

Manual Técnico

Marvin Eduardo Catalán Véliz

Universidad de San Carlos de Guatemala

Facultad de Ingeniería

Escuela de Ciencias y Sistemas

Arquitectura de Computadores y Ensambladores 1 Sección A

Ing. Otto Rene Escobar

Aux. Oscar Rolando Bernard Peralta

MANUAL TECNICO PROYECTO FINAL

Marvin Eduardo Catalán Véliz

201905554

Guatemala, 01 de abril de 2022

INDICE

REQUERIMIENTOS TÉCNICOS PARA SU EJECUCIÓN.....	3
HERRAMIENTAS UTILIZADAS PARA EL DESARROLLO	3
DOSBOX.....	3
MASM 6.11.....	3
EDITOR DE TEXTO	4
MAQUINA UTILIZADA.....	5
SISTEMA OPERATIVO UTILIZADO.....	5
FUNCIONALIDA Y EXPLICACIÓN.....	6
INICIALIZAR EL PROYECTO.....	6
EXPLICACION main.asm y macros.asm.....	9
EXPLICACION main.asm.....	9
.data	9
.code	10
EXPLICACION macros.asm.....	11

REQUERIMIENTOS TÉCNICOS PARA SU EJECUCIÓN

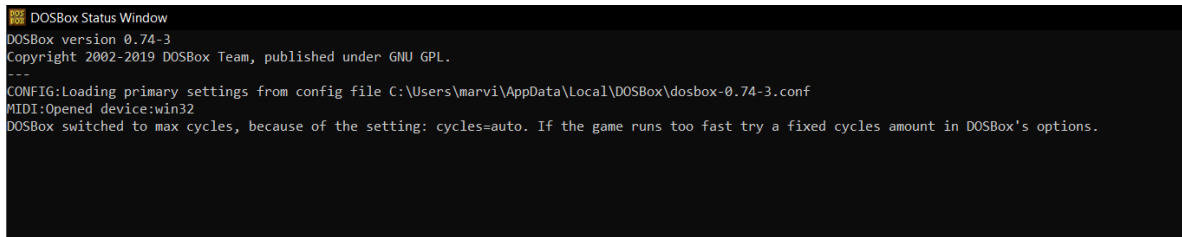
HERRAMIENTAS UTILIZADAS PARA EL DESARROLLO

DOSBOX

¿QUÉ ES?

Es un emulador que recrea un entorno similar al sistema MS-DOS con el objetivo de poder ejecutar programas y videojuegos originalmente escritos para dicho sistema en computadoras más modernas o en diferentes arquitecturas (como Power PC). También permite que estos juegos funcionen en otros sistemas operativos como GNU/Linux. Fue hecho porque Windows XP ya no se basa en MS-DOS y pasó a basarse a Windows NT.

VERSION DOSBOX UTILIZADA



```
DOSBox Status Window
DOSBox version 0.74-3
Copyright 2002-2019 DOSBox Team, published under GNU GPL.
--
CONFIG:Loading primary settings from config file C:\Users\marvi\AppData\Local\DOSBox\dosbox-0.74-3.conf
MIDI:Opened device:win32
DOSBox switched to max cycles, because of the setting: cycles=auto. If the game runs too fast try a fixed cycles amount in DOSBox's options.
```

MASM 6.11

¿QUÉ ES?

El Microsoft Macro Assembler (MASM) es un ensamblador para la familia x86 de microprocesadores. Fue producido originalmente por Microsoft para el trabajo de desarrollo en su sistema operativo MS-DOS, y fue durante cierto tiempo el ensamblador más popular disponible para ese sistema operativo. El MASM soportó una amplia variedad de facilidades para macros y programación estructurada, incluyendo construcciones de alto nivel para bucles, llamadas a procedimientos y alternación (por lo tanto, MASM es un ejemplo de un ensamblador de alto nivel). Versiones posteriores agregaron la capacidad de producir programas para los sistemas operativos Windows.

MASM es una de las pocas herramientas de desarrollo de Microsoft para las cuales no había versiones separadas de 16 bits y 32 bits.

