

MANUAL TÉCNICO

PROYECTO FINAL



201602719

CHRISTOFER WILLIAM BORRAYO LOPEZ

ARQUITECTURA DE COMPUTADORES Y ENSAMBLADORES 1

Contenido

[Main.asm 2](#_Toc60663704)

[Data.asm 3](#_Toc60663705)

[Bss.asm 5](#_Toc60663706)

[User.asm 5](#_Toc60663707)

[Juego.asm 8](#_Toc60663708)

[Admin.asm 15](#_Toc60663709)

[Grafica.asm 19](#_Toc60663710)

[Ordenamiento.asm 24](#_Toc60663711)

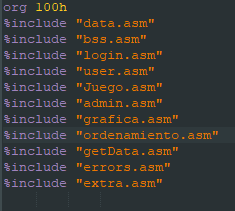
[Extra.asm 28](#_Toc60663712)

[Error.asm 29](#_Toc60663713)

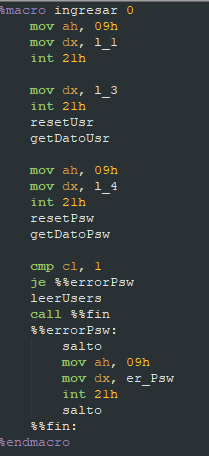
[GetData.asm 30](#_Toc60663714)

# Main.asm

Se incluyeron las importaciones de los demás archivos para mantener un orden y coherencia dentro del código.

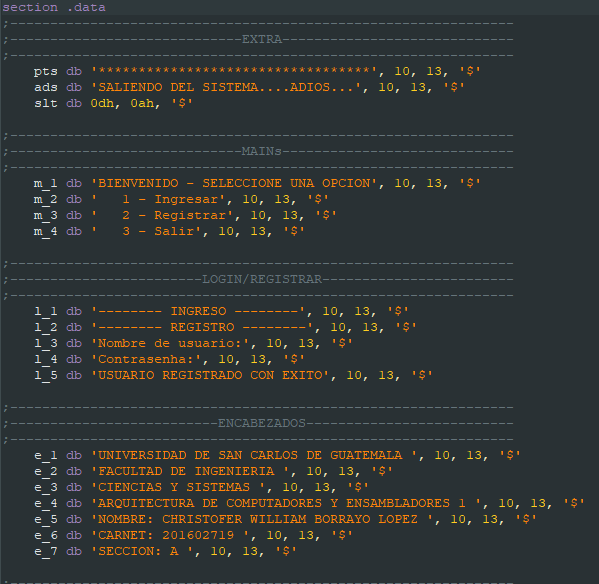


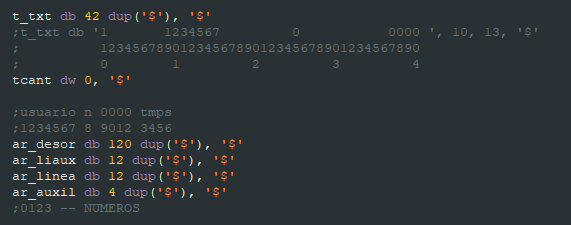
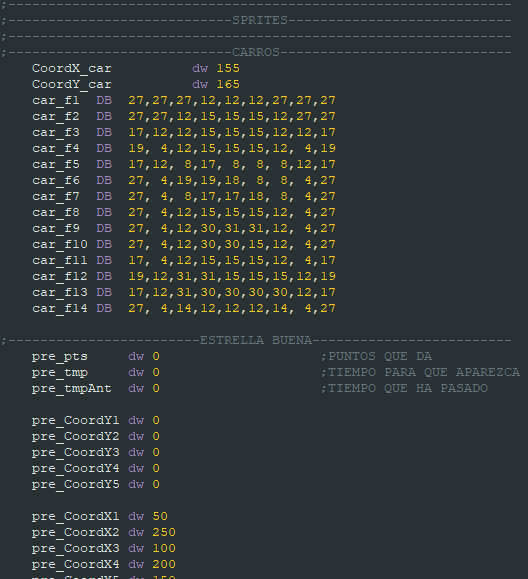
Todos los menús que aparecerán en el programa poseen la misma estructura que el siguiente



# Data.asm

Se incluyeron todos los textos que aparecerán en el programa, así también como vectores, matrices, datos simples, arreglos para formar sprites del juego, rutas, etc.





