EJEMPLOS EN ENSAMBLADOR

Introducción

El lenguaje ensamblador como cualquier lenguaje de programación es un conjunto de palabras que le indican al ordenador lo que tiene que hacer. Sin embargo la diferencia fundamental es que cada instrucción escrita en lenguaje ensamblador tiene una correspondencia exacta con una operación en el procesador. Por lo que son operaciones muy sencillas tales como: “Cargar 16 en el registro BX” o “Transferir el contenido del registro CL al CH”. Así pues, las palabras del lenguaje ensamblador son nemotécnicos que representan el código máquina, lenguaje que entiende el procesador.

EJEMPLO 1

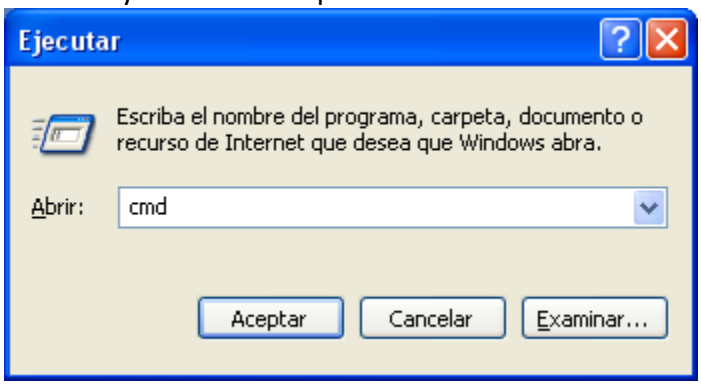
Programa que realiza la suma de 3 números hexadecimales y guardarlo en el registro AX en el depurador de Windows (DEBUG). EJM: AX=2+3+4 RESULTADO: AX=9

PASO A PASO

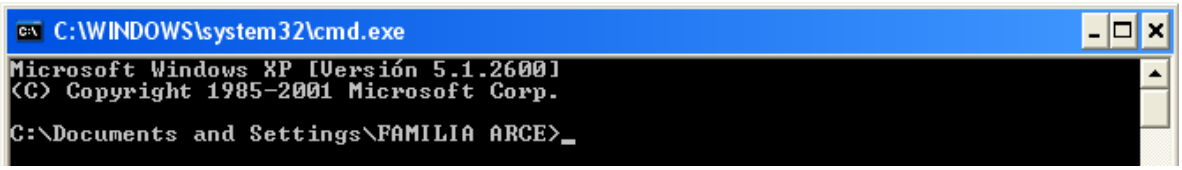
Primero abrir el depurador de windows (debug) de la siguiente manera:

PASO 1

Abrir ejecutar con CTRL + R y escribir CMD para abrir el símbolo del sistema:

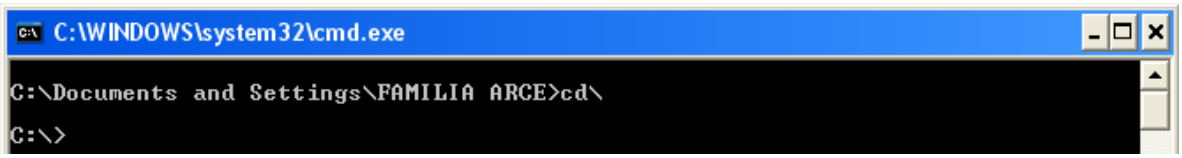


Y nos muestra la siguiente ventana:

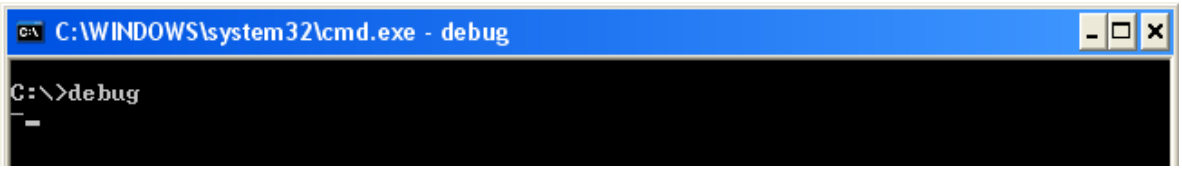


PASO 2

Cambiar la dirección actual y llevarlo al disco C:

