TECNOLÓGICO NACIONAL DE MÉXICO

INSTITUTO TECNOLÓGICO SUPERIOR PROGRESO

PROGRAMA ACADÉMICO DE INGENIERÍA EN SISTEMAS COMPUTACIONALES

ASIGNATURA

Lenguaje de interfaz.

DOCENTE

Edgar Alejandro Sagundo Duarte.

TRABAJO

Examen U2.

PRESENTA

Manuel Ricardo Uc Nicoli 04190032

Progreso Yucatán, 08 de abril de 2022.

Examen 2

Desarrollar los siguientes 3 ejercicios entregar en un PDF las capturas de su funcionamiento y entregar también los .asm, todo en un comprimido, el archivo se llamará con su nombre o apellido. Solo se aceptará en el formato de IBM. Si algo no se cumple no se calificará.

1. Desarrollar un programa en ensamblador que permita teclear dos letras y mostrarlos en grande (mínimo 6 líneas) en dos colores diferentes. Las letras serán de sus iniciales de su nombre y apellido y letra no encontrada que diga que no existe. (5pts).

2. Desarrollar un programa en ensamblador que valide del 1 al 3, para que se repita el programa. Se pedirá una letra y una palabra, se reemplazará el carácter dentro de la palabra por @, mostrar el resultado en otro color y las veces que se reemplazó. (Ejemplo : a, Alejandro, se mostrará @lej@ndro, 2 veces. (15 pts.).

3. Desarrollar un programa en ensamblador que permita escribir tu nombre y luego limpiar la pantalla, colocar un cuadro de un color en el centro de la pantalla, conforme vas presionando los números de1 1 al 9 se forme una figura en el centro del cuadro, validar que sea consecutivo los números. y al finalizar despiiegue una frase de despedida con el nombre que se dio al principio. Para las figuras se pueden basar en el siguiente link https://sites.google.com/site/colgadosenlanet/dibujos-ascii ; (30 pts.)

Los alumnos que entregaron el 1 primer ejercicio completo en la sesión fueron Manuel, Henry, Jonathan, Javier, Angel U., Kenn, Mariana

Resumen de sus intentos previos

Estado	Calificación / 50.00	Revisión
Terminados Enviado Friday, 8 de April de 2022, 21:57	Sin calificar aún	Revisión

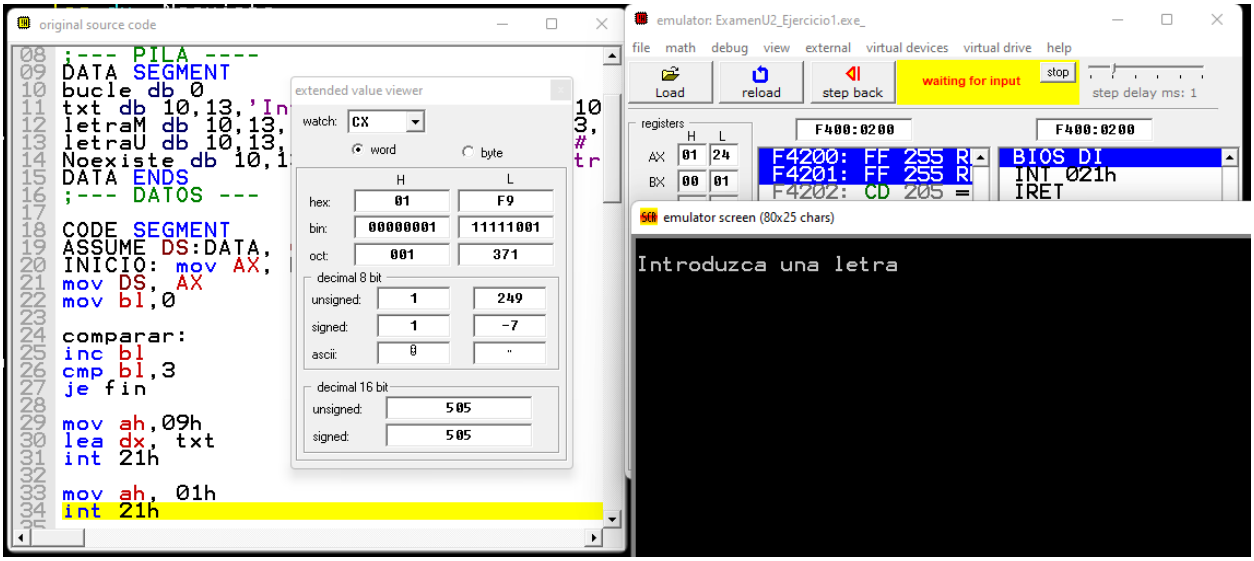
EJERCICIO 1 – CODIGO FUENTE.

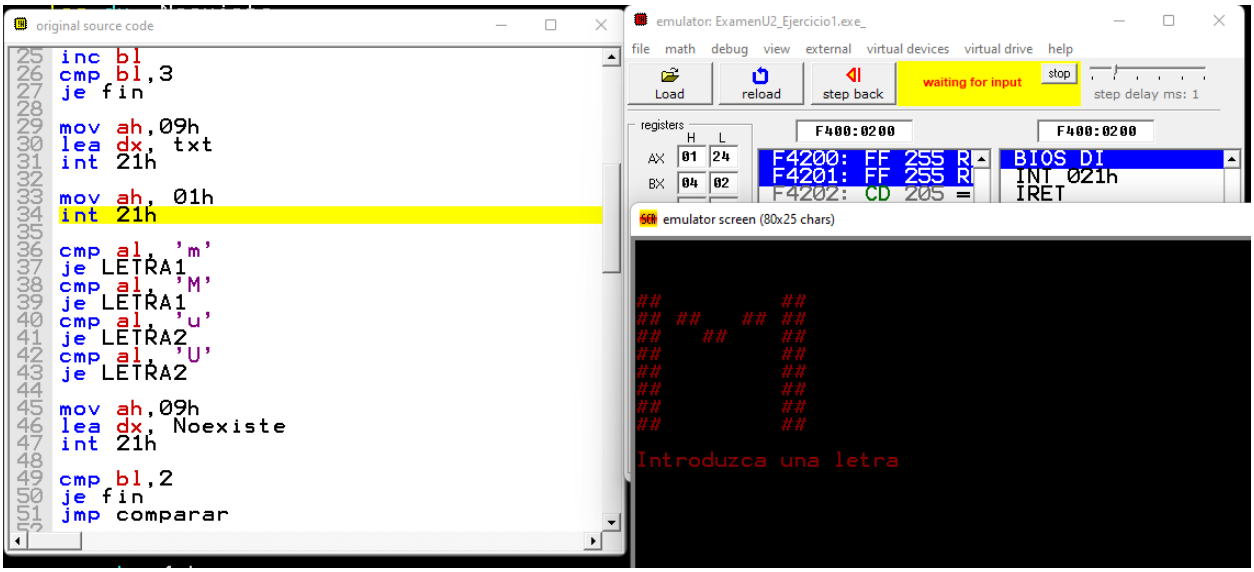
```
;@Autor Manuel Uc Nicoli

;1. Desarrollar un programa en ensamblador que permita teclear dos letras y mostrarlos en grande (mÃ-nimo 6 lÃ-neas)

;en dos colores diferentes. Las letras serÃ¡n de sus iniciales de su nombre y apellido y letra no encontrada que diga que no existe. (5pts).


STACK SEGMENT STACK

DW 64 DUP(?)

STACK ENDS

;--- PILA ----

DATA SEGMENT

bucle db 0

txt db 10,13,'Introduzca una letra',10,13,'$'

letraM db 10,13,'##        ##',10,13,'## ##   ## ##',10,13,'##   ##    ##',10,13,'##        ##',10,13,'##        ##',10,13,'##        ##',10,13,'##        ##',10,13,'$'

letraU db 10,13,'##    ##',10,13,'##    ##',10,13,'##    ##',10,13,'##    ##',10,13,'##    ##',10,13,'##    ##',10,13,'#########',10,13,'$'

Noexiste db 10,13,'No existe esta letra',10,13,'$'

DATA ENDS

;--- DATOS ---


CODE SEGMENT

      ASSUME DS:DATA, CS:CODE, SS:STACK

INICIO: mov AX, DATA

                  mov DS, AX

                  mov bl,0


comparar:

                  inc bl
```

