**Trabajo Práctico Final SPD GRUPO C**

Piedra, Papel o Tijera

línea horizontal

# Integrantes

Deutsch Amarfil, Denise  *niss@live.com.ar*

Sosa, Alex *asosa@estudiantes.unsam.edu.ar*

Ganino, Matias *mati.ganino@gmail.com*

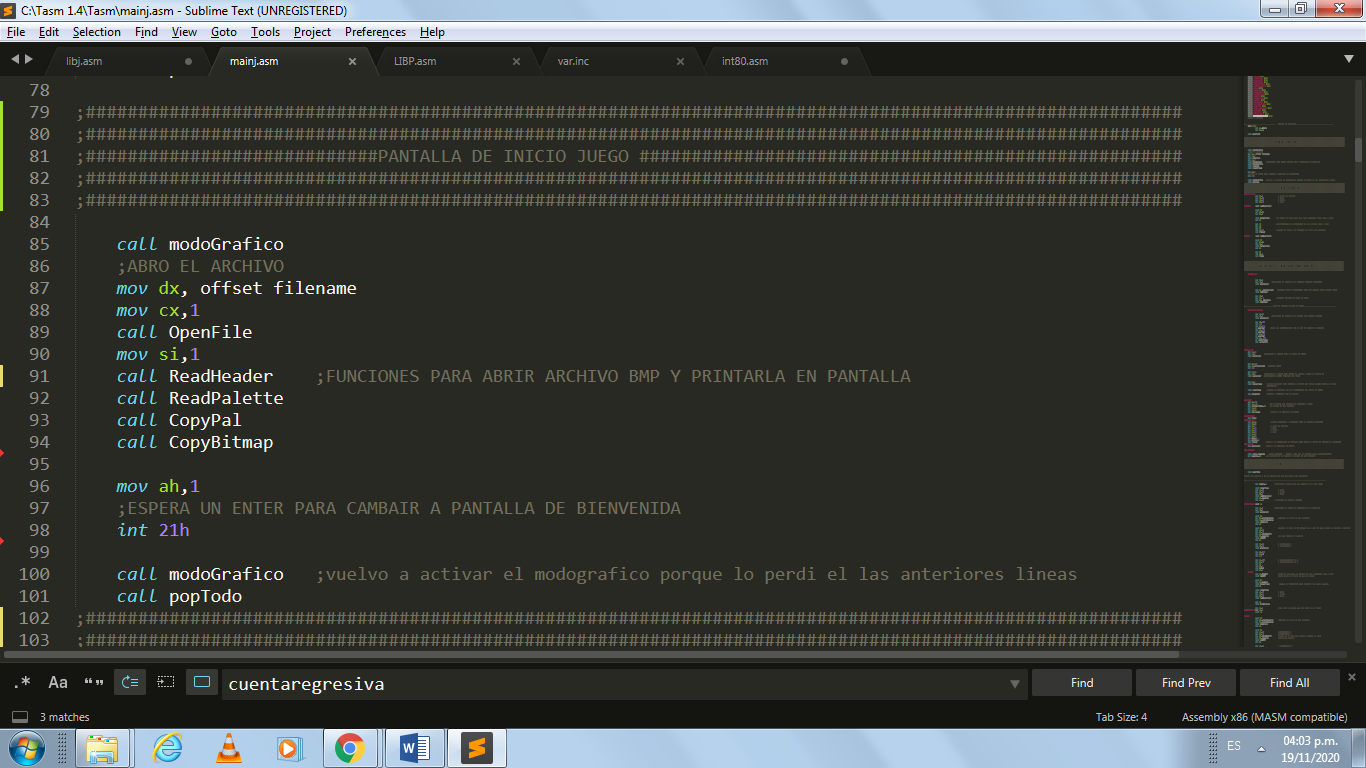
Marquez, Federico *fmarquez@estudiantes.unsam.edu.ar*