# Bss.asm

Se incluyó (aunque solo fuese un tipo de dato) para no confundir su función. Es el handler al momento de escribir archivos.



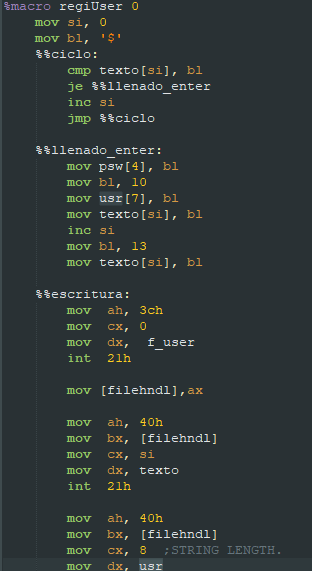
# User.asm

Se puede observar la misma estructura que en el menú de **“Main.asm”**

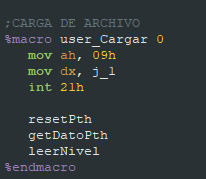


Para registrar usuarios, se hace uso de una matriz llamada **“texto[ ]”** en la cual se almacena los datos del archivo **“usuarios.usr”** y luego se agregan los datos del nuevo usuario.

Se procede a guardar por medio de la interrupción 3ch, también se agrega un carácter de control **‘$’** para su futura lectura o escritura.



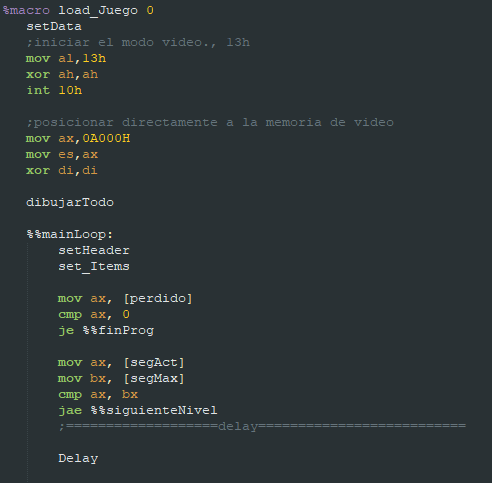
Para cargar el archivo de juego se resetea la información de la ruta para que se pueda ingresar una totalmente nueva, se obtiene la ruta y se procede a leer el archivo. Luego de esto se procede a cargar el juego y cargar los datos fundamentales como las coordenadas del carro, el color de fondo, el mensaje de cabecera y demás.

