Actividad 7

Trabajo en clase. Explicar el funcionamiento del siguiente código. Identificando tópicos vistos en la clase de hoy. Como arreglos, macros, referencias, etc.

```
%macro print_int 1
    mov eax, 4
    mov ebx, 1
    mov ecx, %1
    mov edx, 4
    int 0x80
%endmacro

section .data
    array dd 1, 2, 3, 4, 5

section .text
    global _start

_start:
    mov ecx, 0
    mov eax, 0

bucle:
    mov ebx, [array + ecx*4]
    add eax, ebx

    inc ecx
    cmp ecx, 5
    jl bucle

    print_int eax

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

Este código es un ejemplo en lenguaje ensamblador x86 que suma los elementos de un arreglo y luego imprime el resultado utilizando la llamada al sistema write (con la interrupción int 0x80). Aquí está el funcionamiento del código:

Macro:
Se define una macro llamada print_int que simplifica la impresión de enteros utilizando la llamada al sistema write. Toma un argumento y lo imprime utilizando la interrupción int 0x80. Esta macro se utiliza más adelante para imprimir el resultado de la suma.
Declaración de secciones:
Se definen dos secciones: .data y .text. La sección .data es para datos estáticos, mientras que la sección .text es para el código.
Declaración de datos:
En la sección .data, se define un arreglo llamado array con los elementos 1, 2, 3, 4 y 5. Cada elemento es de 4 bytes (dd).
Sección de código:
En la sección .text, se define el punto de entrada _start, que es donde comienza la ejecución del programa.
Inicialización de registros:
mov ecx, 0: Se inicializa el contador del bucle ecx en 0.
mov eax, 0: Se inicializa el acumulador de la suma eax en 0.
Bucle de suma:
Etiqueta bucle: Se inicia un bucle que recorre el arreglo.
mov ebx, [array + ecx*4]: Se carga el elemento actual del arreglo en ebx. array + ecx*4 es la dirección de memoria del elemento actual, donde ecx*4 se utiliza para acceder a cada elemento del arreglo que es de 4 bytes de tamaño.
add eax, ebx: Se suma el elemento actual al acumulador en eax.
Actualización del contador y comprobación de fin de bucle:
inc ecx: Se incrementa el contador del bucle para pasar al siguiente elemento del arreglo.
cmp ecx, 5: Se compara el contador con 5, la longitud del arreglo.
jl bucle: Si el contador es menor que 5, se salta de vuelta al inicio del bucle.
Impresión del resultado:
Una vez que el bucle ha sumado todos los elementos del arreglo, se llama a la macro print_int para imprimir el resultado contenido en eax.
Terminación del programa:
mov eax, 1: Se carga el número de la llamada al sistema para terminar el programa en eax.
xor ebx, ebx: Se borra ebx, ya que no se utiliza.
int 0x80: Se realiza la llamada al sistema para terminar el programa.