VERSION MASM UTILIZADA



```
C:\>masm
Microsoft (R) MASM Compatibility Driver Version 6.11
Copyright (C) Microsoft Corp 1993. All rights reserved.
```

EDITOR DE TEXTO

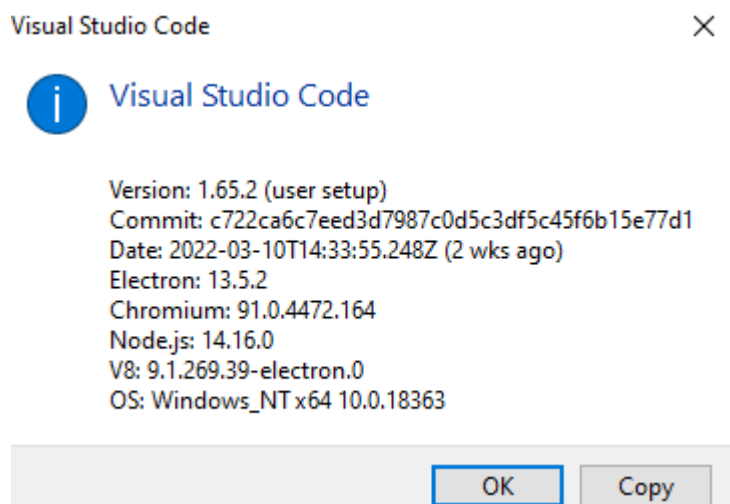
¿QUÉ ES?

Editor de texto es un programa informático que permite crear y modificar archivos digitales compuestos únicamente por textos sin formato, conocidos comúnmente como archivos de texto o "texto plano". El programa lee el archivo e interpreta los bytes leídos según el código de caracteres que usa el editor. Es comúnmente de 7- u 8-bits en ASCII o UTF-8, rara vez EBCDIC.

EDITOR DE TEXTO UTILIZADO: VISUAL STUDIO CODE

Es un editor de código fuente que permite trabajar con diversos lenguajes de programación, admite gestionar tus propios atajos de teclado y refactorizar el código. Es gratuito, de código abierto y nos proporciona una utilidad para descargar y gestionar extensiones con las que podemos personalizar y potenciar esta herramienta.

Las extensiones de Visual Studio Code nos otorgan infinidad de opciones, como colorear tabulaciones, etiquetas o recomendaciones de autocompletado. También hay extensiones que nos ayudan con el lenguaje de programación que vayamos a usar, como por ejemplo para Python, C / C++, JavaScript, etc.

VERSION VSCODE

MAQUINA UTILIZADA

Especificaciones del dispositivo

Nombre del dispositivo	DESKTOP-KMQA5BN
Procesador	Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz 2.71 GHz
RAM instalada	12.0 GB (11.9 GB utilizable)
Id. del dispositivo	369A16F7-DB15-4299-B4CD-780C88B080B8
Id. del producto	00325-96001-12028-AAOEM
Tipo de sistema	Sistema operativo de 64 bits, procesador x64
Lápiz y entrada táctil	Compatibilidad con entrada táctil con 10 puntos táctiles

SISTEMA OPERATIVO UTILIZADO

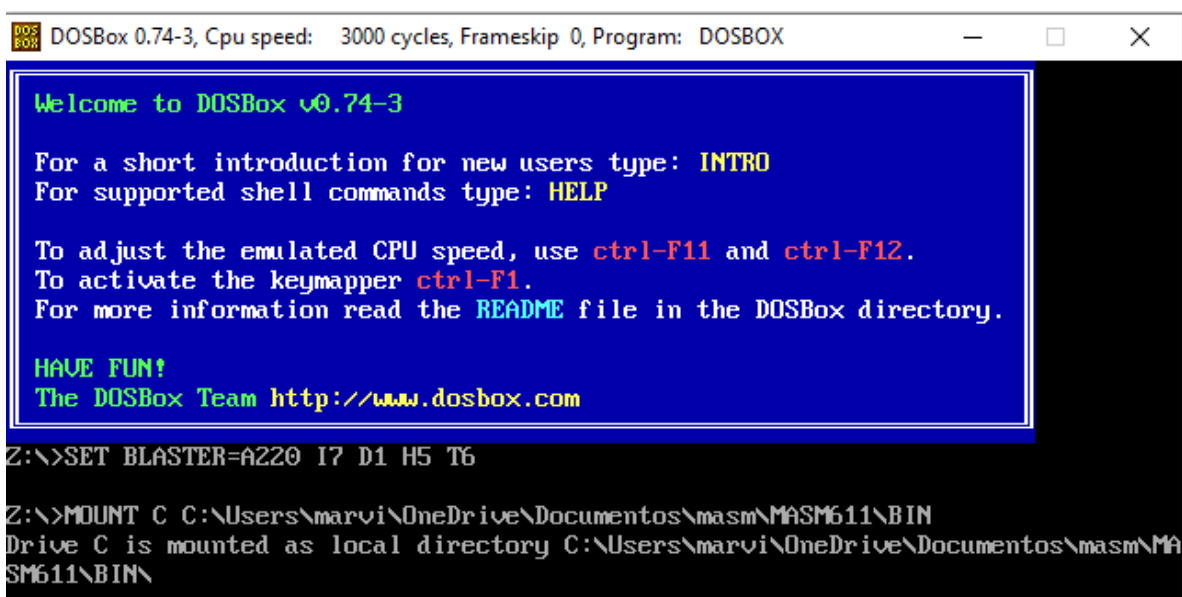
Especificaciones de Windows	
Edición	Windows 10 Home
Versión	1909
Se instaló el	15/07/2019
Compilación del SO	18363.1556