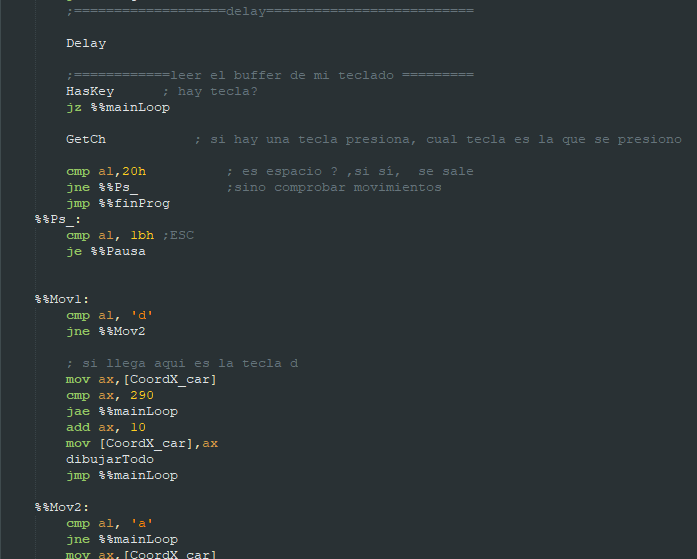


# Juego.asm

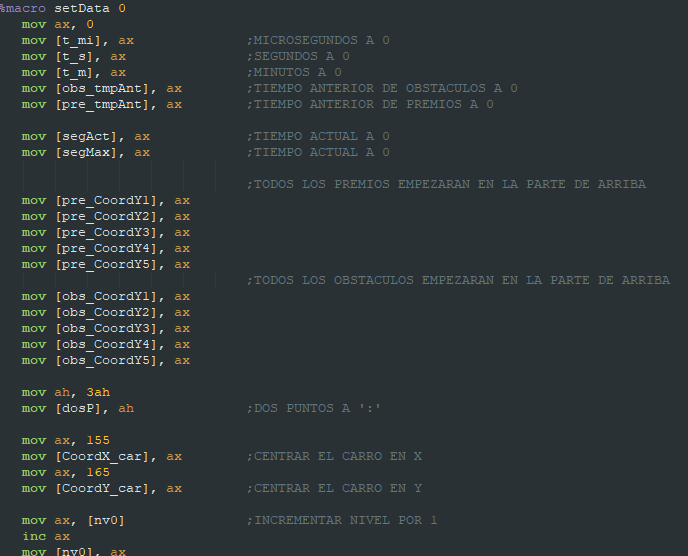
Se utiliza la interrupción 10h con 13h para cargar el modo video, seguidamente se llama al macro **“dibujarTodo”** el cual se encarga de pintar la pantalla por primera vez.

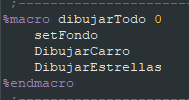
Las siguientes veces la pantalla es refrescada por el macro **“set\_Items”**

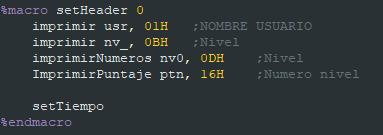




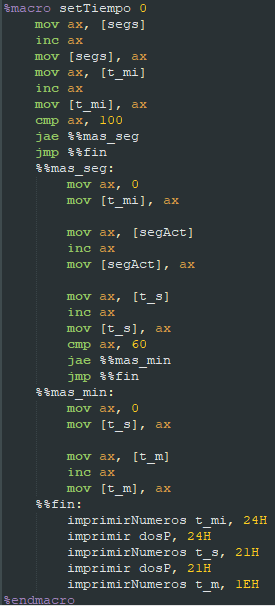
En este macro, la información del juego es puesta para su funcionamiento inicial, esta incluye las coordenadas de las estrellas, coordenadas iniciales del carro, puesta de tiempo a cero, suma de niveles, etc. Este macro es reutilizado al subir de nivel.



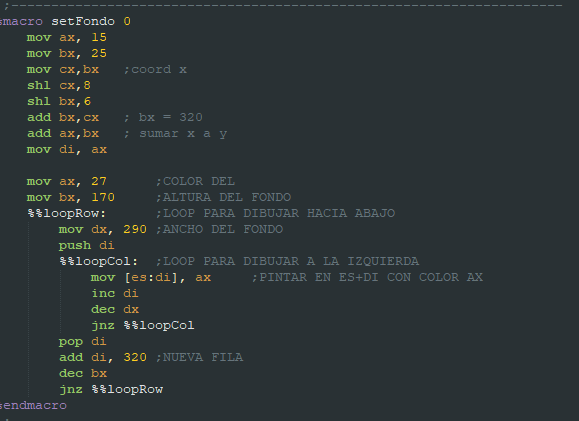


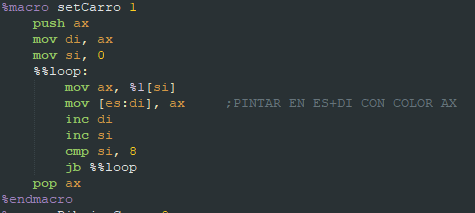


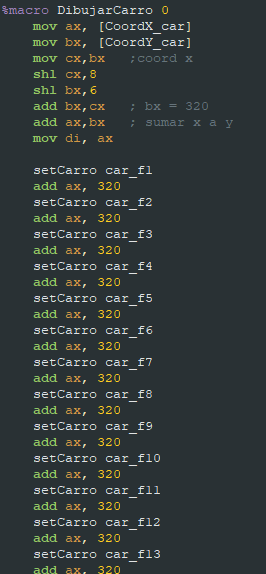
El tiempo es dividido en tres partes, microsegundos, segundos y minutos, estos son aumentados cada delay del programa.

