta
34
35 cmp byte [buf], "2"
36
37 mov esi, 30
38 mov ecx, 30
```

Terminal Output:

```
Selección opción [1: Suma, 2: Resta, 3: Multiplicación, 4: División de la Clase, 5: Potencia ]
1
30
Selección opción [1: Suma, 2: Resta, 3: Multiplicación, 4: División de la Clase, 5: Potencia ]
```

**Nota:** Para poder hacer una suma sin problemas no colocar valores que den números de 3 cifras; ya que la suma mayor que se puede hacer es  $49 + 49 = 98$  o  $98 + 1 = 99$ .

**Tercer Paso:** Para poder operar como segunda parte es la resta en la parte derecha colocamos **La Opción No.2**, y el programa dará el Resultado de la resta.

## Ejemplo:

En este caso la Resta es de  $50 - 30 = 20$ .

Para poder cambiar los valores solamente los cambiamos en la parte izquierda en el recuadro azul en la parte de los registros.

The screenshot shows the 'Online Assembly Compiler' interface. On the left, the assembly code is displayed with line numbers 23 to 61. A blue box highlights the code for the subtraction operation: `cmp byte [buf], "2"` (line 35), `mov esi, 50` (line 37), and `mov ecx, 30` (line 38). Red arrows point from the text above to these lines. On the right, the terminal window shows the program's output: `Seleccion opcion [1: Suma, 2: Resta, 3: Multiplicación, 4: Division de la Clase , 5: Potencia ]`, followed by the input `2` and the result `20`. Another red arrow points from the text above to the `20` output in the terminal.

```
23
24 ;En esta parte empieza la suma
25
26 cmp byte [buf], "1"
27
28 mov esi, 10
29 mov ecx, 20
30
31 je mostrar_Suma
32
33 ;En esta parte empieza la Resta
34
35 cmp byte [buf], "2"
36
37 mov esi, 50
38 mov ecx, 30
39
40
41 je mostrar_Resta
42
43 ;En esta parte empieza la Multiplicación
44
45 cmp byte [buf], "3"
46
47 je mostrar_Multi
48
49 ;En esta parte empieza la División
50
51 cmp byte [buf], "4"
52
53 je mostrar_Div
54
55 ;En esta parte empieza la Potencia
56
57
58 cmp byte [buf], "5"
59
60 mov esi, 1
61 mov ecx, 4
```

Terminal Output:

```
Seleccion opcion [1: Suma, 2: Resta, 3: Multiplicación, 4: Division de la Clase , 5: Potencia ]
2
20
Seleccion opcion [1: Suma, 2: Resta, 3: Multiplicación, 4: Division de la Clase , 5: Potencia ]