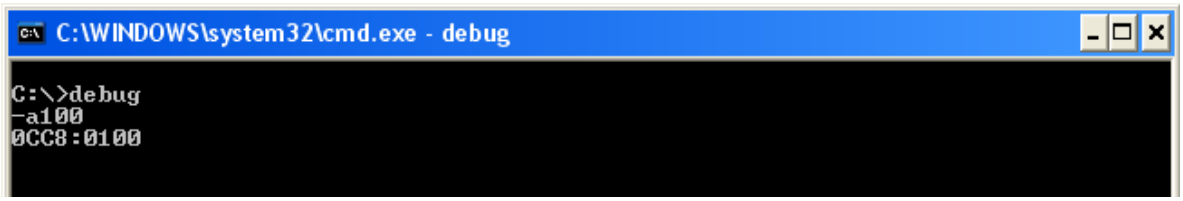


PASO 3 Escribir en la línea de comando la palabra DEBUG y nos muestra lo siguiente:



PASO 4

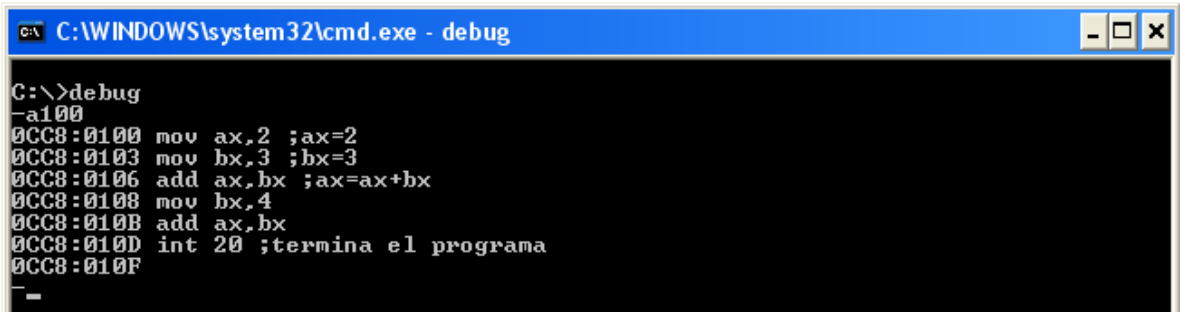
Ahora podemos empezar a ensamblar nuestro programa, asi que escribimos **a100** (a=assembler y el 100 equivale la dirección de inicio del programa) y nos muestra la dirección lógica de la primera instrucción a introducir.
(0CC8 : 0100 = segmento : desplazamiento)



PASO 5

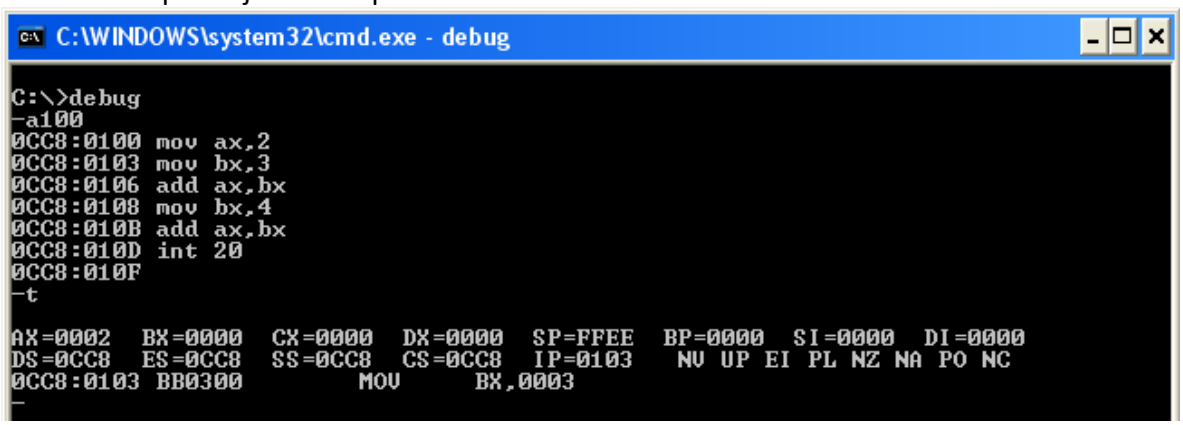
Luego introducimos el siguiente código:

```
mov ax,2        ;asigna al registro ax el valor 2, equivale en C++:    ax=2
mov bx,3        ;asigna al registro bx el valor 3, equivale en C++:    bx=3
add ax,bx       ;suma ambos registros y lo guarda en ax, equivale en C++: ax=ax+bx
mov bx,4        ;asigna al registro bx el valor 4, equivale en C++:    bx=4
add ax,bx       ;suma ambos registros y lo guarda en ax, equivale en C++: ax=ax+bx
int 20          ;Interrupción que termina el programa
```