FUNCIONALIDA Y EXPLICACIÓN

El programa esta basado en lenguaje ensamblador MASM versión 6.11 que se divide en dos archivos principales que son main.asm y macros.asm posteriormente explicaremos de una mejor manera estos dos archivos, pero vamos veremos primero lo necesario para inicializar el proyecto.

INICIALIZAR EL PROYECTO

Como se mencionó en las herramientas utilizadas se creó en Dosbox por lo cual principalmente debemos abrir Dosbox y montar un disco nuevo con la dirección donde tengamos ubicada la carpeta BIN del MASM611



```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DOSBOX

Welcome to DOSBox v0.74-3

For a short introduction for new users type: INTRO
For supported shell commands type: HELP

To adjust the emulated CPU speed, use ctrl-F11 and ctrl-F12.
To activate the keymapper ctrl-F1.
For more information read the README file in the DOSBox directory.

HAVE FUN!
The DOSBox Team http://www.dosbox.com

Z:\>SET BLASTER=A220 I7 D1 H5 T6

Z:\>MOUNT C C:\Users\marvi\OneDrive\Documentos\masm\MASM611\BIN
Drive C is mounted as local directory C:\Users\marvi\OneDrive\Documentos\masm\MA
SM611\BIN\
```

La distribución de teclado utilizada en MS-DOS era un poco distinta a nuestras distribuciones actuales por lo cual posteriormente a montar el disco en la carpeta BIN del MASM cambiaremos la distribución del teclado (Este paso no es obligatorio, yo como desarrollador prefiero la distribución en inglés) con el comando keyb seguido de la distribución requerida.



```
Z:\>keyb us
Keyboard layout us loaded for codepage 437
```

En el siguiente enlace pueden encontrar la distribución que más les parezca o con la que más cómodos se sientan como desarrolladores. KEYB - DOSBoxWiki

Posteriormente a cambiar la distribución ingresaremos al disco montado, y podemos comprobar que efectivamente nos encontramos en un disco con MS-DOS.

```
Z:\>C:
C:\>masm
Microsoft (R) MASM Compatibility Driver Version 6.11
Copyright (C) Microsoft Corp 1993. All rights reserved.
```

Ahora que ya tenemos el disco montado con un ensamblador MASM podemos ejecutar el archivo main para que creamos un código objeto el cuál posteriormente será el archivo utilizado para generar un ejecutable.

Para poder compilar el código objeto lo realizamos con el comando *masm* seguido de la ruta donde se encuentra nuestro código con extensión *asm*.

```
C:\>masm P3EC/main.asm
Microsoft (R) MASM Compatibility Driver
Copyright (C) Microsoft Corp 1993. All rights reserved.

Invoking: ML.EXE /I. /Zm /c /Ta P3EC/main.asm

Microsoft (R) Macro Assembler Version 6.11
Copyright (C) Microsoft Corp 1981-1993. All rights reserved.

Assembling: P3EC/main.asm
```

En este momento si el código que realizamos no tiene ningún error nos ensamblara de forma correcta y nos generará en la carpeta bien un archivo de extensión *.obj* que posteriormente utilizaremos para generar el ejecutable.

```
MAIN      OBJ      8,215 25-03-2022  0:02
```