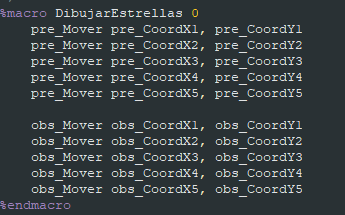


Este es un ejemplo de cómo se gráfica, se inicia con las coordenadas ax como X y bx como Y, a partir de estas se empieza a pintar de izquierda a derecha y arriba abajo. Los colores, altura y ancho del recuadro a pintar se pueden modificar. Lo mismo pasa con el carro. Cabe recordar que para brincar una fila completamente es necesario agregarle a di la cantidad de 320

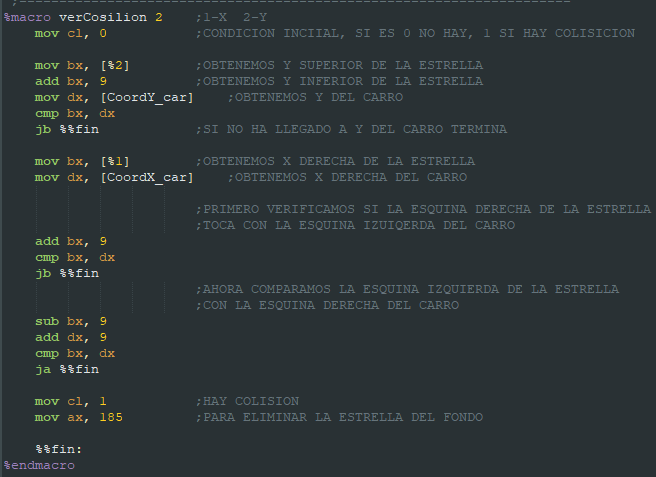




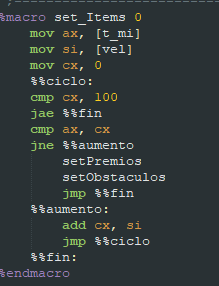




La colisión de las estrellas y el auto se verifica por medio de sus coordenadas, si estas son intersecadas se toma como una colisión y se procede a sumar o restar puntos, dependiendo del objeto con el que colisiono.



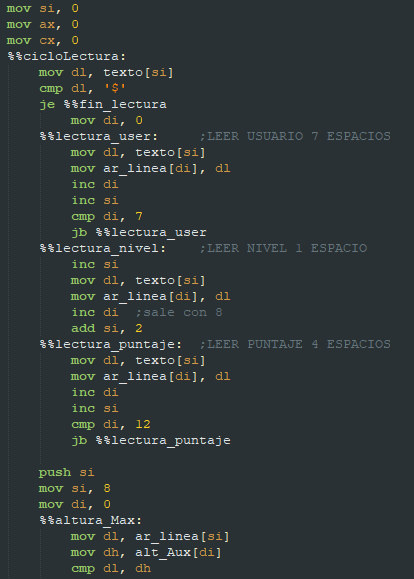
La velocidad de caída de las estrellas esta dada por este macro, el cual simplemente verifica los microsegundos y al cabo de cada cierto múltiplo de un número, este permitirá el movimiento de las estrellas



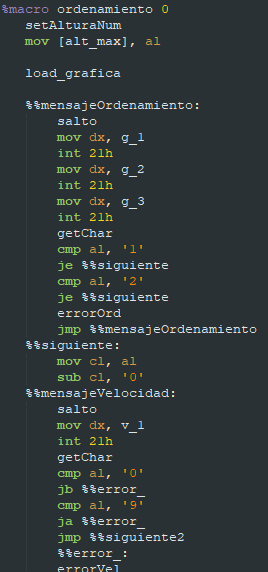
# Admin.asm

Para el administrador se utiliza el mismo concepto que con los usuarios para poder abrir los archivos, la única diferencia es el uso de una matriz de 12x10 para almacenar los datos del top 10, la cual se va llenando con forme se lee el archivo de puntaje de todos los usuarios.





Este macro es simplemente para mostrar las opciones de ordenamiento disponibles

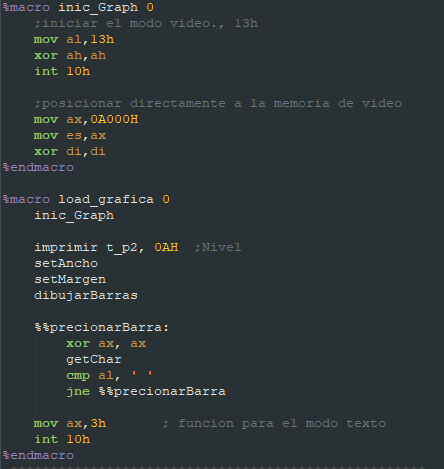


Para guardar el top 10, se utiliza la matriz de 12x10 y se extrae dato por dato, estos son puestos en el archivo y cada vez que se llene un dato entero, por ejemplo, usuario, se procede a colocar una coma y el siguiente dato, en este caso sería el nivel.