```

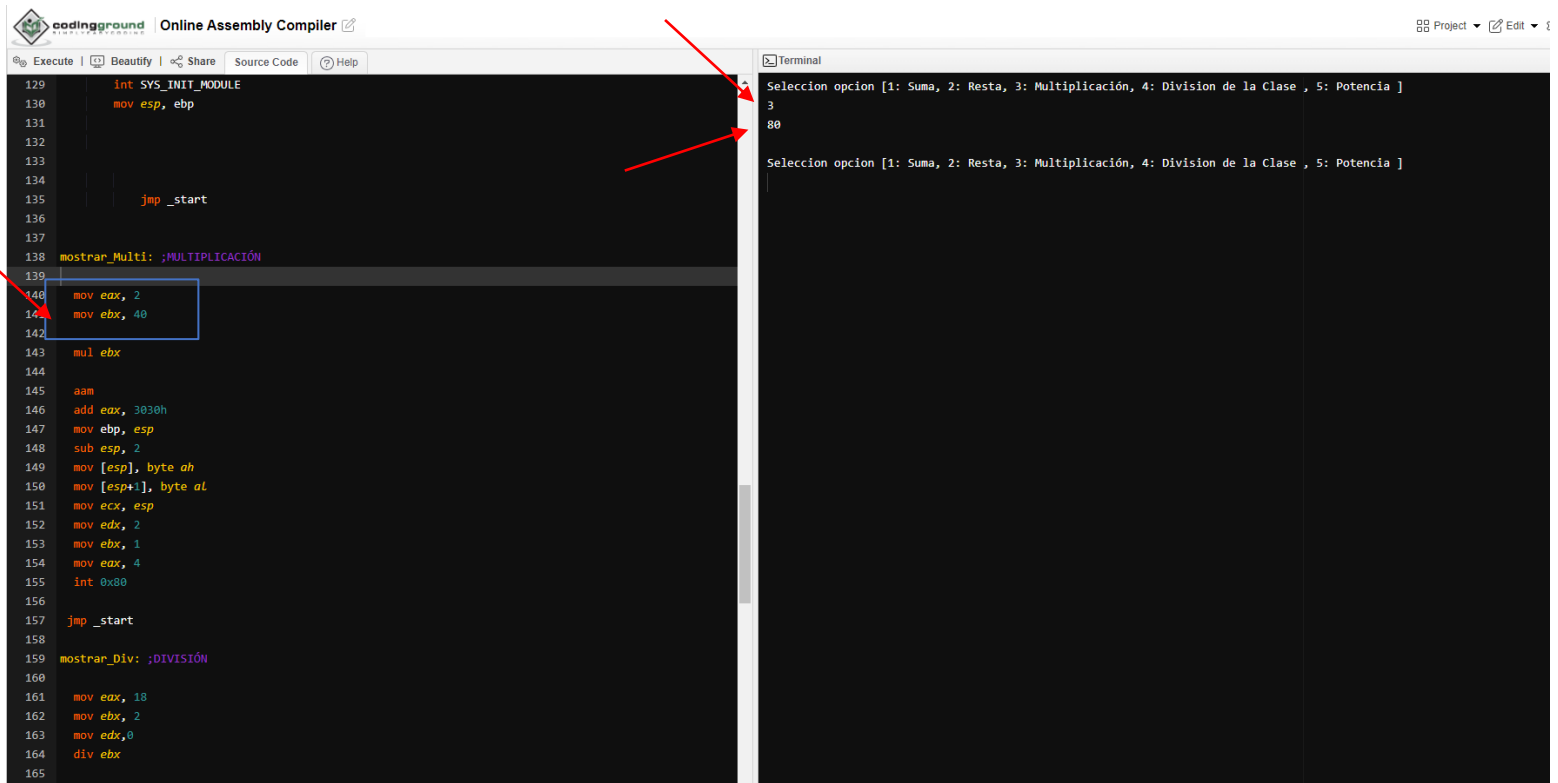
**Nota:** Para poder hacer una resta sin problemas, hacer restas exactas a modo de que nunca den números negativos para que no afecte el funcionamiento del programa y restas solo de 2 cifras ya que el programa no soporta dar el resultado de 3 cifras.

**Cuarto Paso:** Para poder operar como tercera parte es la multiplicación en la parte derecha colocamos **La Opción No.3**, y el programa dará el Resultado de la multiplicación.

## Ejemplo:

En este caso la Multiplicación es de  $2 \times 40 = 80$ .

Para poder cambiar los valores solamente los cambiamos en la parte izquierda en el recuadro azul en la parte de los registros, en las líneas de código 140 y 141 para que el usuario encuentre bien los registros a modificar.



The screenshot displays the Online Assembly Compiler interface. The left pane shows assembly code with line numbers 129 to 165. Lines 140 and 141 are highlighted with a blue box, containing the instructions `mov eax, 2` and `mov ebx, 40`. Red arrows point from these lines to the terminal output. The right pane shows the terminal output with the prompt "Seleccion opcion [1: Suma, 2: Resta, 3: Multiplicación, 4: Division de la Clase , 5: Potencia ]", the input "3", and the result "80".

```
129 int SYS_INIT_MODULE
130 mov esp, ebp
131
132
133
134
135 jmp _start
136
137
138 mostrar_Multi: ;MULTIPlicACIÓN
139
140 mov eax, 2
141 mov ebx, 40
142
143 mul ebx
144
145 aam
146 add eax, 3030h
147 mov ebp, esp
148 sub esp, 2
149 mov [esp], byte ah
150 mov [esp+1], byte al
151 mov ecx, esp
152 mov edx, 2
153 mov ebx, 1
154 mov eax, 4
155 int 0x80
156
157 jmp _start
158
159 mostrar_Div: ;DIVISIóN
160
161 mov eax, 18
162 mov ebx, 2
163 mov edx, 0
164 div ebx
165
```

Terminal Output:

```
Seleccion opcion [1: Suma, 2: Resta, 3: Multiplicación, 4: Division de la Clase , 5: Potencia ]
3
80
Seleccion opcion [1: Suma, 2: Resta, 3: Multiplicación, 4: Division de la Clase , 5: Potencia ]
```

**Nota:** Para poder hacer una multiplicación sin problemas, colocar números que den resultados menores a 100, porque el programa solo da resultados menores a 100.