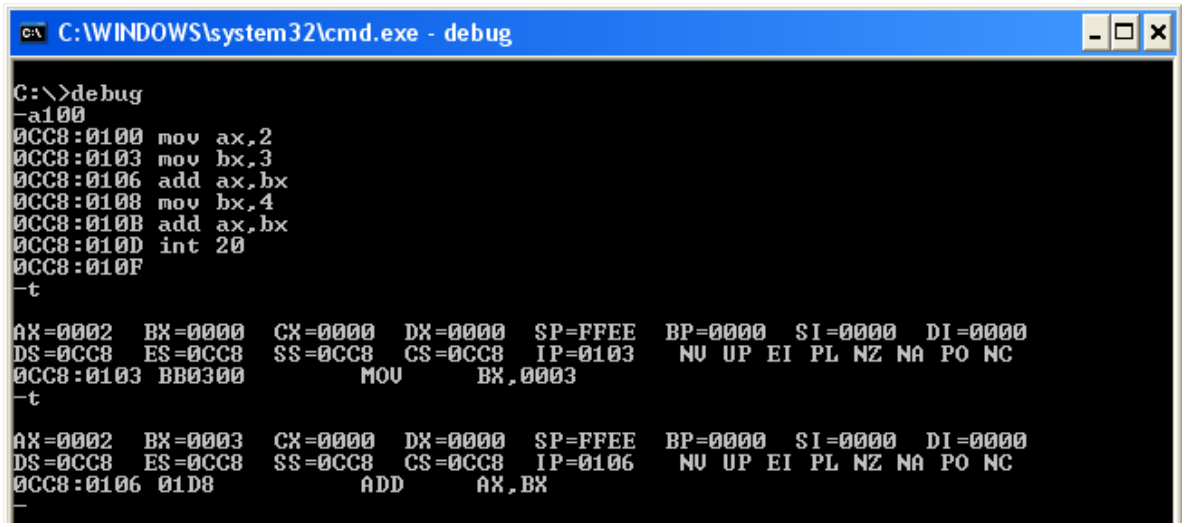
PASO 6

Para hacer correr el programa paso a paso se hace lo siguiente:
Escribimos t para ejecutar la primera instrucción:

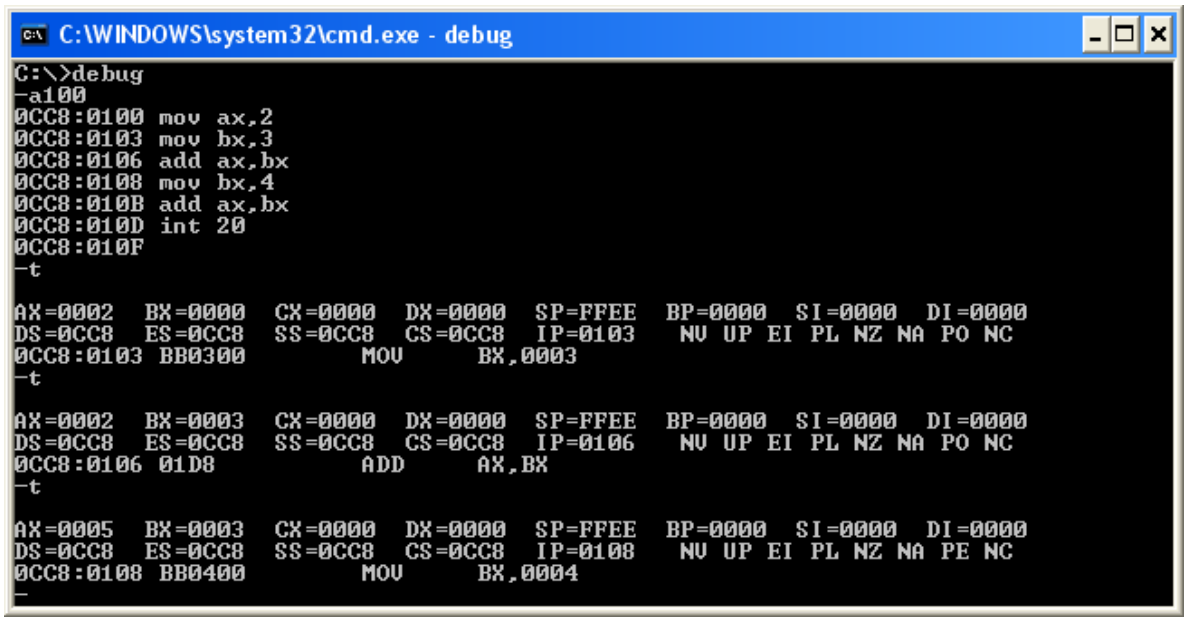


Y nos muestra la primera instrucción que asigna ax=2 y también muestra todos los registros del microprocesador y la ultima línea muestra la siguiente instrucción a ejecutar.

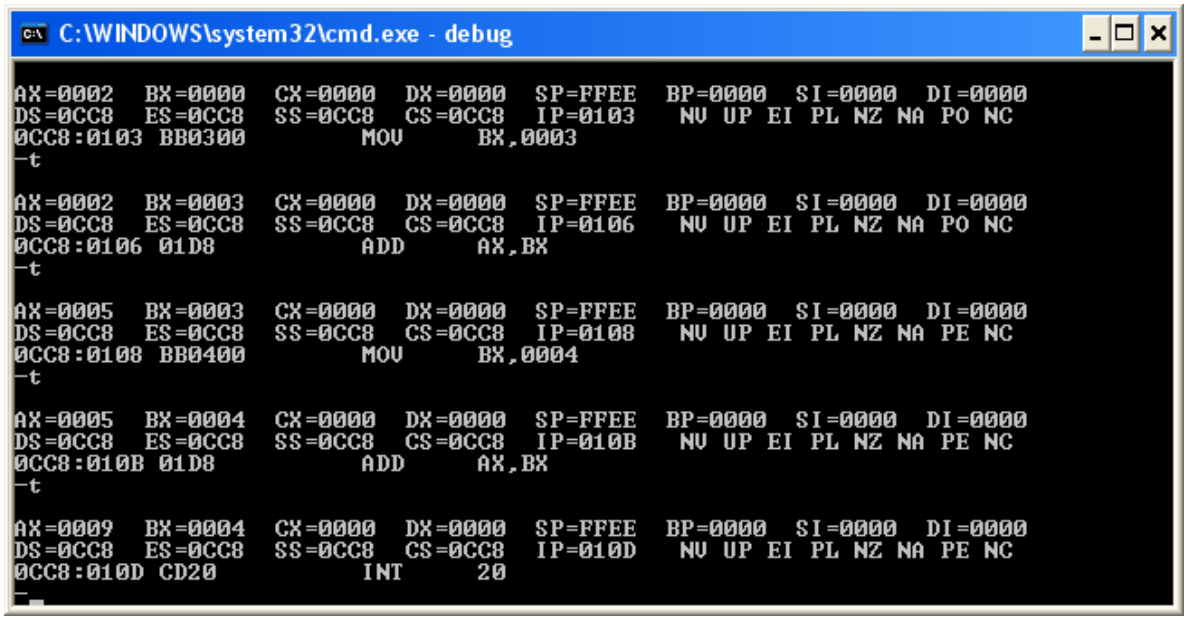
Escribimos otra vez t



Escribimos otra vez t



Escribimos dos veces t hasta llegar a la ultima intruccion y nos muestre el resultado de las suma en el registro AX.

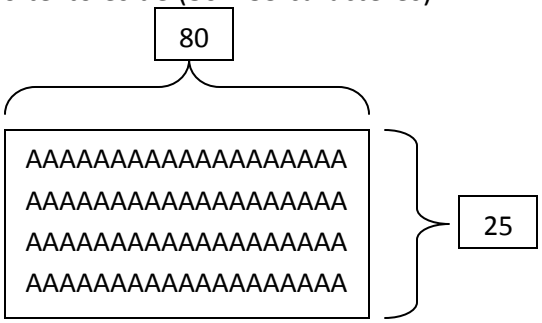


Resultado: AX=9

EJEMPLO 2

HACER UN PROGRAMA QUE MUESTRE EN TODA LA PANTALLA LA LETRA A(MAYUSCULA).