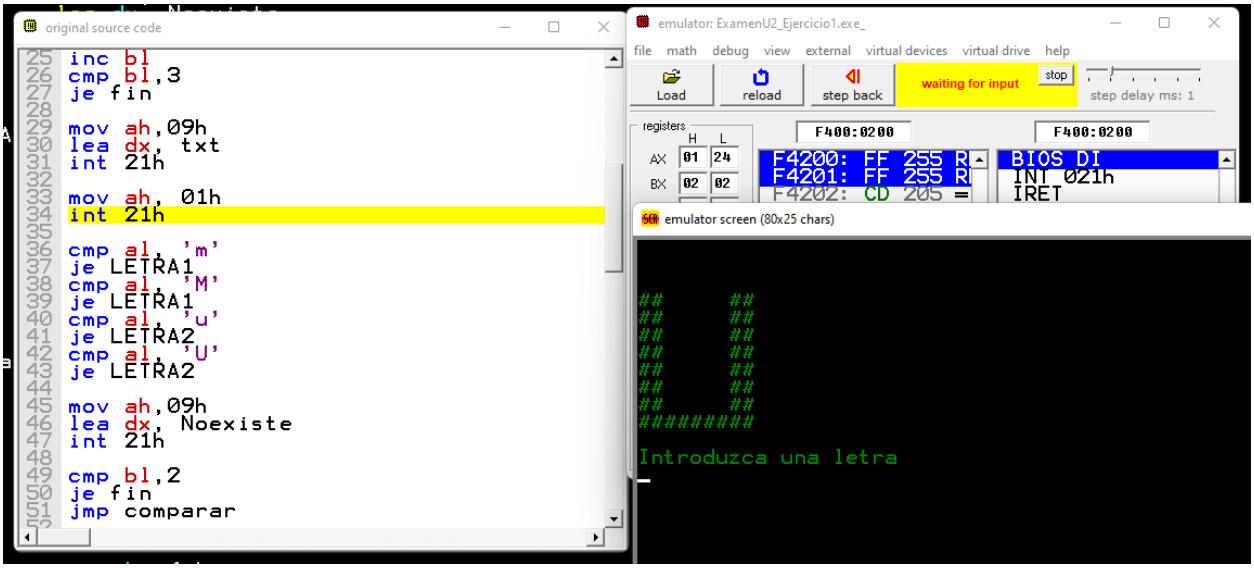
```
                cmp bl,3
                je fin

                mov ah,09h
                lea dx, txt
                int 21h

                mov ah, 01h
                int 21h

                cmp al, 'm'
                je LETRA1
                cmp al, 'M'
                je LETRA1
                cmp al, 'u'
                je LETRA2
                cmp al, 'U'
                je LETRA2

                mov ah,09h
                lea dx, Noexiste
                int 21h

                cmp bl,2
                je fin
                jmp comparar

LETRA1:
                mov ah,06h
```

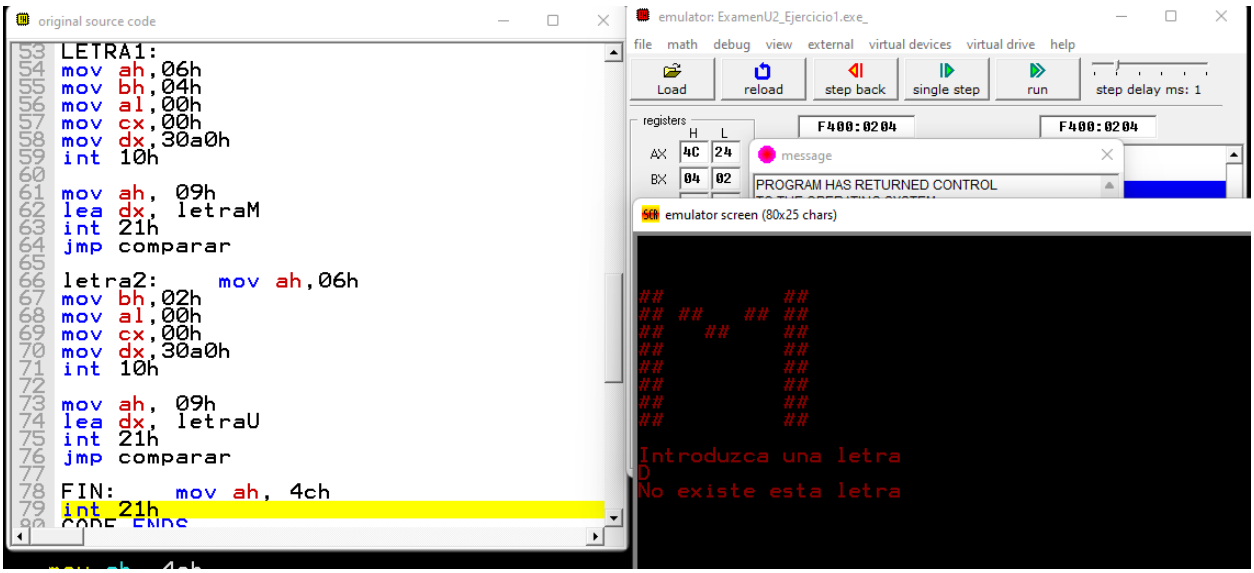
```
                mov bh,04h
        mov al,00h
        mov cx,00h
        mov dx,30a0h
        int 10h

                mov ah, 09h
                lea dx, letraM
                int 21h
                jmp comparar

letra2:  mov ah,06h
        mov bh,02h
        mov al,00h
        mov cx,00h
        mov dx,30a0h
        int 10h

                mov ah, 09h
                lea dx, letraU
                int 21h
                jmp comparar

FIN:    mov ah, 4ch
        int 21h
CODE ENDS
END INICIO
```