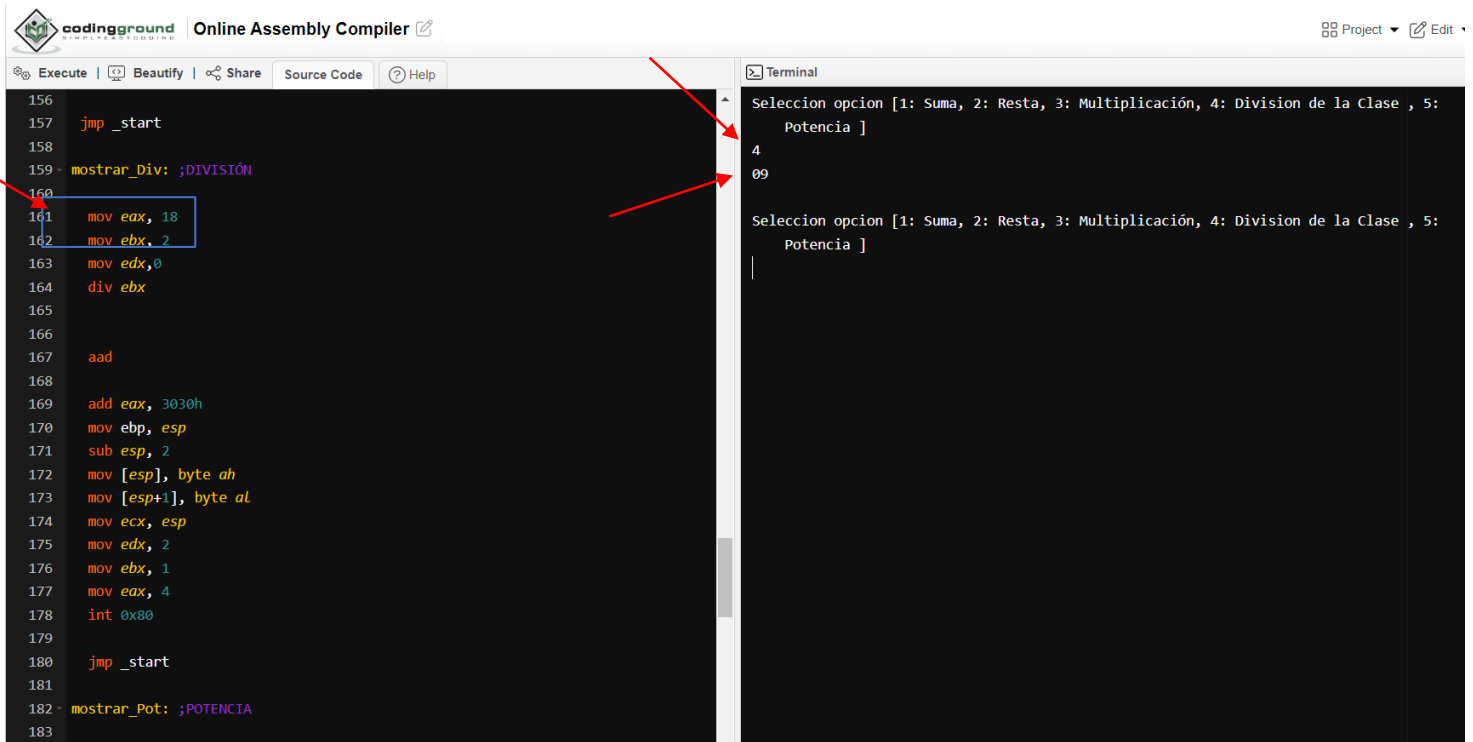
Para que no de errores el programa acepta multiplicaciones correctamente desde la tabla del 1 al 10, menos  $10 \times 10 = 100$  porque dará un resultado que el programa no reconoce porque no es capaz de resolverlo.

**Quinto Paso:** Para poder operar como cuarta parte es la división en la parte derecha colocamos **La Opción No.4**, y el programa dará el Resultado de la división.

## Ejemplo:

En este caso la División es de  $18 / 2 = 09$ .