Resolución de pantalla en modo texto es de (80 x 35 caracteres)



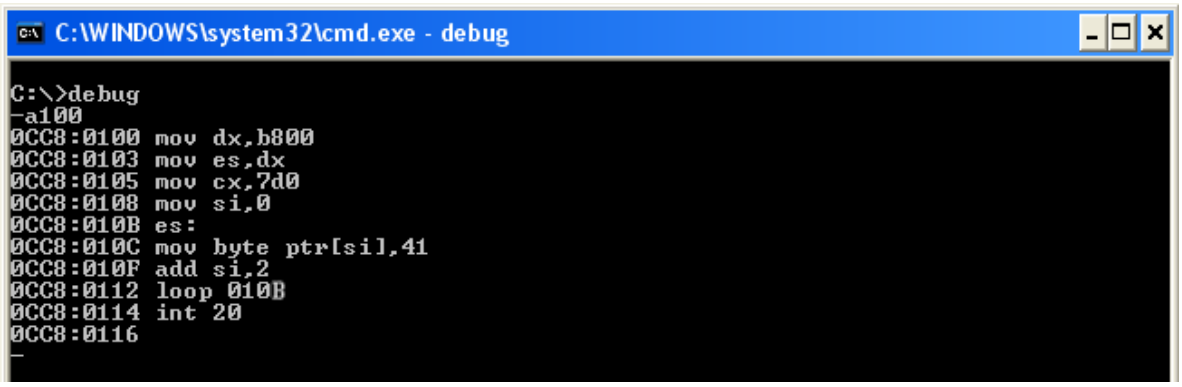
Código ASCII

La letra A mayúscula es igual a (65 decimal) y (41 en Hexadecimal)
Como debug solo usa el sistema hexadecimal usaremos A=41
Segmento de memoria de la pantalla: Esta es igual a B800 donde la letra A se encontraría en la dirección lógica B800:0041

Ya tenemos las pautas necesaria para realizar el programa.

Escribimos el siguiente código en el debug:

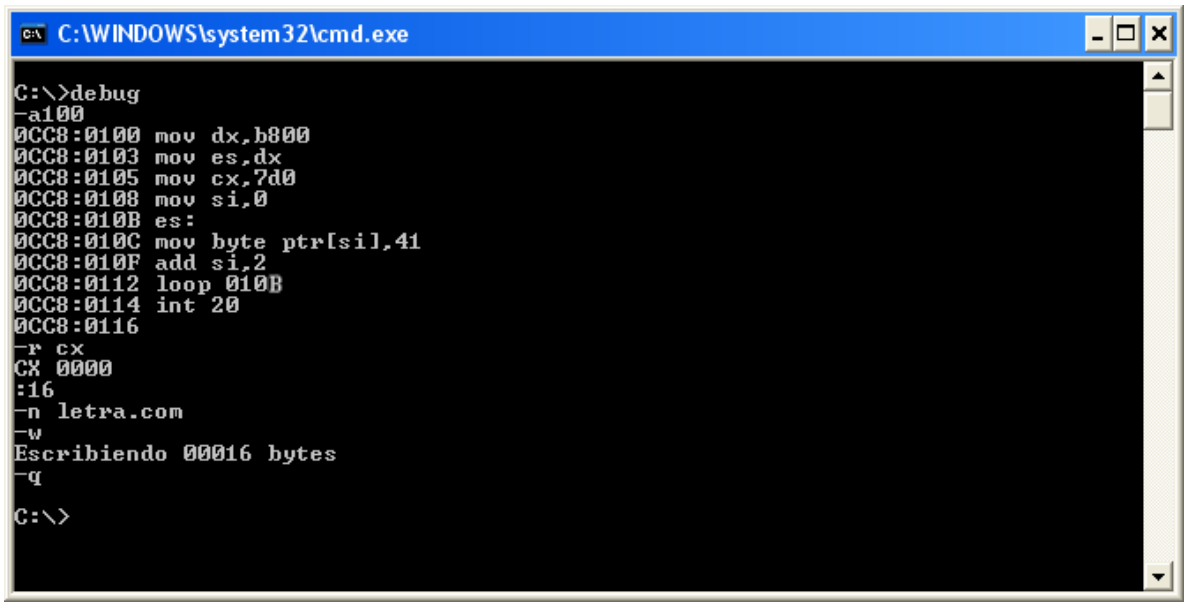
```
XXXX:0100   mov dx,B800 ;asignamos la dirección del segmento de pantalla en dx
XXXX:0103   mov es,dx   ; y asigno al segmento extra el valor de dx
XXXX:0105   mov cx,7D0 ;Asigno el # de ciclos,es decir 80 x 35=2000 en hexa=7D0
XXXX:0108   mov si,0 ;inicializo si en 0 ,que se encargar de recorrer la letra por pantalla
XXXX:010B   es:      ;con esta instrucción me dirijo al segmento de pantalla
XXXX:010C   mov byte ptr[si],41  ;asigno la letra A (41) de tipo byte y apunto a [si]
XXXX:010F   add si,2         ;sumo si=si+2 para desplazar los 2000 caracteres por pantalla
XXXX:0112   loop 10B ;repite cx veces desde la dirección 10B y disminuye cx hasta ser=0
XXXX:0114   int 20 ; terminar el programa
```