Sezaro, Tomas *tomy08\_act@hotmail.com*

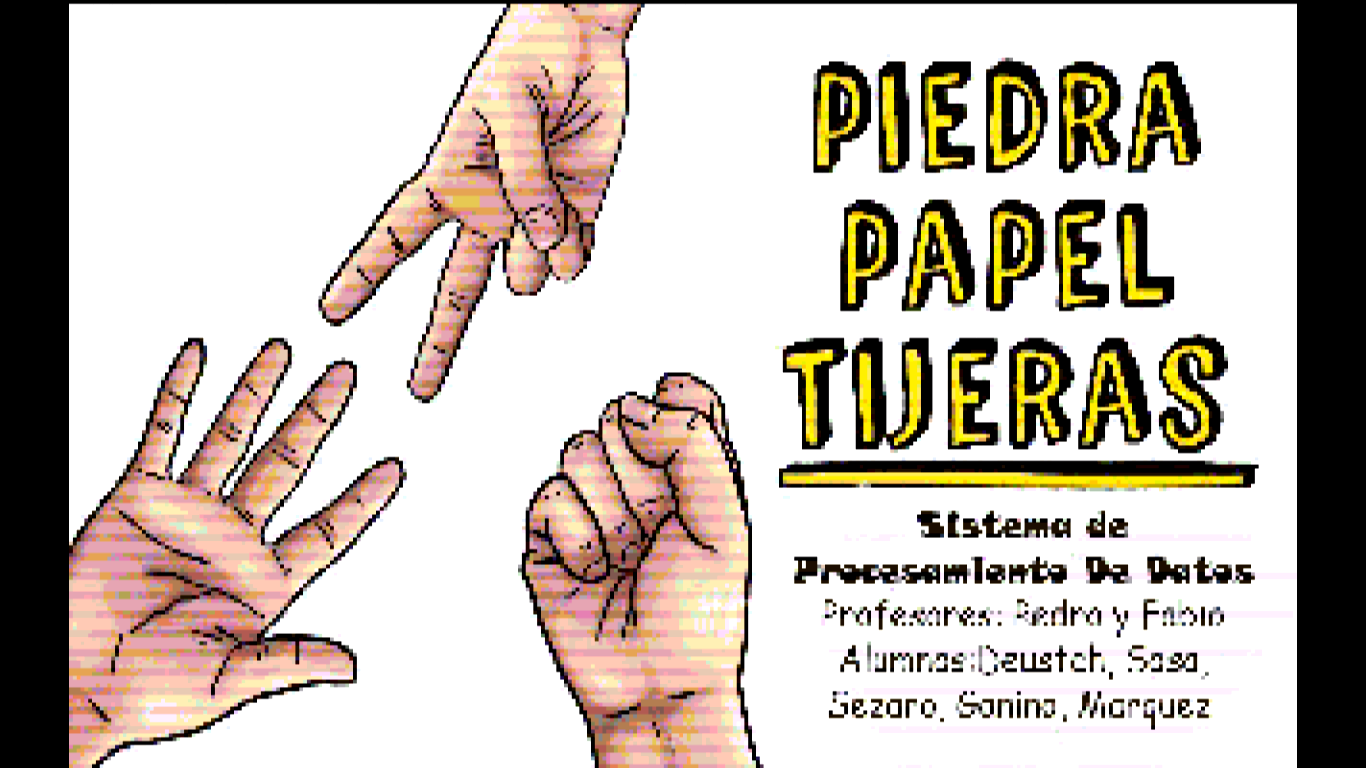
## Juego a Desarrollar

Piedra, Papel o Tijera.

### Descripción



**Primera pantalla**: Imprime la imagen de presentación del juego, usando funciones para abrir el archivo, guardar el encabezado de la imagen (formato bmp), 256 colores y 54 bytes.

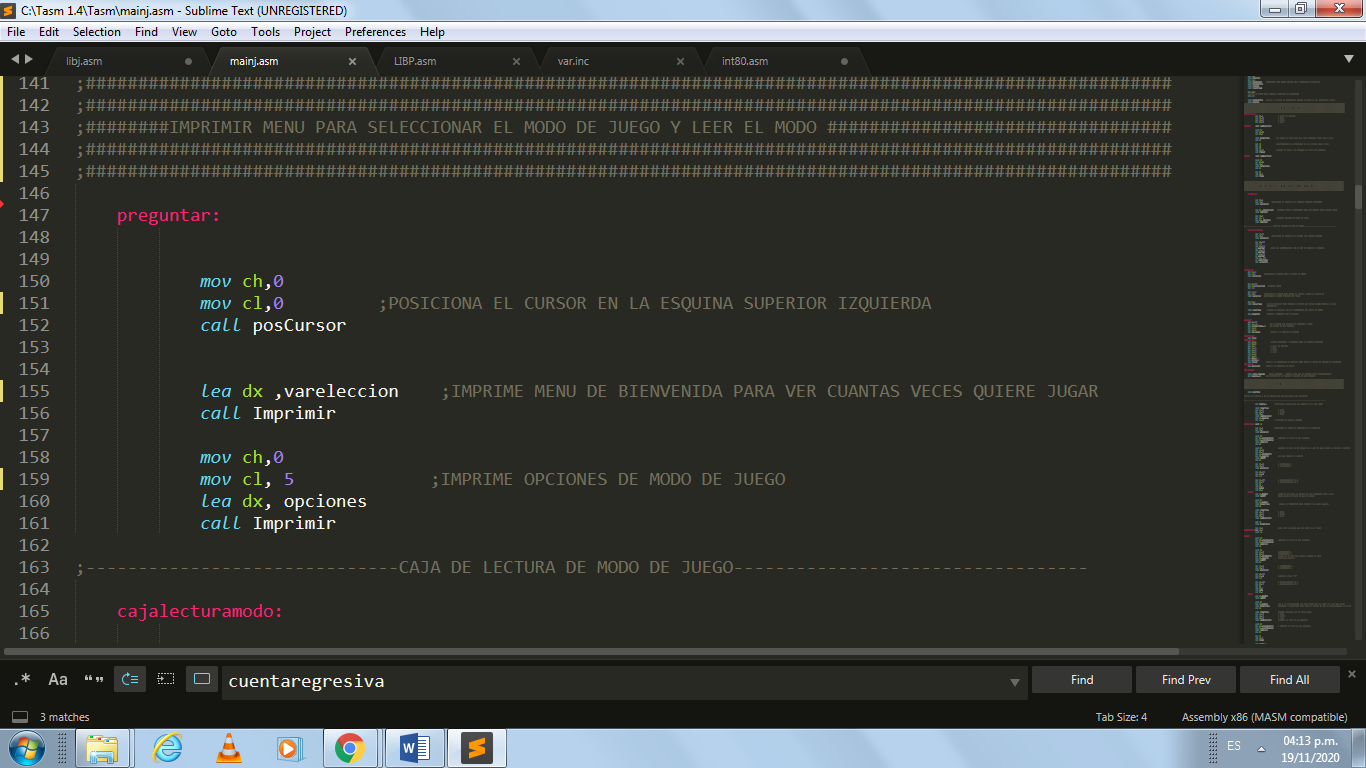


---------------------------------------------------------------------------------------------------------------------------------------------



**Segunda Pantalla**: Transición de color, de negro a azul petróleo (color definido para el juego). Utilizamos una función “cambiaColor” que necesita como parámetros bl = 0(color pantalla anterior, siempre parte del negro), dh = intensidad del color rojo (escala 0-60), ch = intensidad verde (escala 0-60) y cl = intensidad azul (escala 0-60).