Para poder cambiar los valores solamente los cambiamos en la parte izquierda en el recuadro azul en la parte de los registros, que serían las líneas 161 y 162 de código.



The screenshot shows the 'Online Assembly Compiler' interface. On the left, the assembly code is displayed with line numbers 156 to 183. Lines 161 and 162 are highlighted with a blue box: `mov eax, 18` and `mov ebx, 2`. Red arrows point from these lines to the terminal output. The terminal on the right shows the program's execution: it prompts for an option, the user enters '4', and the program outputs '09'. The terminal text is:   
Seleccion opcion [1: Suma, 2: Resta, 3: Multiplicación, 4: Division de la Clase , 5: Potencia ]  
4  
09  
Seleccion opcion [1: Suma, 2: Resta, 3: Multiplicación, 4: Division de la Clase , 5: Potencia ]

**Nota Importante:** En esta parte la división solo hace divisiones pequeñas, es decir, el programa solo acepta los resultados o el cociente de una división que sea menor o igual a 09, si ingresamos una division como por ejemplo,  $20/2=10$  el resultado lo mostrará en Código Ascii por una extraña razón que desconozco.

El código es el mismo que usamos en clase, solamente que lo modifique y no fue necesario de ingresar el que había hecho por mi cuenta. Sin embargo, el programa solo hará divisiones donde el resultado sea menor o igual a 9.

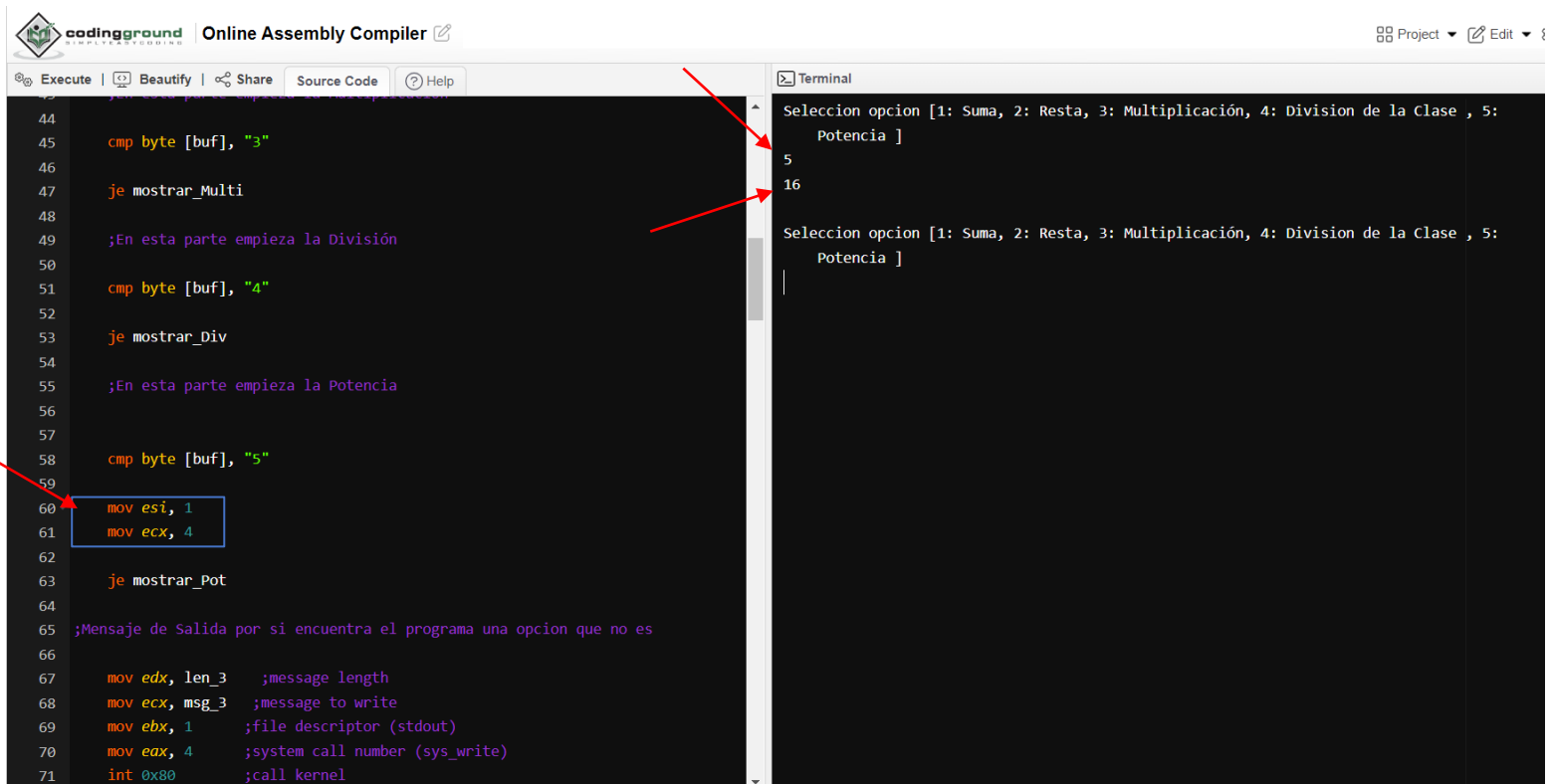
Una de las divisiones como por ejemplo que pueda ser aceptada es  $6/3=2$ , dejo esta nota para que no haya problemas al momento de ingresar los datos 😊.