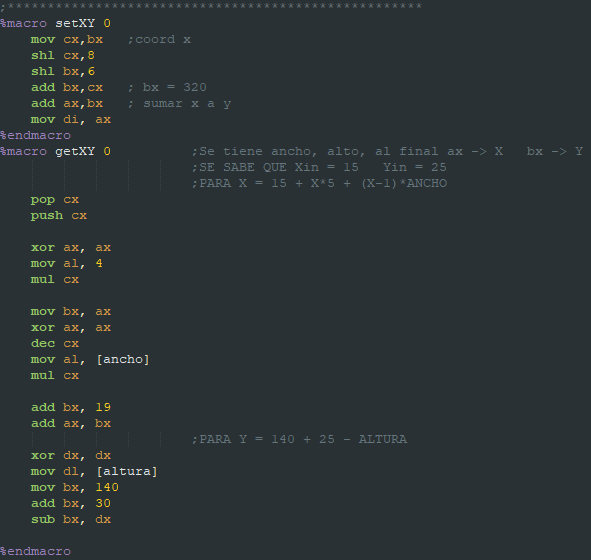


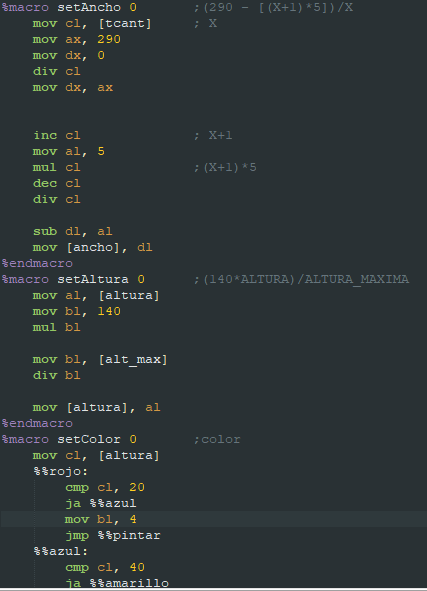
# Grafica.asm

Para la grafica se creo un macro apartado para cargarla dado que se utilizará en diversas ocasiones esta función y para diferentes fines, por lo tanto, se decidió separa el código lo máximo posible

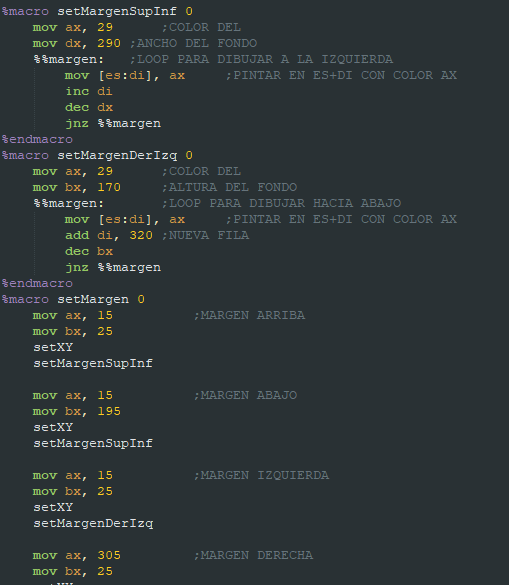


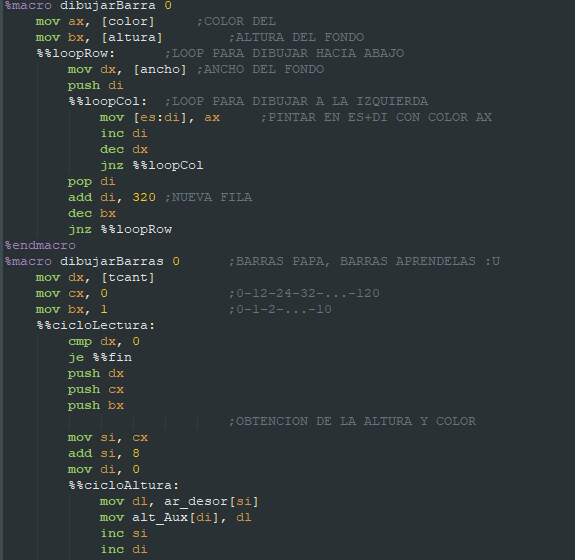
Estos macros ayudan a encontrar la posición X y Y de cada barra en la gráfica, con una formula la cual depende del ancho de cada barra y su alto, los cuales son fácilmente obtenidos dividiendo el tamaño máximo de la pantalla por la cantidad de barras que habrán.





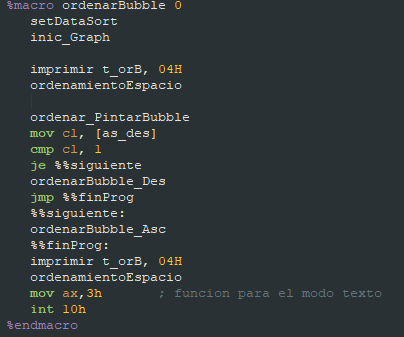
Este macro define los márgenes para la gráfica de barras

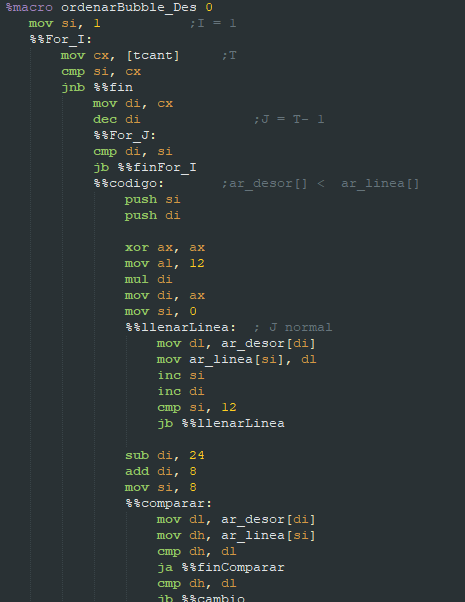


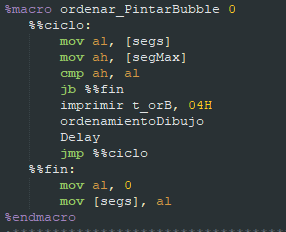


# Ordenamiento.asm

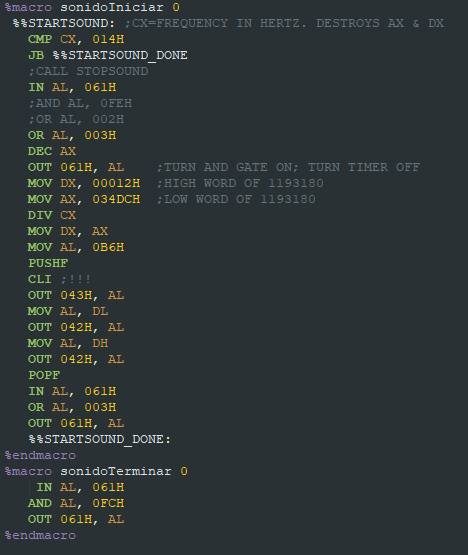
Se pondrá un ejemplo con el ordenamiento de burbuja ya que el Quicksort funciona de la misma manera. La función de esto es simple, se ordena la matriz de 12x10 que contiene los datos a ordenar, esta es ordenada por medio del algoritmo ya definido y por cada movimiento que esta haga, por cada reemplazo o sustitución se procede a encender un sonido, seguido de pintar nuevamente la pantalla, esperar un tiempo, apagar el sonido y proseguir con el ordenamiento.





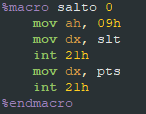


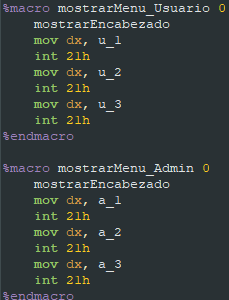




# Extra.asm

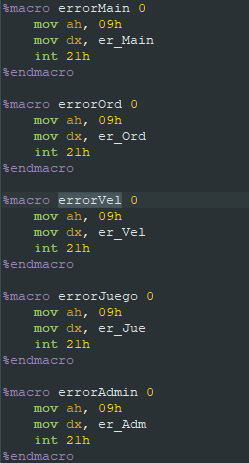
Son los textos que se utilizan mayormente o que su tamaño es considerable pero la importancia no lo es. Por esta razón se optó por crear este archivo y escribir en el todos los encabezado o saltos que se utilizaran.





# Error.asm

Incluye los errores que pueden llegar a existir y los mensajes a mostrar al usuario



# GetData.asm

Se incluye todos los macros orientados a la obtención de información por medio de la entrada del usuario, así como la limpieza de los vectores utilizados para almacenarlas.