Cuando tengamos listo el código objeto de nuestro programa ya podemos generar nuestro ejecutable con el comando *link* seguido del nombre de el archivo *.obj*.

```
C:\>link main.obj

Microsoft (R) Segmented Executable Linker Version 5.31.009 Jul 13 1992
Copyright (C) Microsoft Corp 1984-1992. All rights reserved.

Run File [main.exe]:
List File [nul.map]:
Libraries [.lib]:
Definitions File [nul.def]:
```

Como podemos observar podemos modificar distintos parámetros del ejecutable a crear pero para este proyecto en específico podemos dejar todo default solamente presionando enter y continuando. Al momento de que se ejecute correctamente observaremos que tenemos ya un ejecutable de extensión .exe en nuestra carpeta BIN.

```
MAIN      EXE      6,702 25-03-2022  0:17
```

En este momento ya realizamos todo el proceso para generar nuestro código objeto y posteriormente generar nuestro ejecutable listo para que lo utilicé el usuario final que lo único que debe hacer es ejecutar el .exe obviamente desde el disco donde se encuentra montado el MASM611.

EXPLICACION MAIN.ASM Y MACROS.ASM

Como se mencionó anteriormente explicaríamos el por qué se utilizaban dos archivos, de igual manera puede el lector de este manual se este preguntando el por qué toda la inicialización se realizó solamente con el archivo *main.asm* esto se debe a que este es el archivo principal y que este por medio de un *include macros.asm* consume todo lo que contiene el otro archivo, esto se realizó para tener un mejor orden y que el código sea un poco más modular. Cabe resaltar que en el archivo *macros.asm* como su nombre lo hace intuir solamente contiene macros que en algún momento consume el main.

EXPLICACION MAIN.ASM

.DATA

Empezamos explicando todo lo que se encuentra dentro de este bloque de código, en este lugar declaramos todas aquellas variables que vamos a utilizar a lo largo de nuestro código, en este lugar se encuentran diferentes tipos de variables como sabemos solo podemos usar variables de tipo byte en MASM pero las podemos declarar de diferentes números de bytes dependiendo para que la utilizaremos, la gran mayoría de variables utilizadas en este proyecto son las más pequeñas que son de tipo db.

```

;Mensaje de Bienvenida
mensaje1 db 0A,"Universidad de San Carlos de Guatemala",0A,"Facultad de Ingenieria",0A,"Escuela de Ciencias y Sistemas",0A,"Arquitectura de Compiladores y Ensambladores",0A,"Seccion A",0A,"Marvin Eduardo Catalan Veliz",0A,"281005554",0A,0A 55 ENTER
auxM1 db "$"
;enter para avanzar
espEnter db 0A,"Para continuar presione tecla enter: $"
;MENU PRINCIPAL
Menu db 0A,"Menu",1A,0A,"===== ",0A,"F1. Login",0A,"F2. Register",0A,"F9. Exit",0A,"$"
; Opcion Incorrecta
opt db 0A,"> Opcion NO VALIDA << ",0A,"$"
;MENSAJE USUO DE EQUIVOCARSE 3 VECES
blockIn db ">> Permission denied << ",0A,">> There where 3 failed login attempts << ",0A,">> Please contact the administrator << ",0A,">> Press Enter to go back to menu << ",0A,"$"
;GENERAL (EFRENDOS)
msguser db " user: ", "$"
msgTop10 db " Top 10 Scoreboard ", "$"
msgMyTop10 db " My Top 10 Scoreboard ", "$"
msgUnlockUser db ">> Unlock User << ", "$"
msgPromoteAdmin db ">> Promote to Admin << ", "$"
msgDemoteAdmin db ">> Demote to Admin << ", "$"
msgBubbleTitle db ">> Bubble Sort << ", "$"
msgMetricTitle db ">> Bubble Metric << ", "$"
msgSpeedTitle db ">> Bubble Speed << ", "$"
msgSignal db "===== ",0A,"$"
;LOGIN
msgLogin db 0A,"=====login",0A,"$"
msgExit db " ",0A,"$"
UserIn db 25 dup (14) ;nombre de usuario a ingresar
userIn db "$"
salto db " ",0A,0A,"$"
PasswIn db 25 dup (14) ;contraseña a ingresar
passIn db "$"
msgInE db 0A,">> User does not exist << ",0A,"$"
msgPInC db 0A,">> Incorrect Password << ",0A,"$"
msgBloqueado db ">> Permission denied << ",0A,">> There where 3 failed login attempts << ",0A,">> Please contact the administrator << ",0A,">> Press Enter to go back to menu << ",0A,"$"
entera db 0A,"$"
;MENU DE USUARIO
msgMenu db " User menu ", "$"
MenuUsuario db "F2. Play game",0A,"F3. Show top 10 scoreboard",0A,"F9. Logout",0A,"$"
;MENU USUARIO ADMIN
msgPA db " Menu usuario Admin", "$"
MenuUsuarioAdmin db "F1. Unlock User",0A,"F2. Show top 10 scoreboard",0A,"F4. Play Game",0A,"F5. Bubble Sort",0A,"F6. Heap Sort",0A,"F7. Tim sort",0A,"F9. Logout",0A,"$"
;MENU DE ADMIN
msgMenuAdmin db "Menu de Admin", "$"
userBloq db ">> Username a desbloquear: $"
userAdmin db ">> Username para ascender a admin: $"
userQuitAdmin db ">> Username para remover admin: $"
msgQuitAdmin db ">> No se puede degradar al admin general<< $"
uNoBlock db ">> Este usuario no necesita desbloqueo << ",0A,"$"
UserIn db ">> El usuario ya es administrador << ",0A,"$"
uNoAdmin db ">> Este usuario no era administrador << ",0A,"$"
UserBloqueado db ">> Se desbloqueo el usuario con exito << $"
UserAdmin db ">> Se ascendio a admin a el usuario << $"
QuitAdmin db ">> Se removio de admin al usuario << $"
MenuAdmin db "F1. Unlock User",0A,"F2. Promote user to admin",0A,"F3. Demote user from admin",0A,"F5. Bubble Sort",0A,"F6. Heap Sort",0A,"F7. Tim sort",0A,"F9. Logout",0A,"$"
;DESCRIPCION
contador db 25 dup (14)
;JUEGO
valort1 db 0
vix db "$"
valort2 db 0
v2x db "$"
auxt db 0
contadort db 0
conttb db 0
StringHMT db 4 dup(14)

```

.CODE

En este bloque se encuentra nuestro procedimiento principal que obligatoriamente se tiene que ejecutar en nuestro programa y es el que se ejecuta al correr el ejecutable. Como primer punto obtenemos todas nuestras variables declaradas en data hacia el registro `ax`, posteriormente a esto inicia la lógica de nuestro proyecto que inicia con un menú de tres opciones.

```

1 include macro.asm
2 .MODEL small
3 .STACK
4 .RADIX 16
5 .DATA
6 ;APARTADO PARA LA DECLARACION DE VARIABLES Y LISTAS
7 #Variables
8 .CODE
9 ;APARTADO PARA EL CODIGO
10 start:
11     main proc
12         call pFlujoProyecto2
13     main endp
14
15 pFlujoProyecto2 proc
16     call pAjustarMemoria
17     ;call pBaseDatos ;comentar esto si no se quiere borrar la base de datos
18     call pLimpiarConsole
19     #MostrarString mensaje1
20     ;apartado de espera de un enter-----
21     call pEspEnter
22     ;-----
23     call pLimpiarConsole
24     call pMenuPrincipal
25     ;call pMenuOrd
26     call pRetControl
27     ret
28 pFlujoProyecto2 endp
29
30 ;PROCED PARA LIMPIAR LA CONSOLA
31 pLimpiarConsole proc
32     push ax
33     push bx
34     push cx
35     push dx
36     ;limpia la consola
37     mov ax,0000h ; es igual a mov ah,06 (scroll up windows con el int 10) y mov al,00
38     mov bh,07
39     mov cx,0000h ; es igual a mov ch,0 mov cl,0 , filas y columnas de derecha a izquierda
40     mov dx,184FH ;filas y columnas de both
41     int 10
42     ;posicione el cursor en la pos 0
43     mov ah,02
44     mov bh,0 ;numero de pagina
45     mov dl,0 ;columna
46     mov dh,0 ;fila
47     int 10
48     pop dx
49     pop cx
50     pop bx
51     pop ax
52     ret
53 pLimpiarConsole endp
54 ;PROC PARA DEVOLVER EL CONTROL AL SISTEMA
55 pRetControl proc
56     mov al,10
57     mov ah,4C
58     int 21
59     ret
60 pRetControl endp
61 ;PROC PARA AJUSTAR LA MEMORIA AL SISTEMA
62 pAjustarMemoria proc
63     mov dx,@DATA
64     mov ds,dx
65     mov es,dx
66     ret
67 pAjustarMemoria endp
68 ;PROC PARA CREAR BASE DE DATOS (USUARIOS Y SCORES)

```

Este segmento es gigante, por lo cual no se podrá explicar al pie de la letra en este documento(y también porque son las 2 am) pero en este se mueve todo el flujo y toda la lógica a nivel superior del programa, a nivel más bajo y específico se hablara posteriormente en las macros. En el código se puede visualizar por los comentarios todo el flujo y su funcionamiento para poder hacer actualizaciones o nuevas implementaciones.

EXPLICACION MACROS.ASM

Como se habló anteriormente este contiene todas las macros utilizadas por el main, estas macros son repetitivas con respecto a instrucciones se trata solo que acomodándolas a la lógica necesaria para poder ejecutar nuestro proyecto en curso, explicar cada una de estas sería meterse a un tema demasiado técnico y repetitivo que se puede resumir en la documentación oficial de masm, donde nos basamos en diferentes instrucciones y muy importantes también interrupciones para poder

realizar las acciones requeridas. Se adjunta una documentación muy completa en la que estoy completamente seguro de que si no terminan de entender el por qué de algún comando o alguna interrupción les dejara todo muy claro. [emu8086 documentation \(yassinebridi.github.io\)](https://yassinebridi.github.io)

```

macro.asm > {} m/variables
272 dia db 4 dup (0)
273 mes db 4 dup (0)
274 año db 4 dup (0)
275 hora db 4 dup (0)
276 min db 4 dup (0)
277 segun db 4 dup (0)
278 year du 0
279 month du 0
280 day du 0
281 hour du 0
282 minutes du 0
283 seconds du 0
284 ;MANEJO DE UN ARCHIVO EXTERNO =====
285 electul db 0 ;variable que contendrá cada elemento leído por el programa
286 a db "s"
287 handler du 0
288 msgregar db 0A,"Ingrese el nombre del archivo a cargar: ","$"
289 namefile db 20 dup(0)
290 ofcaux db "$"
291 cargood db 0A,"Cargo con éxito! (presione cualquier tecla)","$"
292 carbad db 0A,"falló la carga! (presione cualquier tecla)","$"
293 estadocarga db 0 ;si se logra cargar algo o no
294 savegood db 0A,"Guardado con éxito! ","$"
295 savebad db 0A,"No se guardó el archivo!","$"
296 creacionCorrecta db 0 ;si se creo con éxito un nuevo documento su valor será 1, caso contrario será 0
297 poslectura db 0 ;variable con la cual se llega a 0 luego de instanciar la macro readfile significa que
298 ;el documento llegó al final de este
299 idEncontrado db 0 ;se encontró la palabra en especial que se requería
300
301 ;APARTADO PARA LOS ARCHIVOS QUE FUNCIONAN COMO BASE DE DATOS=====
302 usersb db "users.gal",0
303 scoresb db "scores.gal",0
304 RepOrdname db "LASTSORT.REP",0
305 auxarchivo db 0
306 aux1 db "$"
307 ;REPORTES DE ORDENAMIENTO =====,0A
308 sepRepOrden db "-----",0A
309 auxseprep db "$"
310 filaScore db 25 dup (0)
311 aux2 db "$"
312 msgEnter db 0A
313 aux3 db "$"
314 msgType db "Tipo: "
315 msgSentido db "Sentido: "
316 msgFecha db "Fecha: "
317 msgHora db "Hora: "
318 slash db 2fh
319 msgAscen db "Ascendente"
320 msgDescen db "Descendente"
321 msgTitleRep db "Rank Player N Points Time",0A
322 auxTitleRep db "$"
323 msgEspacios db " "

```

variables macro
 ;Mensaje de
 mensaje de
 aux1 db "\$"
 ;enter para
 enter para
 enter db
 ;pero para
 Menu db 0A,
 ; Opcion In

Recorte guardado en