**Sexto Paso:** Para poder operar como quinta parte es la Potencia en la parte derecha colocamos **La Opción No.5**, y el programa dará el Resultado de la Potencia.

## Ejemplo:

En este ejemplo puse la potencia de 4 que es 16.

Para poder cambiar los valores solamente los cambiamos en la parte izquierda en el recuadro azul en la parte de los registros.



The screenshot shows the Online Assembly Compiler interface. The left pane displays assembly code with line numbers 44 to 71. The right pane shows the terminal output. Red arrows indicate the relationship between the code and the output: one arrow points from the `cmp byte [buf], "3"` instruction to the first prompt in the terminal, another from `cmp byte [buf], "4"` to the second prompt, and a third from the highlighted `mov esi, 1` and `mov ecx, 4` instructions to the result '16'.

```
44  
45     cmp byte [buf], "3"  
46  
47     je mostrar_Multi  
48  
49     ;En esta parte empieza la División  
50  
51     cmp byte [buf], "4"  
52  
53     je mostrar_Div  
54  
55     ;En esta parte empieza la Potencia  
56  
57  
58     cmp byte [buf], "5"  
59  
60     mov esi, 1  
61     mov ecx, 4  
62  
63     je mostrar_Pot  
64  
65     ;Mensaje de Salida por si encuentra el programa una opcion que no es  
66  
67     mov edx, len_3    ;message length  
68     mov ecx, msg_3    ;message to write  
69     mov ebx, 1        ;file descriptor (stdout)  
70     mov eax, 4        ;system call number (sys_write)  
71     int 0x80         ;call kernel
```

Terminal Output:

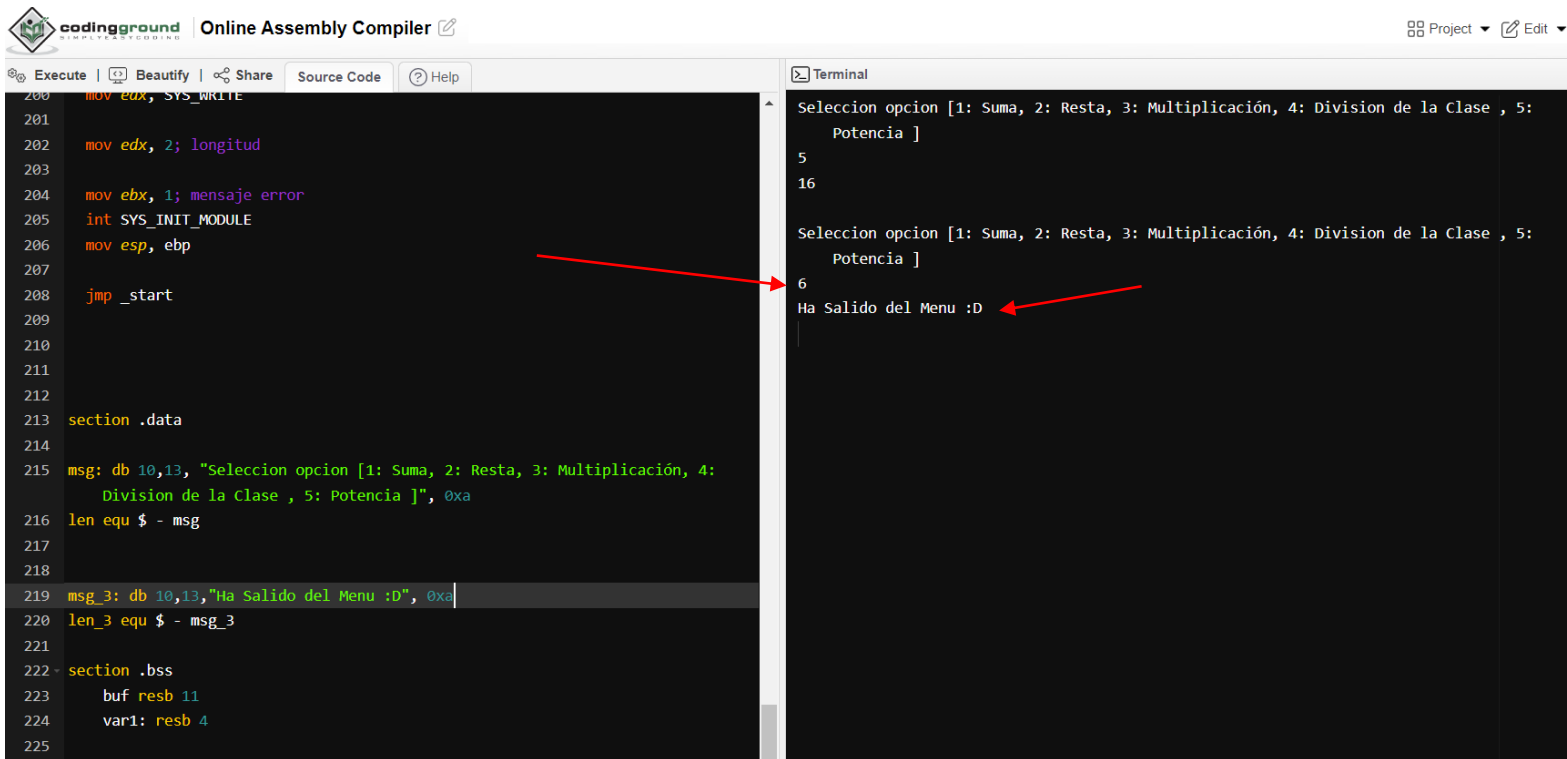
```
Seleccion opcion [1: Suma, 2: Resta, 3: Multiplicación, 4: Division de la Clase , 5:  
