

5

| NOMBRE DE LA PRÁCTICA | SUMA DE DOS NÚMEROS CON DOS DÍGITOS | | | No. | UNIDAD 1 |
|-----------------------|-------------------------------------|-----------|------|-------|---------------|
| ASIGNATURA: | LENGUAJE INTERFAZ | CARRER A: | ISIC | PLAN: | ISIC-2010-204 |

Nombre: Vanesa Hernández Martínez

Grupo: 3501

Objetivo: Desplegar la suma de dos números con dos dígitos y su resultado sea también de dos dígitos también.

Elabora un programa en ensamblador que realice la suma de dos números con dos dígitos y el resultado también sea de dos dígitos. Explica lo que realiza cada renglón.

```

hola.asm  Func.asm  eje1.asm  eje2.asm*  eje3.asm  eje4.asm  reto.asm  x  eje5.asm  eje6.asm
; Define el modelo de memoria como "small". lo que permite que el código y los datos se almacenen en un único segmento.
; Define el segmento de pila.
; Inicia la sección de datos donde se almacenan variables y mensajes.
num1 db 0
num2 db 0
resultado db 0
mensaje db 10, 13, "Primer numero (dos digitos): $" ; Mensaje que se muestra para solicitar el primer número.
mensaje2 db 10, 13, "Segundo numero (dos digitos): $" ; Mensaje que se muestra para solicitar el segundo número.
mensaje3 db 10, 13, "La suma es: $" ; Mensaje que se muestra antes del resultado de la suma.
nueva_linea db 10, 13, "$" ; Carácter de nueva línea (10 = LF, 13 = CR).

; Inicia la sección de código.
main proc
; Inicio del procedimiento principal.
mov ax, @data ; Cargar la dirección del segmento de datos en AX.
mov ds, ax ; Mover AX al registro DS para establecer el segmento de datos.

; Solicitar primer número de dos dígitos
mov ah, 09h ; Función DOS para mostrar una cadena.
lea dx, mensaje ; Cargar la dirección del mensaje en DX.
int 21h ; Interrupción DOS para mostrar el mensaje.

; Leer el primer dígito del primer número
mov ah, 01h ; Función DOS para leer un carácter de entrada.
int 21h ; Interrupción DOS para leer el primer carácter.
sub al, 30h ; Convertir el carácter ASCII a su valor numérico restando 30h (ASCII de '0').
mov bl, al ; Guardar el primer dígito en BL.

; Leer el segundo dígito del primer número
mov ah, 01h ; Función DOS para leer un carácter.
int 21h ; Leer el segundo carácter.
sub al, 30h ; Convertir el carácter ASCII a valor numérico.
mov bh, al ; Guardar el segundo dígito en BH.

; Calcular num1 = (primer dígito * 10) + segundo dígito
mov al, bl ; Mover el primer dígito (BL) a AL.
mov ah, 0 ; Limpiar AH.
mov cl, 10 ; Cargar 10 en CL.
mul cl ; Multiplicar AL por 10.
add al, bh ; Sumar el segundo dígito al resultado.
mov num1, al ; Guardar el número completo en num1.

; Solicitar segundo número de dos dígitos
mov ah, 09h ; Función DOS para mostrar un mensaje.
lea dx, mensaje2 ; Cargar la dirección del mensaje2 en DX.
int 21h ; Interrupción DOS para mostrar el mensaje.

; Leer el primer dígito del segundo número
mov ah, 01h ; Función DOS para leer un carácter.
int 21h ; Leer el primer carácter.
sub al, 30h ; Convertir el carácter ASCII a valor numérico.
mov bl, al ; Guardar el primer dígito en BL.

; Leer el segundo dígito del segundo número
mov ah, 01h ; Función DOS para leer un carácter.
int 21h ; Leer el segundo carácter.
sub al, 30h ; Convertir el carácter ASCII a valor numérico.
mov bh, al ; Guardar el segundo dígito en BH.

; Calcular num2 = (primer dígito * 10) + segundo dígito
mov al, bl ; Mover el primer dígito (BL) a AL.
mov ah, 0 ; Limpiar AH.
mov cl, 10 ; Multiplicar AL por 10.
mul cl ; Multiplicar AL por 10.
add al, bh ; Sumar el segundo dígito.
mov num2, al ; Guardar el número completo en num2.

; Sumar num1 y num2
mov al, num1 ; Cargar num1 en AL.
add al, num2 ; Sumar num2.
mov resultado, al ; Guardar la suma en resultado.

; Mostrar el mensaje de la suma
mov ah, 09h ; Función DOS para mostrar un mensaje.
lea dx, mensaje3 ; Cargar la dirección del mensaje3 en DX.
int 21h ; Interrupción DOS para mostrar el mensaje.

; Preparar para mostrar el resultado
mov al, resultado ; Cargar el resultado en AL.
xor ah, ah ; Limpiar AH (poner a cero).
cmp al, 0 ; Comparar el resultado con 0.
jz mostrar_cero ; Si es cero, saltar a la rutina para mostrar "0".

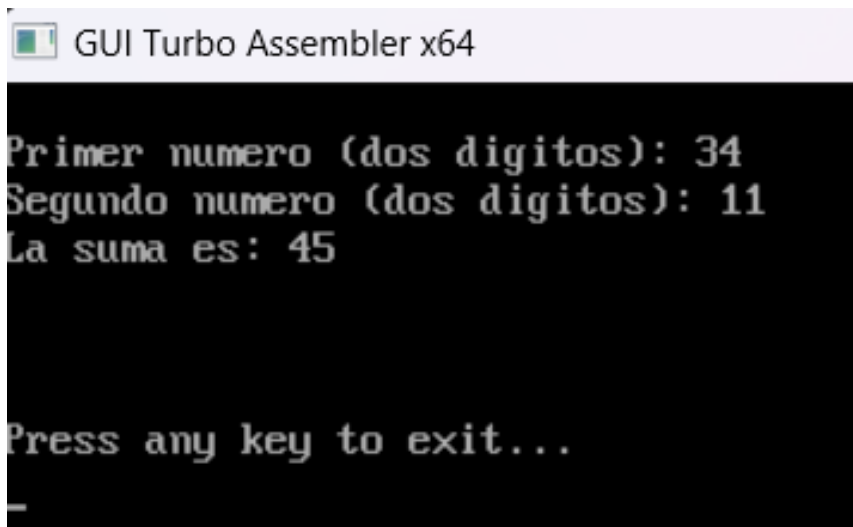
```

```

81 ; Convertir a caracteres ASCII
82 mov bx, 10 ; Preparar el divisor (10 para convertir a base decimal).
83 xor cx, cx ; Limpiar CX, que se usará como contador de dígitos.
84
85
86
87 convertir:
88 xor dx, dx ; Limpiar DX antes de dividir.
89 div bx ; Dividir AX entre 10, el residuo quedará en DX.
90 push dx ; Guardar el residuo (dígito) en la pila.
91 inc cx ; Incrementar el contador de dígitos.
92 test ax, ax ; Comprobar si el cociente es 0.
93 jnz convertir ; Si no es 0, repetir el proceso.
94
95 ; Mostrar los dígitos en orden inverso
96 mostrar_digito:
97 pop dx ; Obtener el dígito de la pila.
98 add dl, 30h ; Convertir a ASCII.
99 mov ah, 02h ; Función DOS para mostrar un carácter.
100 int 21h ; Interrupción DOS para mostrar el carácter.
101 loop mostrar_digito ; Repetir hasta mostrar todos los dígitos.
102
103 jmp fin ; Saltar al final.
104
105
106 mostrar_cero:
107 ; Mostrar el número cero
108 mov dl, '0' ; Cargar el carácter '0' en DL.
109 mov ah, 02h ; Función DOS para mostrar un carácter.
110 int 21h ; Interrupción DOS para mostrar "0".
111
112 fin:
113 ; Mostrar una nueva línea
114 mov ah, 09h ; Función DOS para mostrar una cadena.
115 lea dx, nueva_linea ; Cargar la dirección de nueva_linea en DX.
116 int 21h ; Interrupción DOS para mostrar la nueva línea.
117
118 ; Terminar el programa
119 mov ax, 4C00h ; Función DOS para terminar el programa.
120 int 21h ; Interrupción DOS para salir del programa.
121
122 main endp ; Fin del procedimiento principal.
123 end main ; Fin del programa.