CAPTURAS DE FUNCIONAMIENTO.






original source code
inc bl
cmp bl,3
je fin
mov ah,09h
lea dx, txt
int 21h
mov ah, 01h
int 21h
cmp al, 'm'
je LETRA1
cmp al, 'M'
je LETRA1
cmp al, 'u'
je LETRA2
cmp al, 'U'
je LETRA2
mov ah,09h
lea dx, Noexiste
int 21h
cmp bl,2
je fin
jmp comparar
emulator: ExamenU2_Ejercicio1.exe_
file math debug view external virtual devices virtual drive help
Load
reload
step back
waiting for input
stop
step delay ms: 1
registers
AX 01 24
BX 02 02
F400:0200
F4200: FF 255 R
F4201: FF 255 R
F4202: CD 205 =
BIOS DI
INT 021h
IRET
emulator screen (80x25 chars)
Introduzca una letra


original source code
LETRA1:
mov ah,06h
mov bh,04h
mov al,00h
mov cx,00h
mov dx,30a0h
int 10h
mov ah, 09h
lea dx, letraM
int 21h
jmp comparar
letra2: mov ah,06h
mov bh,02h
mov al,00h
mov cx,00h
mov dx,30a0h
int 10h
mov ah, 09h
lea dx, letraU
int 21h
jmp comparar
FIN: mov ah, 4ch
int 21h
emulator: ExamenU2_Ejercicio1.exe_
file math debug view external virtual devices virtual drive help
Load
reload
step back
single step
run
step delay ms: 1
registers
AX 4C 24
BX 04 02
F400:0204
message
PROGRAM HAS RETURNED CONTROL
emulator screen (80x25 chars)
Introduzca una letra
D
No existe esta letra

EJERCICIO 2 – CODIGO FUENTE.

```
;@Autor Manuel Uc Nicoli
;Desarrollar un programa en ensamblador que valide del 1 al 3, para que se repita el programa.
;Se pedirÃ¡ una letra y una palabra, se reemplazarÃ¡ el carÃ¡cter dentro de la palabra por @,
;mostrar el resultado en otro color y las veces que se reemplazÃ³. 15 pts

STACK SEGMENT STACK
DW 64 DUP(?)
STACK ENDS
;--- PILA ----

DATA SEGMENT
repit db 0                              ;variable para determinar cuantas veces se repetira el programa
cambios db 0                    ;almacena el numero de cambios
var db 0            ;varible que almacena cambio de pantalla de colores
caractercambiar db 0    ;variable para almacenar el caracter a cambiar
espacio db 10,13,'$'      ;salto de linea
txt db 10,13,'Introduzca una palabra',10,13,'$'
txt1 db 10,13,'Introduzca un dijito para repetir el programa',10,13,'$'
remplazo db 10,13,'Caracter a remplazar',10,13,'$'
CC db ' cambios realizados a la palabra$'
cadena1 db 10 dup (' '),'$'                     ;cadena original
cadena2 db 10 dup (' '),'$'                     ;cadena copia

DATA ENDS
;--- DATOS ---

CODE SEGMENT
    ASSUME DS:DATA, CS:CODE, SS:STACK
```

```
INICIO: mov AX , DATA
        mov DS, AX

                mov ah,09h
                lea dx, txt1            ;espera el dijito
                int 21h

                mov ah,01h
        int 21h
                mov repit,al            ;muevo el dijito introducido de AL a BL para saber cuantas veces
ejecutar el programa

                cmp al,'1'
                je texto1
                cmp al,'2'
                je texto1
                cmp al,'3'
                je texto1
                jne fin

texto1:
                mov cx,10
                mov si,0

                mov ah,09h
                lea dx, txt     ;pedira el programa que ingrese su palabra
                int 21h

                mov dl,var
```

```
                inc dl
                mov var, dl

leer:   mov ah,07h      ;introduce caracter x caracter
                int 21h
                cmp al,13
                je Integrar
                mov dl,al       ;almacenaremos el dato guardado en AL
                mov ah,02h
                int 21h
                mov cadena1 [si], al    ;almacenamos en la cada lo del puntero
                inc si                  ;incremetamos el puntero para seguir leyendo
caracteres
                loop leer

Integrar: mov si,0
                mov dl,0
                mov cx,10

Copiar: mov al,cadena1[si]
                mov cadena2[si],al
                inc si
                loop Copiar

Programa:
                mov ah, 09h
                lea dx, remplazo        ;caracter a remplezar
                int 21h
```

```
                mov ah, 01h      ;espera el caracter y lo muestra en pantalla. Almacena en al
                int 21h
                mov caractercambiar,al ;guardamos el caracter a remplezar

                mov ah,09h
                lea dx, espacio ;salto de linea
                int 21h

cadenas:
                mov si,0         ;puntero a utilizar
                mov cx,10        ;iniciamos el contador

Comparacion:
                mov al,cadena2[si]                ;comparamos lo ingresado con la cadena copiada
                cmp al,caractercambiar ;Compara lo del puntero con el caracter a cambiar
                je Cambiar       ;si ocurre va al metodo y cambia por el @ de lo contrario solo continua
el programa
                inc si
                loop Comparacion
                jmp Mchanged

Cambiar:mov cadena2[si],'@'    ;se realiza el cambio si coincide con el caracter establecido por el usuario
                mov dl, cambios
                inc dl
                mov cambios,dl
                inc si
                jmp Comparacion

;---------- Impresion de los resultados en pantalla ----------;
```

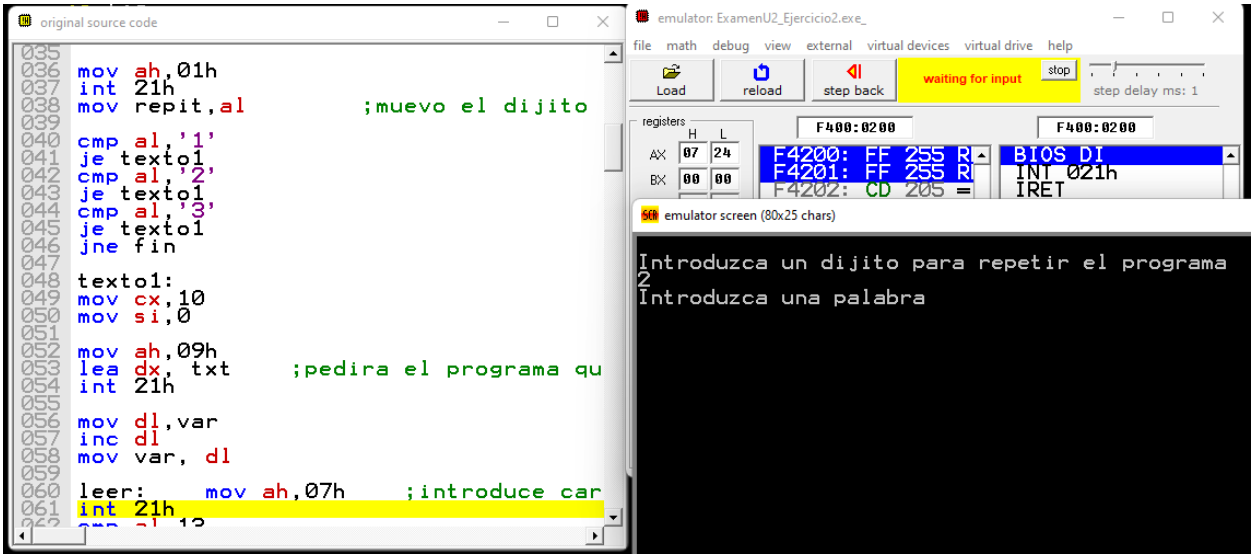
```
Mchanged:
                mov dl,var      ;dato guardado desde el inicio
                cmp dl, 1
                je pantalla1
                cmp dl, 2
                je pantalla2
                cmp dl, 3
                je pantalla3

pantalla1:
    mov ah,00h
    int 10h

                mov ah,06h
                mov bh,02h
    mov al,00h
    mov cx,00h
    mov dx,3030h
    int 10h                     ;opcion de video

                mov ah,09h
                lea dx, cadena2 ;leemos la cadena y hacemos un salto de linea.
                int 21h
                mov ah, 09h
                lea dx, espacio
                int 21h

                mov dl,cambios
                add dl,30h
```

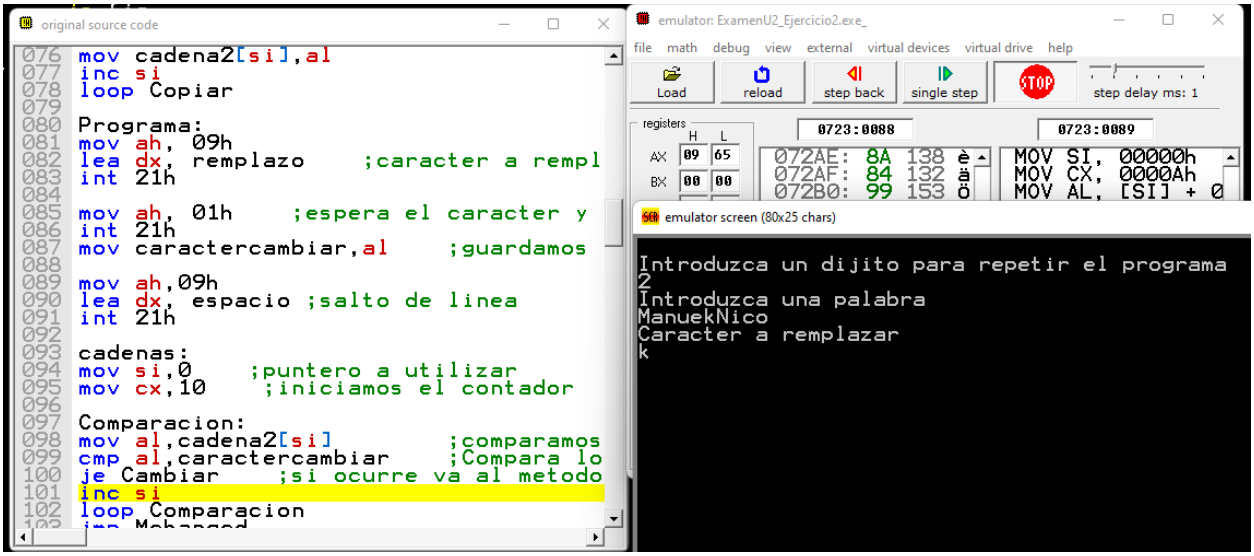
```
                mov ah, 02h ;exibe el caracter que tenga dl, en este caso los cambios.
                int 21h
                mov dl,0
                mov cambios,dl

                mov ah,09h
                lea dx, cc
                int 21h

                cmp repit,'1'
                je fin
                jne texto1

pantalla2:
        mov ah,00h
        int 10h

                mov ah,06h
                mov bh,06h
        mov al,00h
        mov cx,00h
        mov dx,3030h
        int 10h                         ;opcion de video

                mov ah,09h
                lea dx, cadena2 ;leemos la cadena y hacemos un salto de linea.
                int 21h
                mov ah, 09h
                lea dx, espacio
```

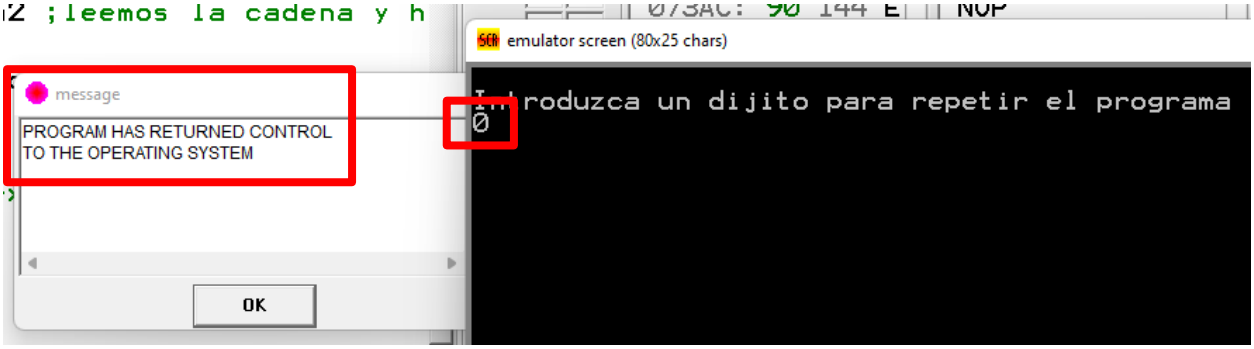
```
                int 21h

                mov dl,cambios
                add dl,30h
                mov ah, 02h ;exibe el caracter que tenga dl, en este caso los cambios.
                int 21h
                mov dl,0
                mov cambios,dl

                mov ah,09h
                lea dx, cc
                int 21h

                cmp repit,'2'
                je fin
                jne texto1

pantalla3:
        mov ah,00h
        int 10h

                mov ah,06h
                mov bh,01h
        mov al,00h
        mov cx,00h
        mov dx,3030h
        int 10h                 ;opcion de video

                mov ah,09h
```

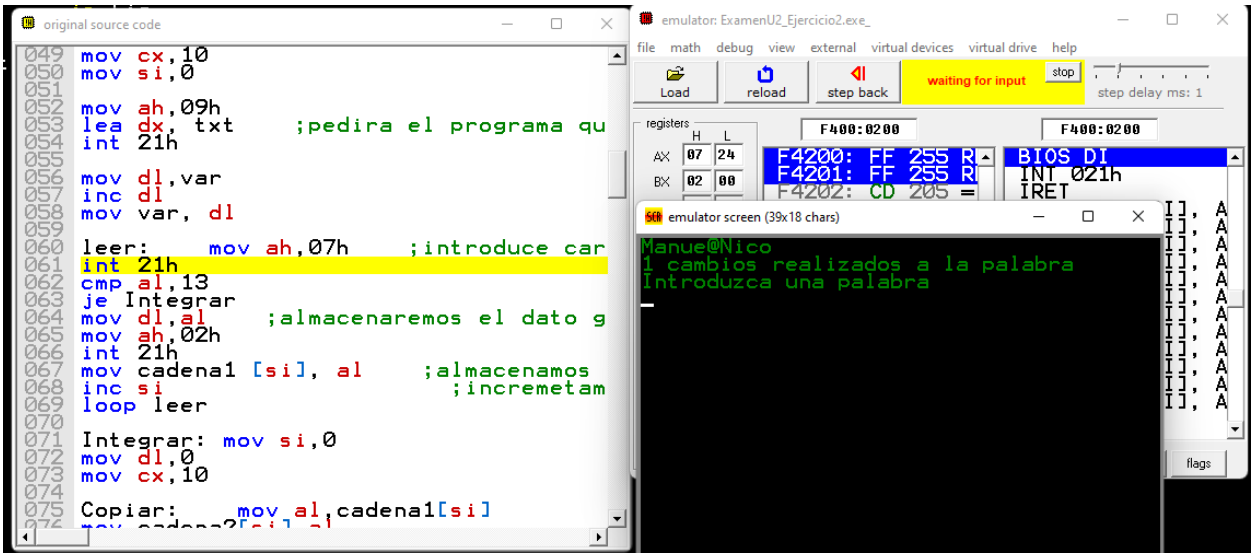
```
        lea dx, cadena2 ;leemos la cadena y hacemos un salto de linea.
        int 21h
        mov ah, 09h
        lea dx, espacio
        int 21h

        mov dl,cambios
        add dl, 30h
        mov ah, 02h ;exibe el caracter que tenga dl, en este caso los cambios.
        int 21h
        mov dl,0
        mov cambios,dl

        mov ah,09h
        lea dx, cc
        int 21h

FIN:    mov ah, 4ch
    int 21h
CODE ENDS
END INICIO
```

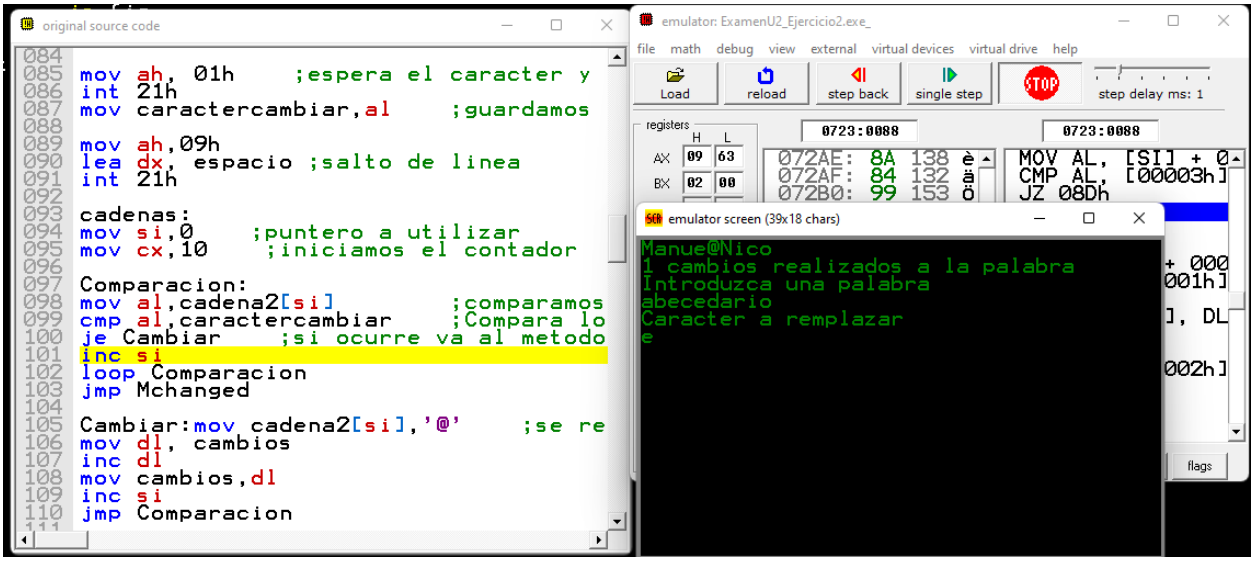
CAPTURAS DE FUNCIONAMIENTO.

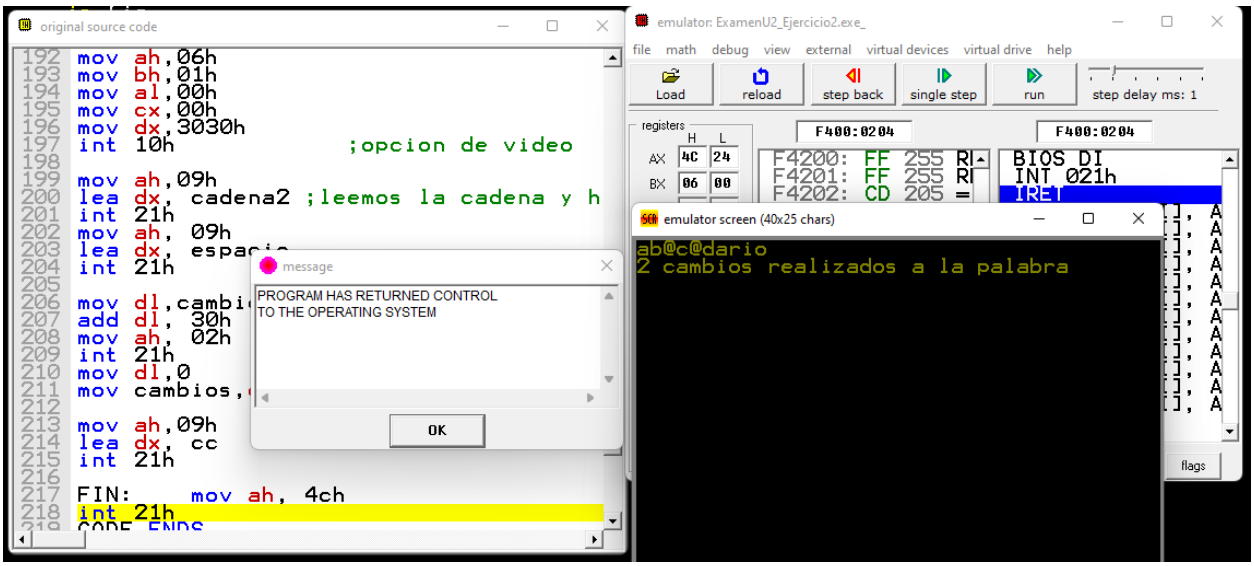







original source code
049 mov cx,10
050 mov si,0
051
052 mov ah,09h
053 lea dx, txt ;pedira el programa qu
054 int 21h
055
056 mov dl,var
057 inc dl
058 mov var, dl
059
060 leer: mov ah,07h ;introduce car
061 int 21h
062 cmp al,13
063 je Integrar
064 mov dl,al ;almacenaremos el dato g
065 mov ah,02h
066 int 21h
067 mov cadena1 [si], al ;almacenamos
068 inc si ;incremetam
069 loop leer
070
071 Integrar: mov si,0
072 mov dl,0
073 mov cx,10
074
075 Copiar: mov al,cadena1[si]
emulator: ExamenU2_Ejercicio2.exe_
file math debug view external virtual devices virtual drive help
Load
reload
step back
waiting for input
stop
step delay ms: 1
registers
AX 07 24
BX 02 00
F400:0200
F4200: FF 255 R
F4201: FF 255 R
F4202: CD 205 =
F400:0200
BIOS DI
INT 021h
IRET
emulator screen (39x18 chars)
Manue@Nico
1 cambios realizados a la palabra
Introduzca una palabra
flags


original source code
084
085 mov ah, 01h ;espera el caracter y
086 int 21h
087 mov caractercambiar,al ;guardamos
088
089 mov ah,09h
090 lea dx, espacio ;salto de linea
091 int 21h
092
093 cadenas:
094 mov si,0 ;puntero a utilizar
095 mov cx,10 ;iniciamos el contador
096
097 Comparacion:
098 mov al,cadena2[si] ;comparamos
099 cmp al,caractercambiar ;Compara lo
100 je Cambiar ;si ocurre va al metodo
101 inc si
102 loop Comparacion
103 jmp Mchanged
104
105 Cambiar:mov cadena2[si],'@' ;se re
106 mov dl, cambios
107 inc dl
108 mov cambios,dl
109 inc si
110 jmp Comparacion
emulator: ExamenU2_Ejercicio2.exe_
file math debug view external virtual devices virtual drive help
Load
reload
step back
single step
STOP
step delay ms: 1
registers
AX 09 63
BX 02 00
0723:0088
072AE: 8A 138 è
072AF: 84 132 ä
072B0: 99 153 ö
0723:0088
MOV AL, [SI] + 0
CMP AL, [00003h]
JZ 08Dh
emulator screen (39x18 chars)
Manue@Nico
1 cambios realizados a la palabra
Introduzca una palabra
abecedario
Caracter a remplazar
e
flags


original source code
192 mov ah,06h
193 mov bh,01h
194 mov al,00h
195 mov cx,00h
196 mov dx,3030h
197 int 10h ;opcion de video
198
199 mov ah,09h
200 lea dx, cadena2 ;leemos la cadena y h
201 int 21h
202 mov ah, 09h
203 lea dx, espacio
204 int 21h
205
206 mov dl,cambi
207 add dl, 30h
208 mov ah, 02h
209 int 21h
210 mov dl,0
211 mov cambios,
212
213 mov ah,09h
214 lea dx, cc
215 int 21h
216
217 FIN: mov ah, 4ch
218 int 21h
message
PROGRAM HAS RETURNED CONTROL
TO THE OPERATING SYSTEM
OK
emulator: ExamenU2_Ejercicio2.exe_
file math debug view external virtual devices virtual drive help
Load
reload
step back
single step
run
step delay ms: 1
registers
AX 4C 24
BX 06 00
F400:0204
F4200: FF 255 RI
F4201: FF 255 RI
F4202: CD 205 =
F400:0204
BIOS DI
INT 021h
IRET
emulator screen (40x25 chars)
ab@c@dario
2 cambios realizados a la palabra
flags

EJERCICIO 3 – CODIGO FUENTE.

```
;@Autor Manuel Uc Nicoli
;3. Desarrollar un programa en ensamblador que permita escribir tu nombre y luego limpiar la pantalla
;colocar un cuadro de un color en el centro de la pantalla, conforme vas presionando los nÃºmeros de1
;1 al 9 se forme una figura en el centro del cuadro, validar que sea consecutivo los nÃºmeros.
;y al finalizar despiiegue una frase de despedida con el nombre que se dio al principio. 30 pts

STACK SEGMENT STACK
DW 64 DUP(?)
STACK ENDS
;--- PILA ----

DATA SEGMENT

var db 49                    ;validar numeros consecutivos
txt db 10,13,'Coloque su nombre',10,13,'$'
nombre db 15 dup (' '),'$'          ;cadena para el nombre
espacio db 10,13,'$'
fig1 db 'Diseno de pistola$'
fig2 db '.....________________________ , ,__$'
fig3 db '....../ `---_____________----_____|] = = = = = D$'
fig4 db '...../_==o;;;;;;;;_______.:/$'
fig5 db '.....), ---.(_(__) /$'
fig6 db '....// (..) ), ----"$'
fig7 db '...//___//$'
fig8 db '..//___//$'
fig9 db './//___//%$'
alternativo db 10,13,'Deben ser numeros consecutivos del 1-9 :(',10,13,'Hasta luego $'
despedida db 10,13,'Hasta luego $'
```

```
DATA ENDS
;--- DATOS ---

CODE SEGMENT
    ASSUME DS:DATA, CS:CODE, SS:STACK
INICIO: mov AX , DATA
    mov DS, AX
            mov cx,15
            mov si,0        ;contador y apuntador

            mov ah,09h
            lea dx,txt      ;pedir el nombre
            int 21h

leer:   mov ah,01h      ;introduce caracter x caracter
            int 21h
            cmp al,13       ;si es enter, brinca a la siguiente instruccion
            je pantalla
            mov nombre [si], al     ;almacenamos en la cada lo del puntero
            inc si                  ;incremetamos el puntero para seguir leyendo
caracteres
            loop leer

Pantalla:
            mov ah,06h
            mov bh,81h
    mov al,00h
    mov cx,00h
    mov dx,3030h
```

```
    int 10h

            mov ah,02h
            mov dh,6            ;fila
            mov dl,0            ;columna
            mov bh,0            ;pagina
            int 10h

Consecutivos1:
            mov ah,09h
            lea dx, espacio
            int 21h

            mov ah,07h
            int 21h

            mov bl,var
            cmp al,bl
            je      txt1
            jne FinA        ;si no coincide de manera consecutiva, ira al final alternativo

Consecutivos2:
            mov ah,09h
            lea dx, espacio
            int 21h

            mov ah,07h
            int 21h
```

```
                mov bl,var
                cmp al,bl
                je      txt2
                jne FinA        ;si no coincide de manera consecutiva, ira al final alternativo

Consecutivos3:
                mov ah,09h
                lea dx, espacio
                int 21h

                mov ah,07h
                int 21h

                mov bl,var
                cmp al,bl
                je      txt3
                jne FinA        ;si no coincide de manera consecutiva, ira al final alternativo

Consecutivos4:
                mov ah,09h
                lea dx, espacio
                int 21h

                mov ah,07h
                int 21h

                mov bl,var
                cmp al,bl
                je      txt4
```

```
                jne FinA        ;si no coincide de manera consecutiva, ira al final alternativo

Consecutivos5:
                mov ah,09h
                lea dx, espacio
                int 21h

                mov ah,07h
                int 21h

                mov bl,var
                cmp al,bl
                je      txt5
                jne FinA        ;si no coincide de manera consecutiva, ira al final alternativo

Consecutivos6:
                mov ah,09h
                lea dx, espacio
                int 21h

                mov ah,07h
                int 21h

                mov bl,var
                cmp al,bl
                je      txt6
                jne FinA        ;si no coincide de manera consecutiva, ira al final alternativo

Consecutivos7:
```

```
                mov ah,09h
                lea dx, espacio
                int 21h

                mov ah,07h
                int 21h

                mov bl,var
                cmp al,bl
                je      txt7
                jne FinA        ;si no coincide de manera consecutiva, ira al final alternativo

Consecutivos8:
                mov ah,09h
                lea dx, espacio
                int 21h

                mov ah,07h
                int 21h

                mov bl,var
                cmp al,bl
                je      txt8
                jne FinA        ;si no coincide de manera consecutiva, ira al final alternativo

Consecutivos9:
                mov ah,09h
                lea dx, espacio
                int 21h
```

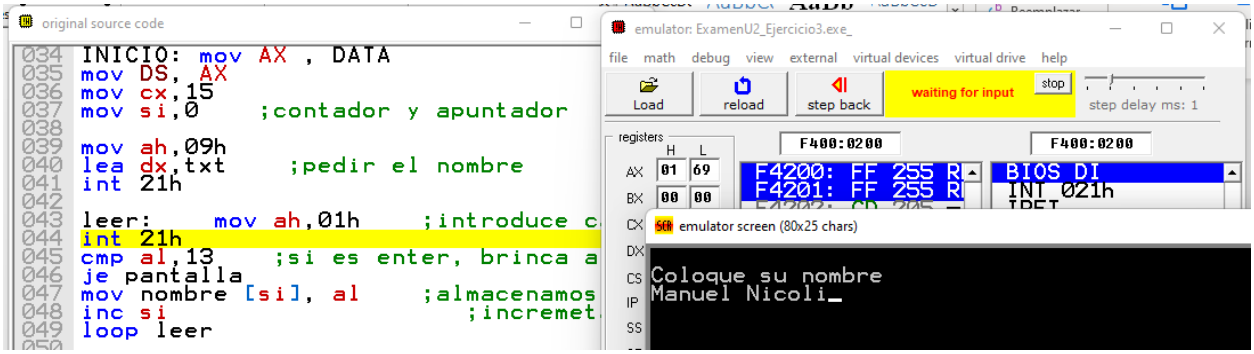
```
                mov ah,07h
                int 21h

                mov bl,var
                cmp al,bl
                je      txt9
                jne FinA        ;si no coincide de manera consecutiva, ira al final alternativo

txt1:   mov ah,09h
                lea dx, fig1
                int 21h
                mov bl,var
                add bl,1
                mov var,bl
                jmp Consecutivos2

txt2:   mov ah,09h
                lea dx, fig2
                int 21h
                mov bl,var
                add bl,1
                mov var,bl
                jmp Consecutivos3

txt3:   mov ah,09h
                lea dx, fig3
                int 21h
                mov bl,var
```

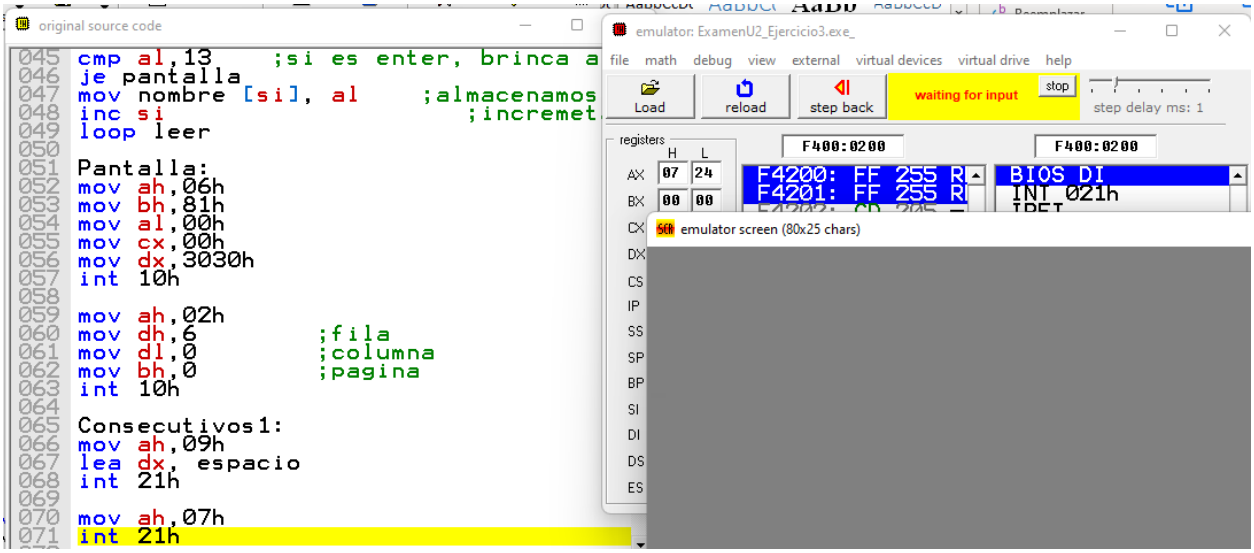
```
                add bl,1
                mov var,bl
                jmp Consecutivos4

txt4:   mov ah,09h
                lea dx, fig4
                int 21h
                mov bl,var
                add bl,1
                mov var,bl
                jmp Consecutivos5

txt5:   mov ah,09h
                lea dx, fig5
                int 21h
                mov bl,var
                add bl,1
                mov var,bl
                jmp Consecutivos6

txt6:   mov ah,09h
                lea dx, fig6
                int 21h
                mov bl,var
                add bl,1
                mov var,bl
                jmp Consecutivos7

txt7:   mov ah,09h
```

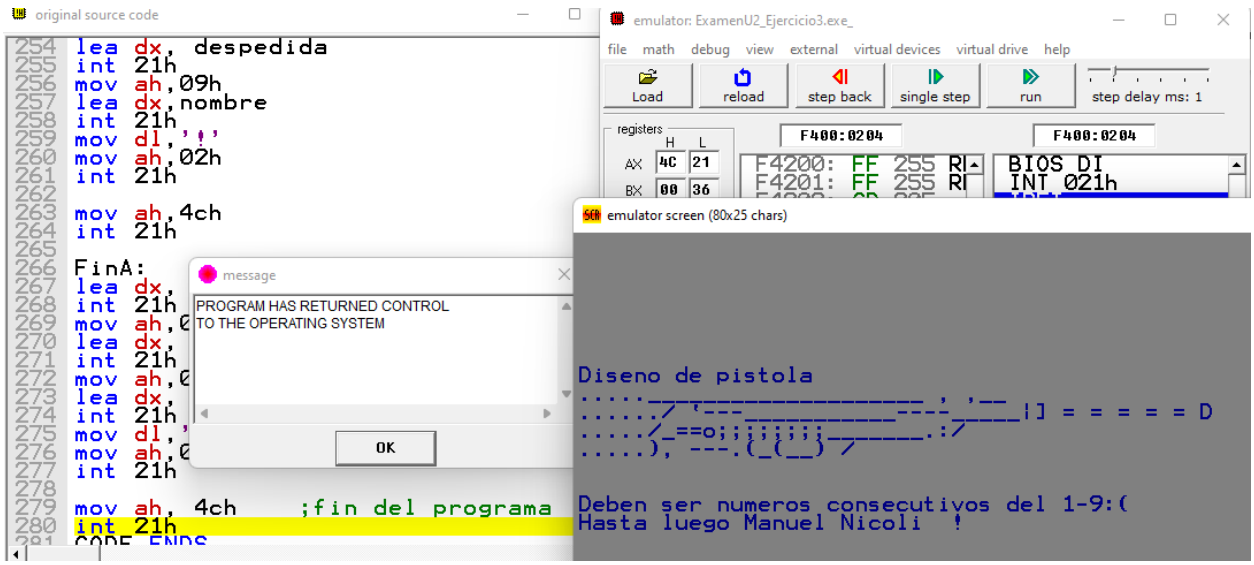
```
                lea dx, fig7
                int 21h
                mov bl,var
                add bl,1
                mov var,bl
                jmp Consecutivos8

txt8:   mov ah,09h
                lea dx, fig8
                int 21h
                mov bl,var
                add bl,1
                mov var,bl
                jmp Consecutivos9

txt9:   mov ah,09h
                lea dx, fig9
                int 21h

Fin:    mov ah,09h
        lea dx, espacio
        int 21h
        mov ah,09h
                lea dx, despedida
                int 21h
                mov ah,09h
                lea dx,nombre
                int 21h
                mov dl,'!'
```

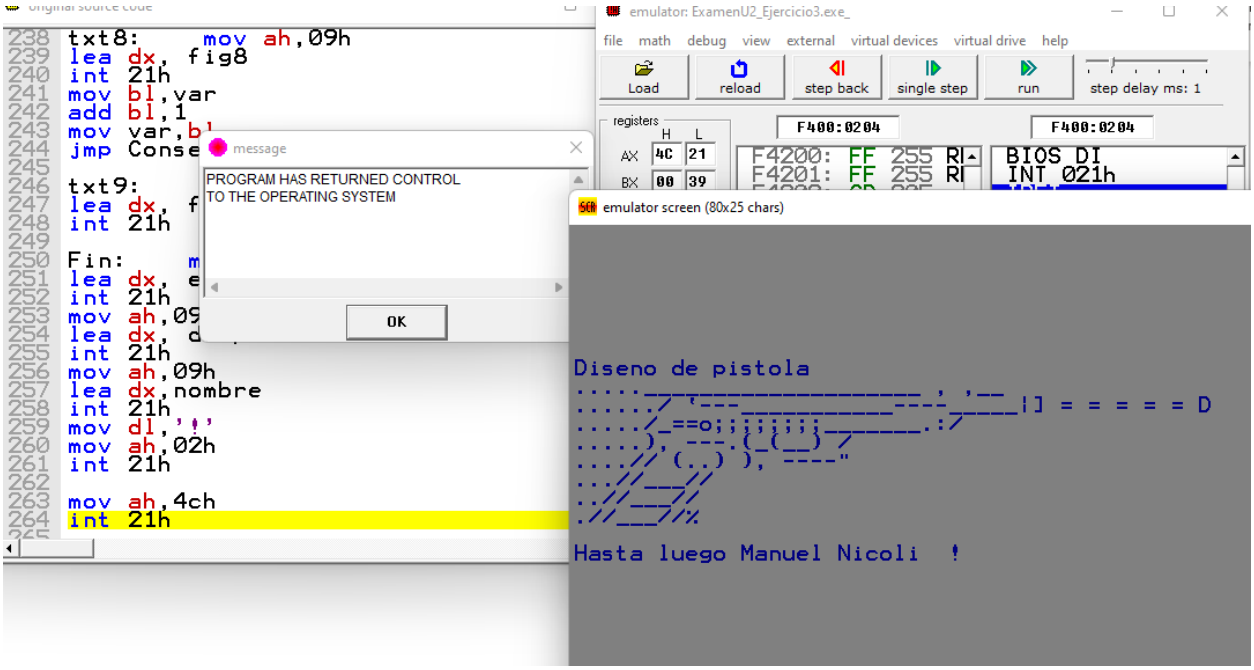
```
                mov ah,02h
                int 21h

                mov ah,4ch
                int 21h

FinA:   mov ah,09h
        lea dx, espacio
        int 21h
        mov ah,09h
                lea dx, alternativo
                int 21h
                mov ah,09h
                lea dx, nombre
                int 21h
                mov dl,'!'
                mov ah,02h
                int 21h

                mov ah, 4ch     ;fin del programa
        int 21h
CODE ENDS
ENDS INICIO
```

CAPTURAS DE FUNCIONAMIENTO.








txt8: mov ah,09h
lea dx, fig8
int 21h
mov bl,var
add bl,1
mov var,bl
jmp Conse
txt9:
lea dx, f
int 21h
Fin:
lea dx, e
int 21h
mov ah,09
lea dx, d
int 21h
mov ah,09h
lea dx,nombre
int 21h
mov dl,'!'
mov ah,02h
int 21h
mov ah,4ch
int 21h
message
PROGRAM HAS RETURNED CONTROL
TO THE OPERATING SYSTEM
OK
emulator: ExamenU2_Ejercicio3.exe_
file math debug view external virtual devices virtual drive help
Load
reload
step back
single step
run
step delay ms: 1
registers
H L
AX 4C 21
BX 00 39
F400:0204
F4200: FF 255 RI
F4201: FF 255 RI
F400:0204
BIOS DI
INT 021h
emulator screen (80x25 chars)
Diseno de pistola
Hasta luego Manuel Nicoli !