Potencia ]  
5  
16  
Seleccion opcion [1: Suma, 2: Resta, 3: Multiplicación, 4: Division de la Clase , 5:  
Potencia ]
```



**Séptimo Paso:** Ya para finalizar el programa presionamos cualquier número que no sea del 1 al 5, para que no seleccione ninguna de las opciones establecidas dentro del programa.

**Ejemplo:**

**En este caso presionaré el 6 para que me finalice el programa.**



The screenshot displays the Online Assembly Compiler interface. The left pane shows assembly code with line numbers 200 to 225. The right pane shows the terminal output. A red arrow points from the `jmp _start` instruction (line 208) to the terminal output. Another red arrow points from the `msg_3` definition (line 219) to the terminal output.

```
200 mov eax, SYS_WRITE
201
202 mov edx, 2; longitud
203
204 mov ebx, 1; mensaje error
205 int SYS_INIT_MODULE
206 mov esp, ebp
207
208 jmp _start
209
210
211
212
213 section .data
214
215 msg: db 10,13, "Seleccion opcion [1: Suma, 2: Resta, 3: Multiplicación, 4:
      Division de la Clase , 5: Potencia ]", 0xa
216 len equ $ - msg
217
218
219 msg_3: db 10,13, "Ha Salido del Menu :D", 0xa
220 len_3 equ $ - msg_3
221
222 section .bss
223 buf resb 11
224 var1: resb 4
225
```

Terminal Output:

```
Seleccion opcion [1: Suma, 2: Resta, 3: Multiplicación, 4: Division de la Clase , 5:
Potencia ]
5
16
Seleccion opcion [1: Suma, 2: Resta, 3: Multiplicación, 4: Division de la Clase , 5:
Potencia ]
6
Ha Salido del Menu :D
```

**Nota Importante:** Para poder salir del menú agregue un mensaje que al momento de no encontrar una opcion de las 5 que estan preestablecidas automaticamente saldrá el programa.