```

Escribe las instrucciones y captura de pantalla que demuestre que el programa si corrió:



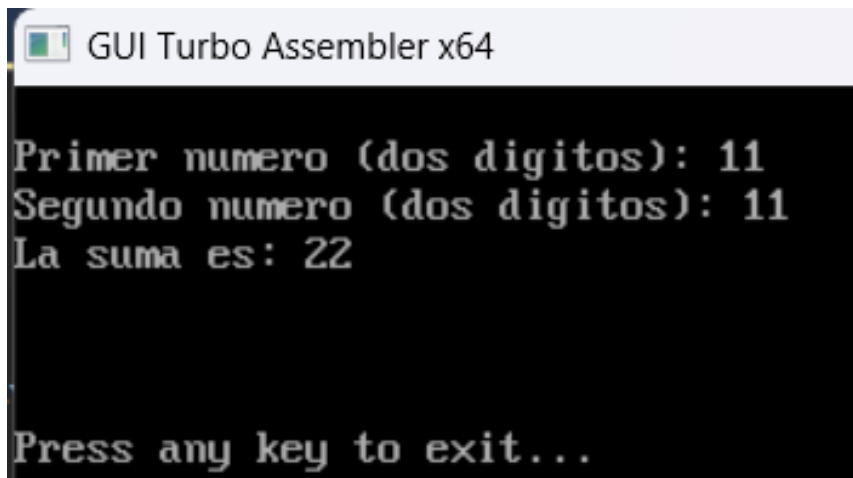
```

GUI Turbo Assembler x64

Primer numero (dos digitos): 34
Segundo numero (dos digitos): 11
La suma es: 45

Press any key to exit...

```





```

GUI Turbo Assembler x64

Primer numero (dos digitos): 11
Segundo numero (dos digitos): 11
La suma es: 22

Press any key to exit...

```

| | | |
|---|--|--|
|  GOBIERNO DEL ESTADO DE MÉXICO | <h1>RETO</h1> <p>Ing. y Esp. Rodolfo Guadalupe Alcántara Rosales</p> |  TESI TECNOLÓGICO DE ESTUDIOS SUPERIORES JILOTEPEC |
|---|--|--|

CONCLUSIONES

El código en ensamblador presentado es un ejemplo claro de cómo trabajar directamente con la memoria y los registros de un procesador. Utiliza interrupciones del sistema DOS para interactuar con el usuario, leer entradas y mostrar salidas. La estructura del programa sigue un flujo básico para realizar una operación aritmética simple, la suma de dos números de dos dígitos, lo que demuestra cómo se puede manipular directamente los valores en registros y convertir datos desde formato ASCII a su representación numérica.