Guardar el programa

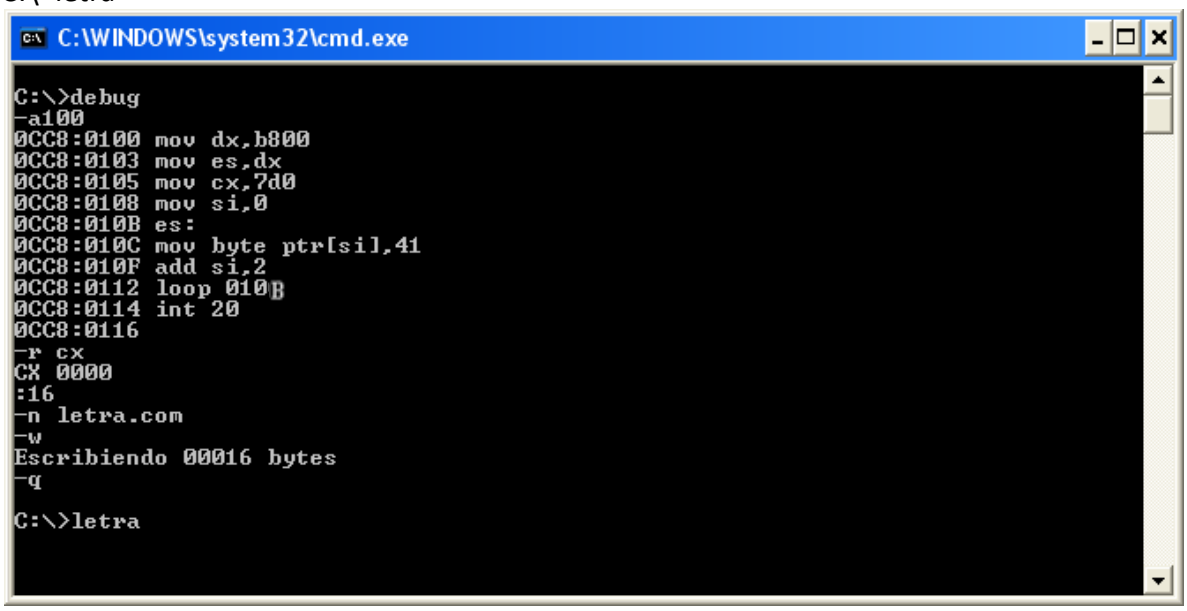
- Calcular cuanto pesa el programa restando la dirección final con la dirección de inicio por ejm: 0116 – 0100=16
- Reservar memoria para el programa con el comando r cx donde será igual a 16
- Asignar un nombre al programa con extensión .com con el comando n por ejm: n letra.com
- Escribir el programa en disco con el comando w

```
-r cx
cx 0000
: 16
-n letra.com
-w
-q   (Comando para salir del debug)
```



EJECUTAR EL PROGRAMA

Simplemente para ejecutar el programa llamarlo por su nombre en este caso escribir C:\>letra



Y nos muestra lo siguiente: