Universidad Autónoma de Baja California
Facultad de Ingeniería Arquitectura y Diseño

Ingeniería en Software y Tecnologías Emergentes

Materia: Organización de Computadoras

Alumno: Jesus Eduardo Rodríguez Ramírez

Profesor: Jonatan Crespo Ragland

Grupo 932

Trabajo: CÓDIGOS Taller 11

Ensenada, B.C; a 22 de noviembre del 2024

Fecha (dd/mm/yyyy):

```
fecha:
db 0; Día
db 0; Mes
dw0; Año
```

Correo electrónico:

```
section .data
correoUsuario db "usuario", 0
correoDominio db "dominio.com", 0
correoCompleto db "usuario@dominio.com", 0

section .text
    global _start

_start:
    mov eax, 4
    mov ebx, 1
    mov ecx, correoCompleto
    mov edx, 20         ; Tamaño aproximado del correo completo
    int 0x80            ; Imprimir

    mov eax, 1
    xor ebx, ebx
    int 0x80
```

Dirección completa:

```
section .data
formato db "Calle: %s, Numero: %d, Colonia: %s", 0

section .text
    ; Simular concatenación y formato similar al printf
```

Curp:

```
_start:
    ; Verificar longitud de la CURP
    mov esi, curp
    xor ecx, ecx
contarCaracteres:
    lodsb
    cmp al, 0
    je mostrarCURP
    inc ecx
    jmp contarCaracteres

mostrarCURP:
    cmp ecx, 18
    jne error

    ; Imprimir CURP
    mov eax, 4
    mov ebx, 1
    mov ecx, curp
    mov edx, 18
    int 0x80

    mov eax, 1
    xor ebx, ebx
    int 0x80

error:
    ; Mostrar mensaje de error
```

Código 1

```
section .data
    num1 db 5              ; Primer número
    num2 db 11             ; Segundo número
    result db 0            ; Resultado de la suma
    message db "Resultado: ", 0 ; Mensaje antes del resultado, terminado en null

section .bss
    buffer resb 4          ; Buffer de 4 bytes para almacenamiento temporal

section .text
    global _start          ; Punto de inicio del programa

    ; Macro para imprimir una cadena
    %macro PRINT_STRING 1
        mov eax, 4         ; Llamada al sistema: write
        mov ebx, 1         ; Salida estándar (stdout)
        mov ecx, %1        ; Dirección de la cadena
        mov edx, 13        ; Longitud de la cadena a imprimir
        int 0x80           ; Ejecutar la llamada al sistema
    %endmacro

    ; Macro para imprimir un número
    %macro PRINT_NUMBER 1
        mov eax, %1        ; Carga el número a imprimir en EAX
        add eax, '0'       ; Convierte el valor numérico a ASCII
        mov [buffer], al   ; Almacena el carácter ASCII en el buffer
        mov eax, 4         ; Llamada al sistema: write
        mov ebx, 1         ; Salida estándar (stdout)
        mov ecx, buffer    ; Dirección del buffer
        mov edx, 1         ; Longitud de la cadena (1 byte)
        int 0x80           ; Ejecutar la llamada al sistema
    %endmacro

_start:
    ; Realiza la suma de num1 y num2
    mov al, [num1]         ; Carga el valor de num1 en AL
    add al, [num2]         ; Suma el valor de num2 al contenido de AL
    mov [result], al       ; Almacena el resultado en la variable result

    ; Imprime el mensaje "Resultado: "
    PRINT_STRING message

    ; Imprime el resultado de la suma
    PRINT_NUMBER [result]

    ; Salida del programa
    mov eax, 1             ; Llamada al sistema: exit
    xor ebx, ebx           ; Código de salida 0
    int 0x80               ; Ejecutar la llamada al sistema
```

Código 2

```
section .data
    message db "La suma de los valores es: ", 0  ; Mensaje inicial
    newline db 10, 0                             ; Nueva línea
    val1 db 3                                    ; Primer valor
    val2 db 5                                    ; Segundo valor
    val3 db 7                                    ; Tercer valor

section .bss
    buffer resb 4                                ; Buffer para la conversión de núme

section .text
    global _start

    ; Macro para imprimir una cadena
    %macro PRINT_STRING 1
        mov eax, 4            ; Syscall para write
        mov ebx, 1            ; Salida estándar
        mov ecx, %1           ; Dirección de la cadena
        mov edx, 25           ; Longitud fija para mensaje
        int 0x80              ; Ejecuta la syscall
    %endmacro

    ; Macro para imprimir un número
    %macro PRINT_NUMBER 1
        mov eax, %1           ; Número a imprimir
        mov ecx, buffer + 3   ; Apunta al final del buffer
        mov ebx, 10           ; Divisor decimal

    .next_digit:
        xor edx, edx          ; Limpia edx
        div ebx               ; Divide eax entre 10
        add dl, '0'           ; Convierte el residuo a ASCII
        dec ecx               ; Retrocede en el buffer
        mov [ecx], dl         ; Almacena el dígito en el buffer
        test eax, eax         ; Verifica si quedan dígitos
        jnz .next_digit       ; Repite si hay más dígitos

        ; Imprime el número
        mov eax, 4            ; Syscall para write
        mov ebx, 1            ; Salida estándar
        mov edx, buffer + 4
        sub edx, ecx          ; Longitud del número
        mov ecx, ecx          ; Dirección del número
        int 0x80              ; Ejecuta la syscall
    %endmacro

_start:
    ; Imprime el mensaje inicial
    PRINT_STRING message

    ; Suma los tres valores
    mov al, [val1]            ; Primer valor
    add al, [val2]            ; Suma el segundo valor
    add al, [val3]            ; Suma el tercer valor
    movzx eax, al             ; Expande a 32 bits

    ; Imprime el resultado
    PRINT_NUMBER eax
    PRINT_STRING newline

    ; Salida del programa
    mov eax, 1                ; Syscall para exit
    xor ebx, ebx              ; Código de salida 0
    int 0x80                  ; Ejecuta la syscall
```