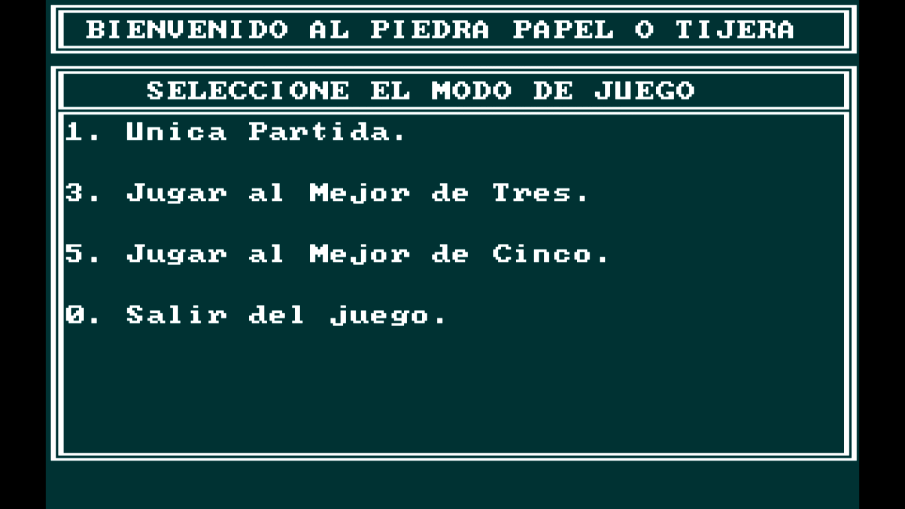
Al ingresarlo en un loop e ir decrementando los colores, se logra un efecto de cambio gradual de color de pantalla.



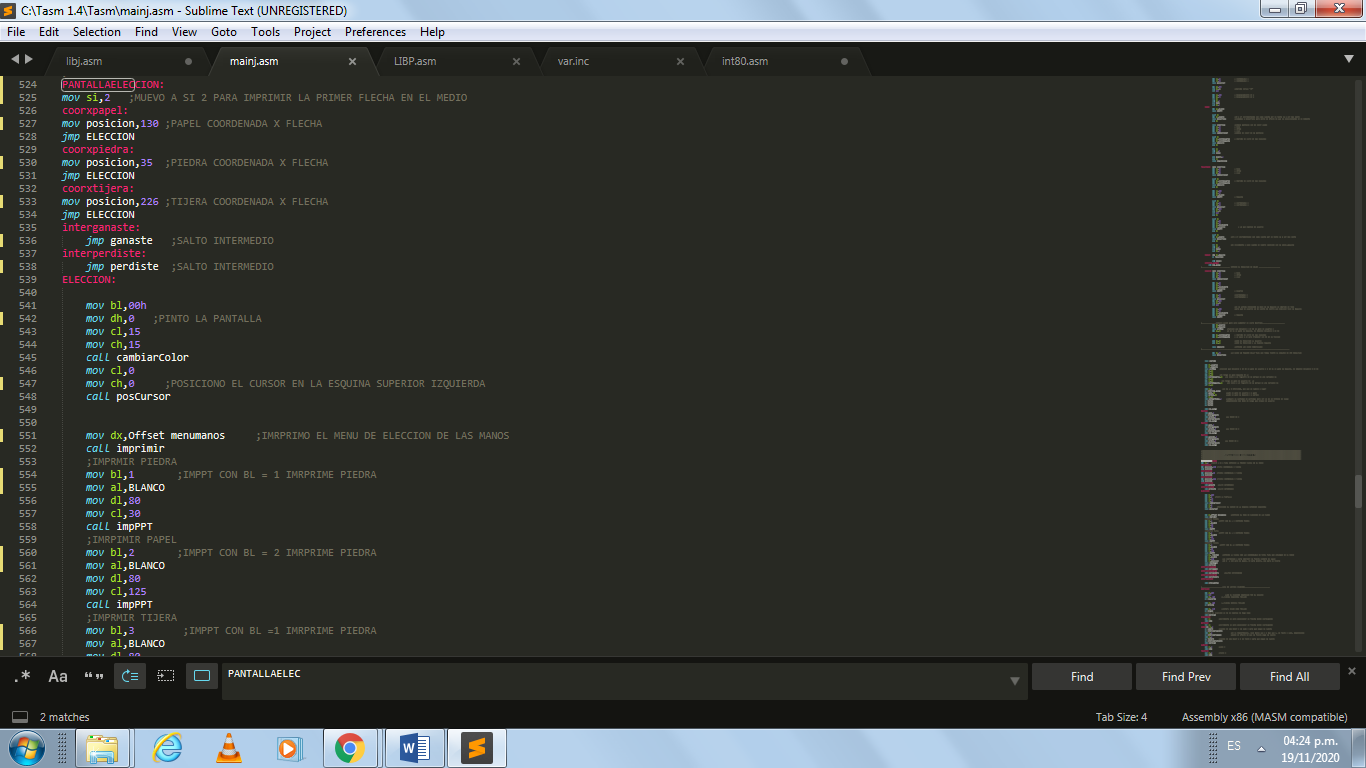
**Tercer** **Pantalla**: Se posiciona el cursor en la esquina superior izquierda y se imprime el menú de opciones del modo de juego (jugar 1 única vez, al mejor de 3, al mejor de 5 y 0 para salir del juego).



Se lee la opción elegida por el usuario y de ser errónea (algo distinto a 1,3,5 o 0) se imprimirá un cartel especificando que su opción es incorrecta, durante un lapso corto de tiempo generando el efecto de desaparecer y aparecer cada vez que el usuario ingresa una tecla no indicada. Si el usuario ingresa una opción correcta a esta se le substraen 30 hexadecimal para dejarlo como valor entero y luego se lleva a una variable para determinar la cantidad de partidas a jugar

.

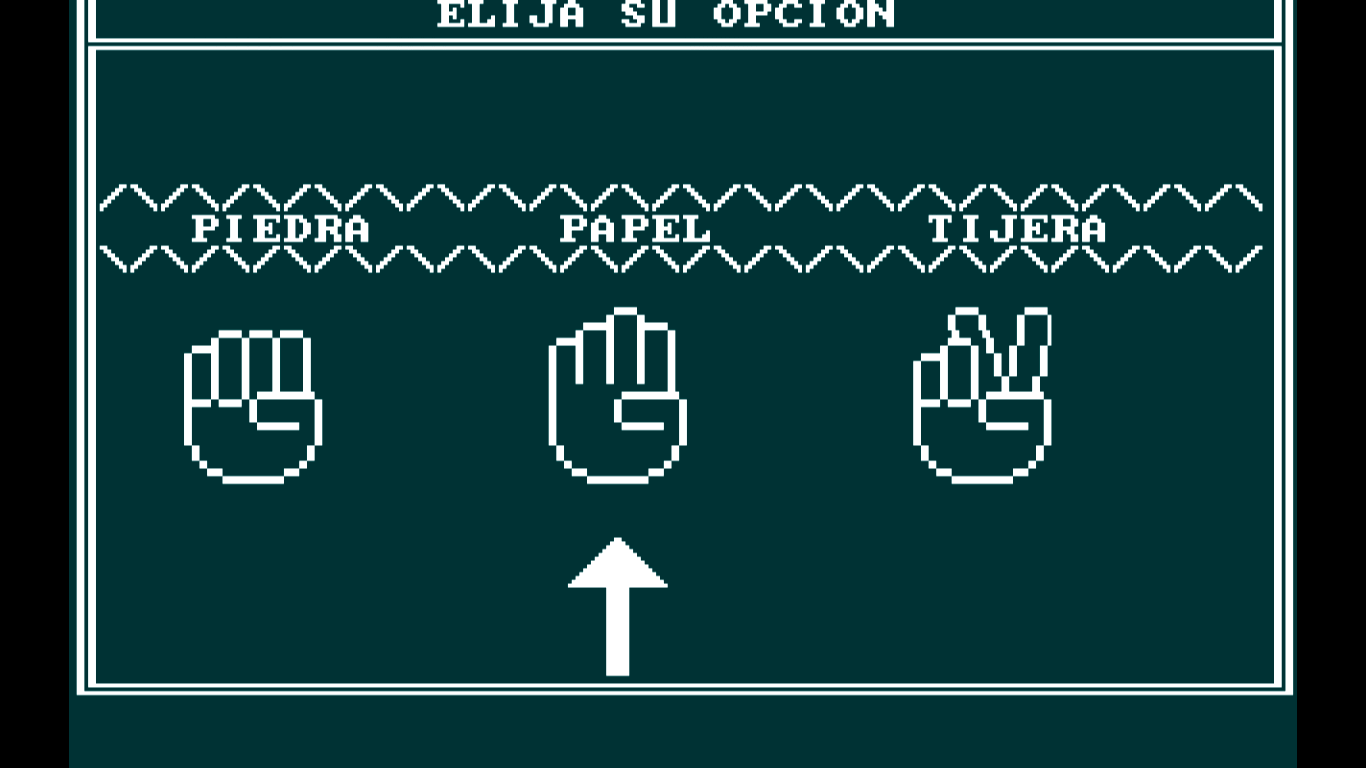
-------------------------------------------------------------------------------------------------------------------------------------------------



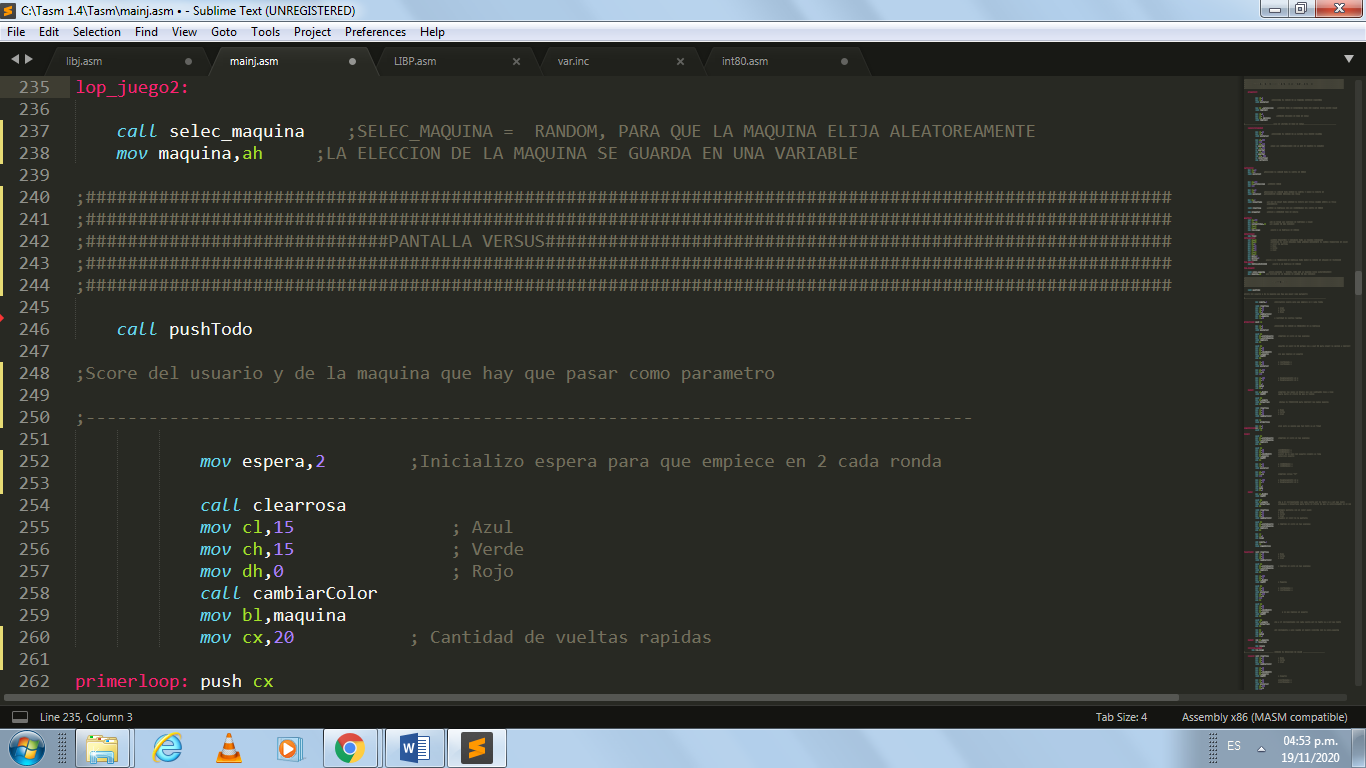
**Pantalla cuatro:** Con el registro SI se le van dando las coordenadas en x en donde imprimir la flecha para la selección del usuario, se cambia el color de la pantalla como ya aclarado en las pantallas anteriores, se imprime el menú con las opciones de piedra, papel o tijera para luego llamar a la función” IMPPT” e imprimir las manos en sus respectivas posiciones.



Aquí, como se aprecia en la imagen, se creó una caja de lectura de scan code que solo acepta la flecha izquierda, flecha derecha y enter, cualquier otra cosa no tiene reacción alguna. El registro SI comienza en 2, por lo cual la flecha selectora se posiciona en el medio de la pantalla, apuntando hacia el papel. Si el usuario ingresa “flecha izquierda”, el registro se decremento en 1 pero si el usuario ingresa más de una vez izquierda se compara el registro contra uno y si es menor se le suma 3, posicionándose en la posición 3 (tijera) generando el efecto de rotación. Lo mismo sucede, pero inversamente cuando el usuario ingresa “flecha derecha” y se pasa de 3, al llegar a 4 se le substraen 3 enteros al registro posicionándolo en 1(piedra).

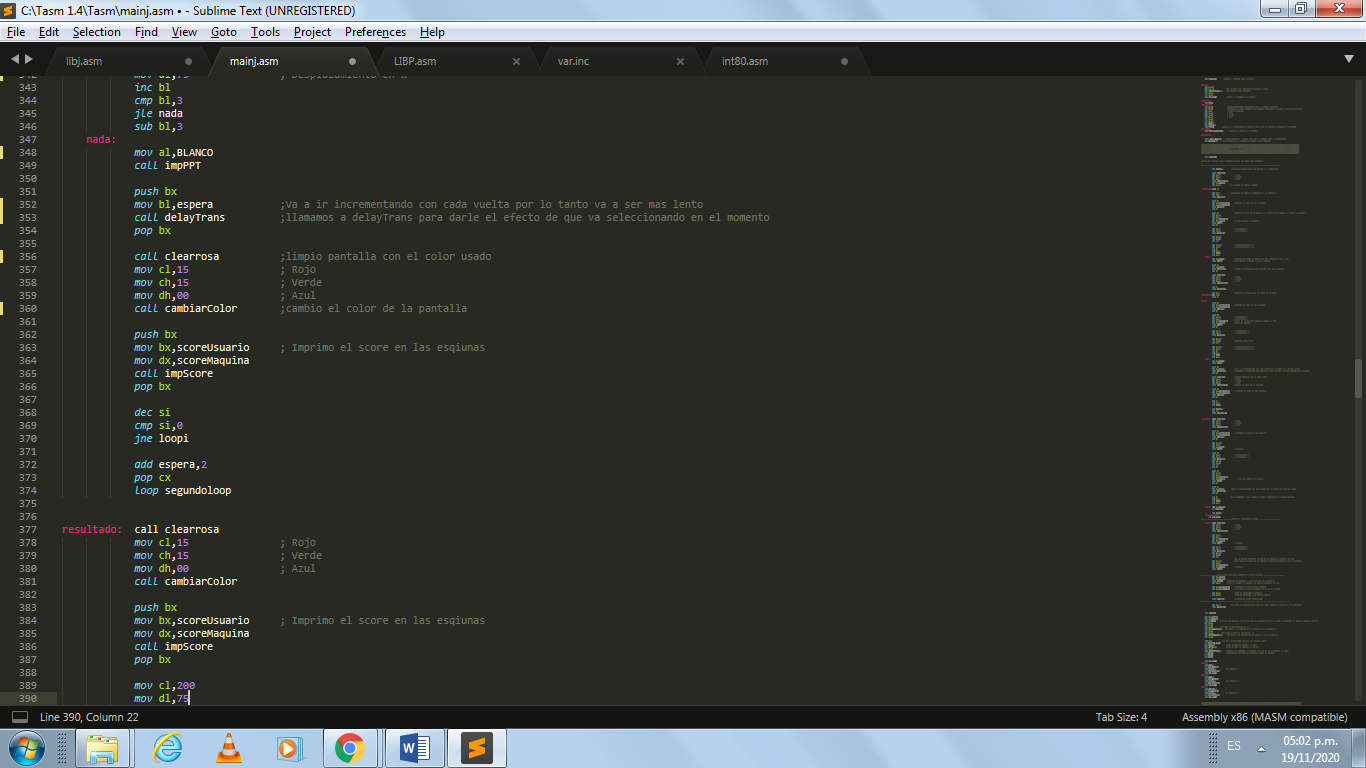


----------------------------------------------------------------------------------------------------------------------------------------------------







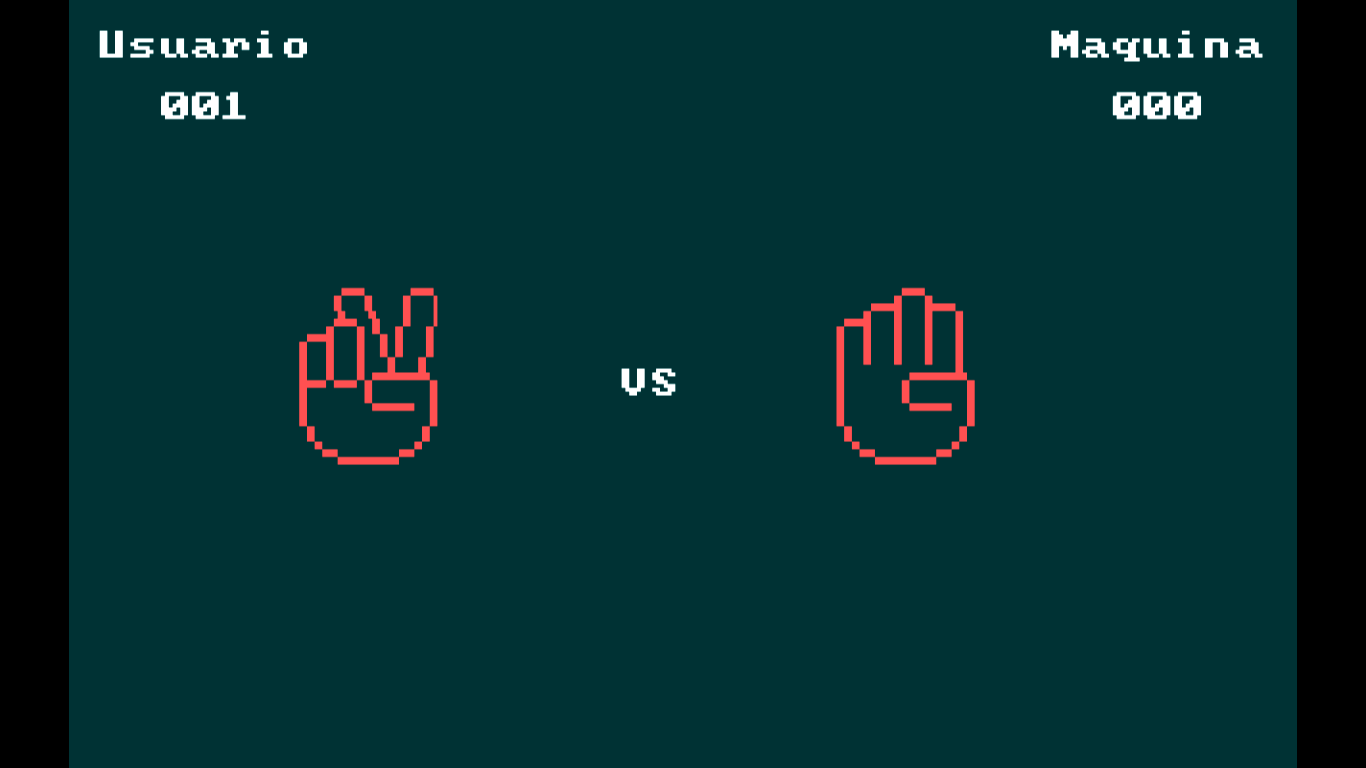






**Pantalla cinco:**

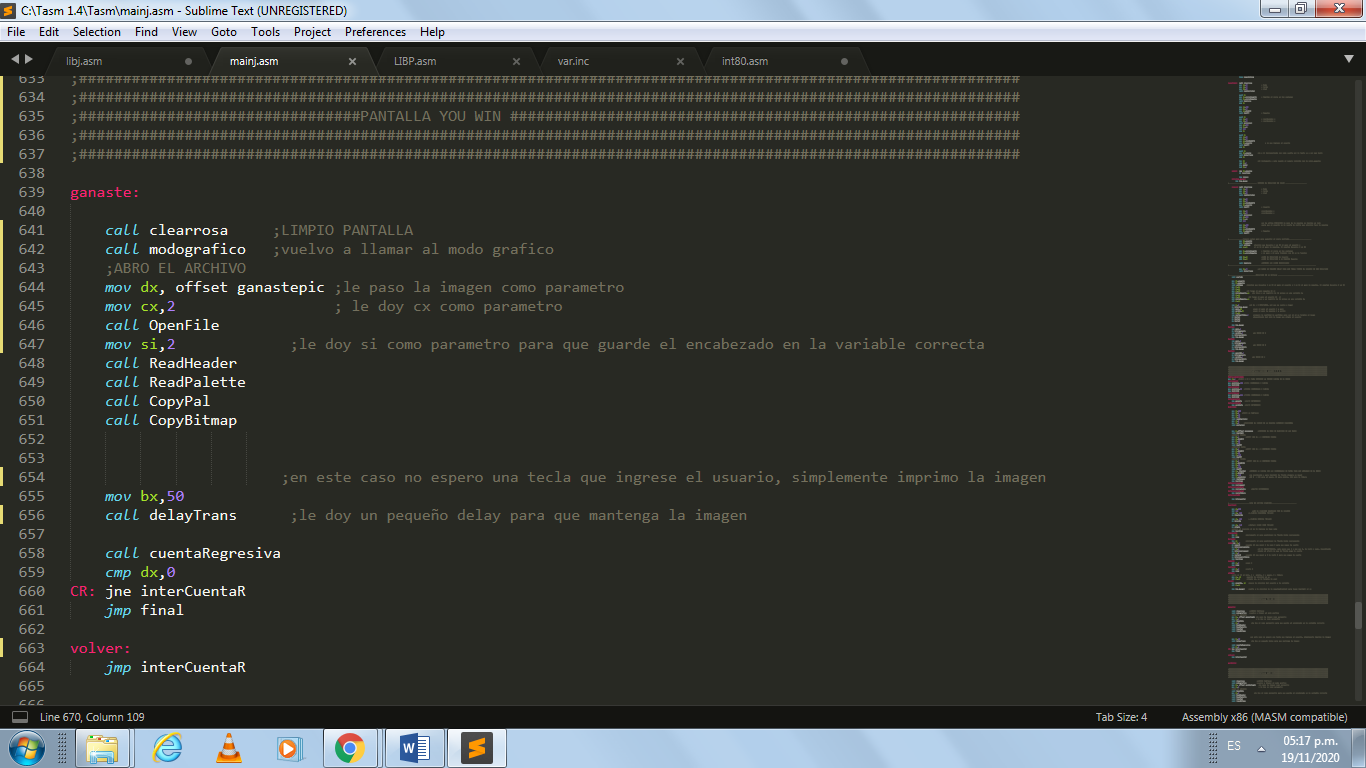
Antes de comenzar la pantalla de versus, se llama a la función “selec\_maquina” que devuelve (1,2,3) de forma aleatoria para enfrentarse con la elección del usuario. Luego se ingresa en la pantalla de versus, lo que se hace aquí es imprimir la mano del usuario, en color rojo, para luego ir imprimiendo piedra, papel, tijera dentro de un loop y con un delay el cual va incrementándose a medida que el loop va progresando, lo cual genera un efecto de lentitud en la impresión de las manos hasta llegar a la elección random de la máquina. Al llegar al resultado random seleccionado previamente se imprime en rojo demostrándole al usuario si perdió o gano de forma visual.



-----------------------------------------------------------------------------------------------------------------------------------------



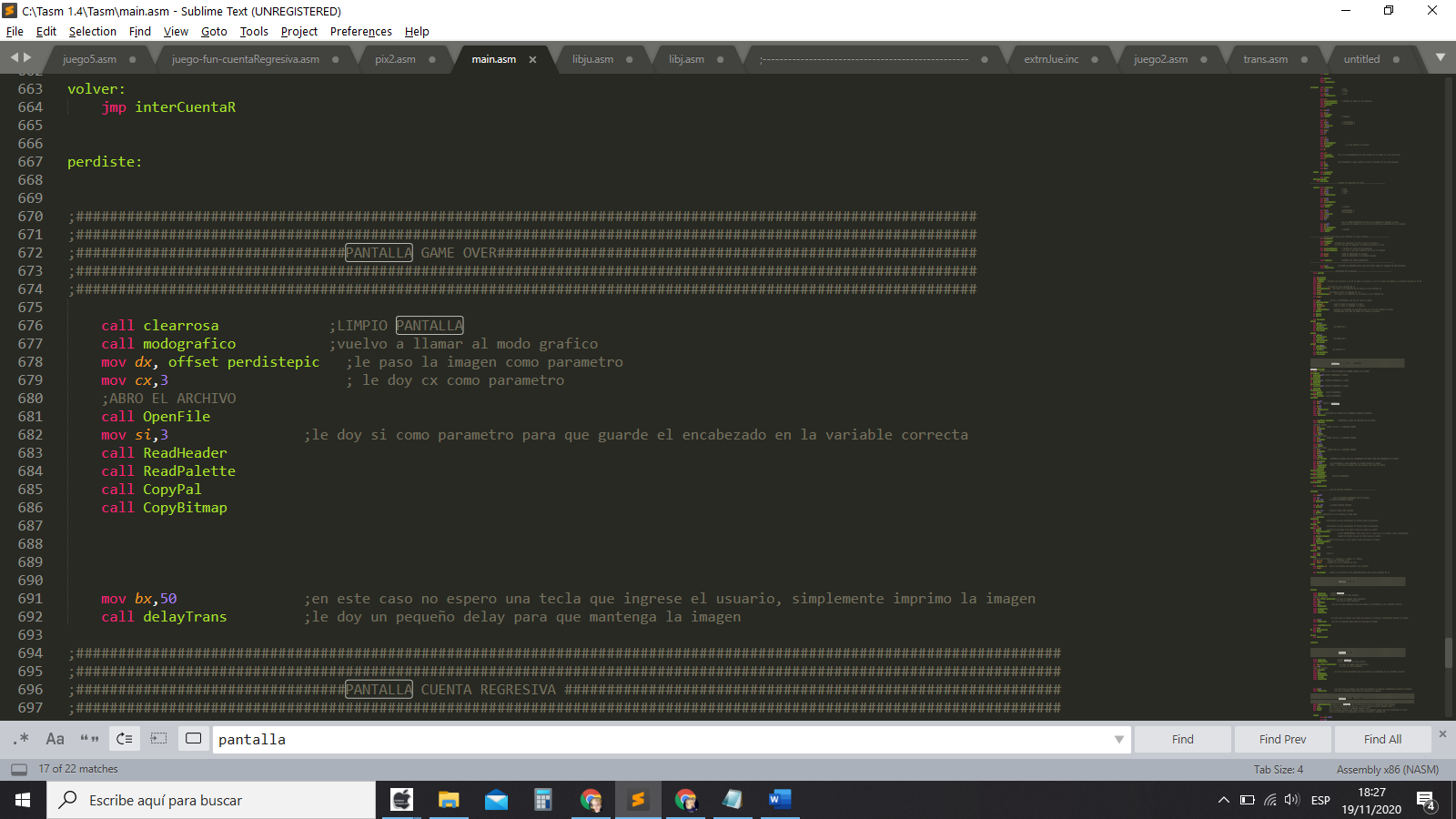
**Resultado batalla:** Esta parte del código no es una pantalla en sí, más bien representa la definición de la partida haciendo uso de la función “UsvsMaq” para definir quien gano, si la función devuelve dh = 1 gano el usuario, dl = 1 gano la máquina y dx = 0 hubo empate. Este registro se les suman a las variables “gano” y “perdió” simultáneamente para luego corroborar si ya termino la partida total dependiendo del modo de juego, de igual manera se le adiciona al score para poder imprimir en pantalla el resultado final de la partida.



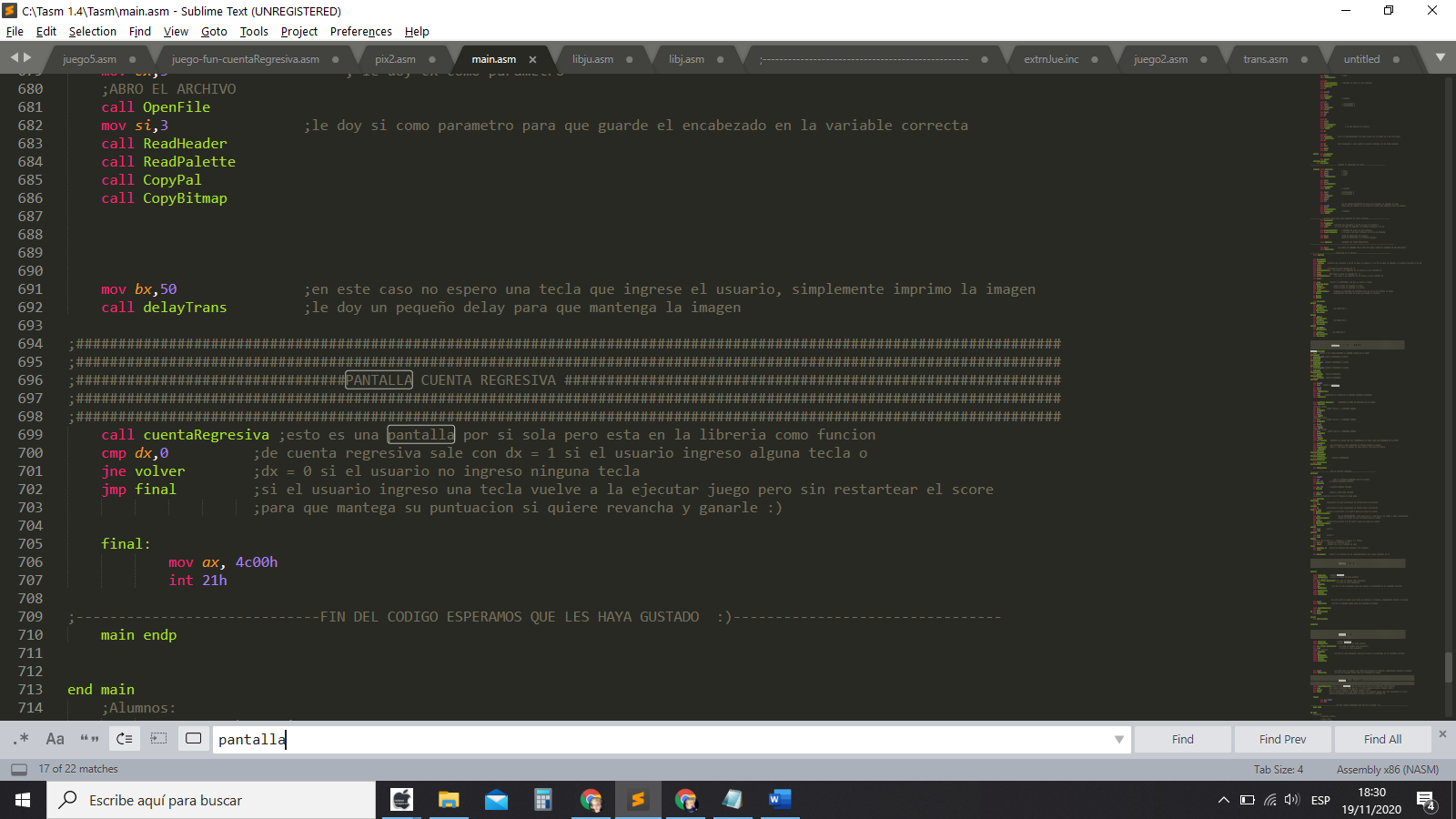
**Pantalla seis:** En el caso de que el usuario gane, se imprime una imagen con las palabras “you win”, usando funciones para abrir el archivo con formato bmp.



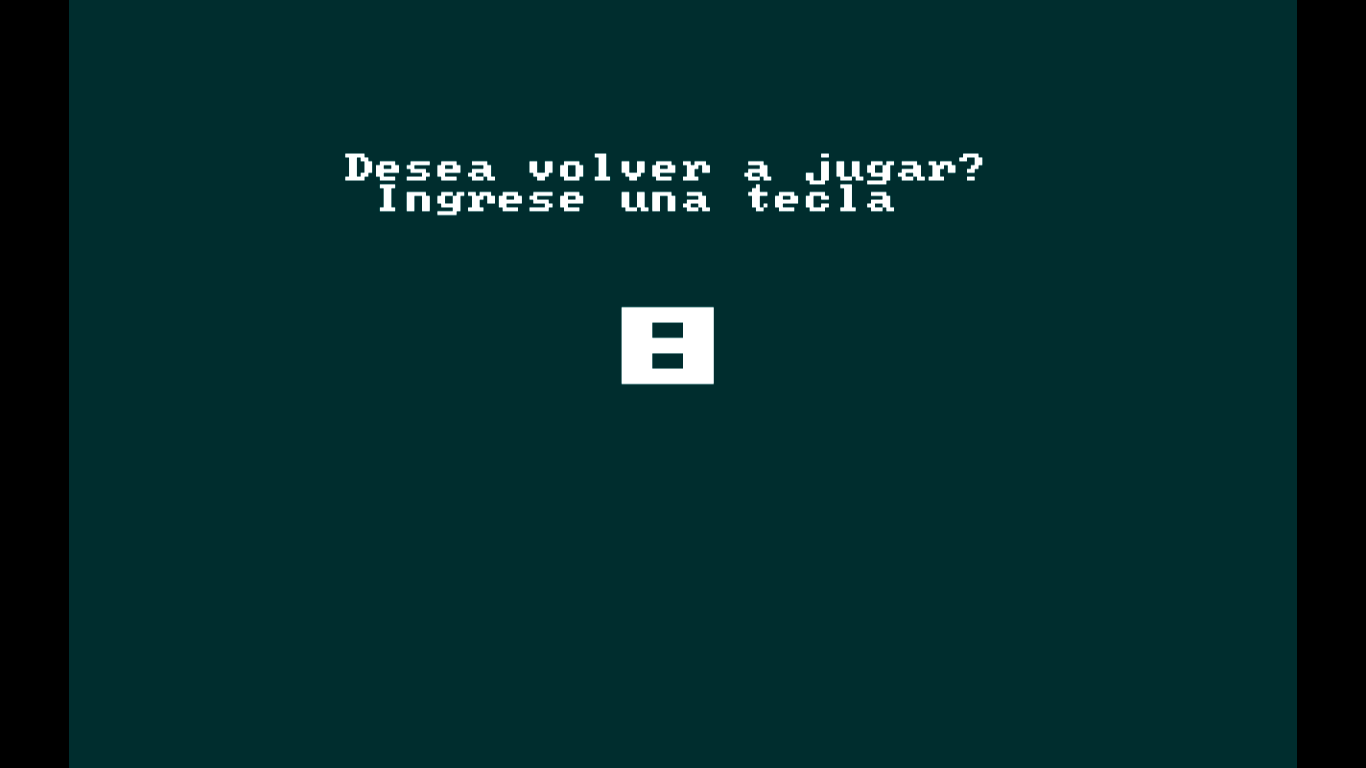




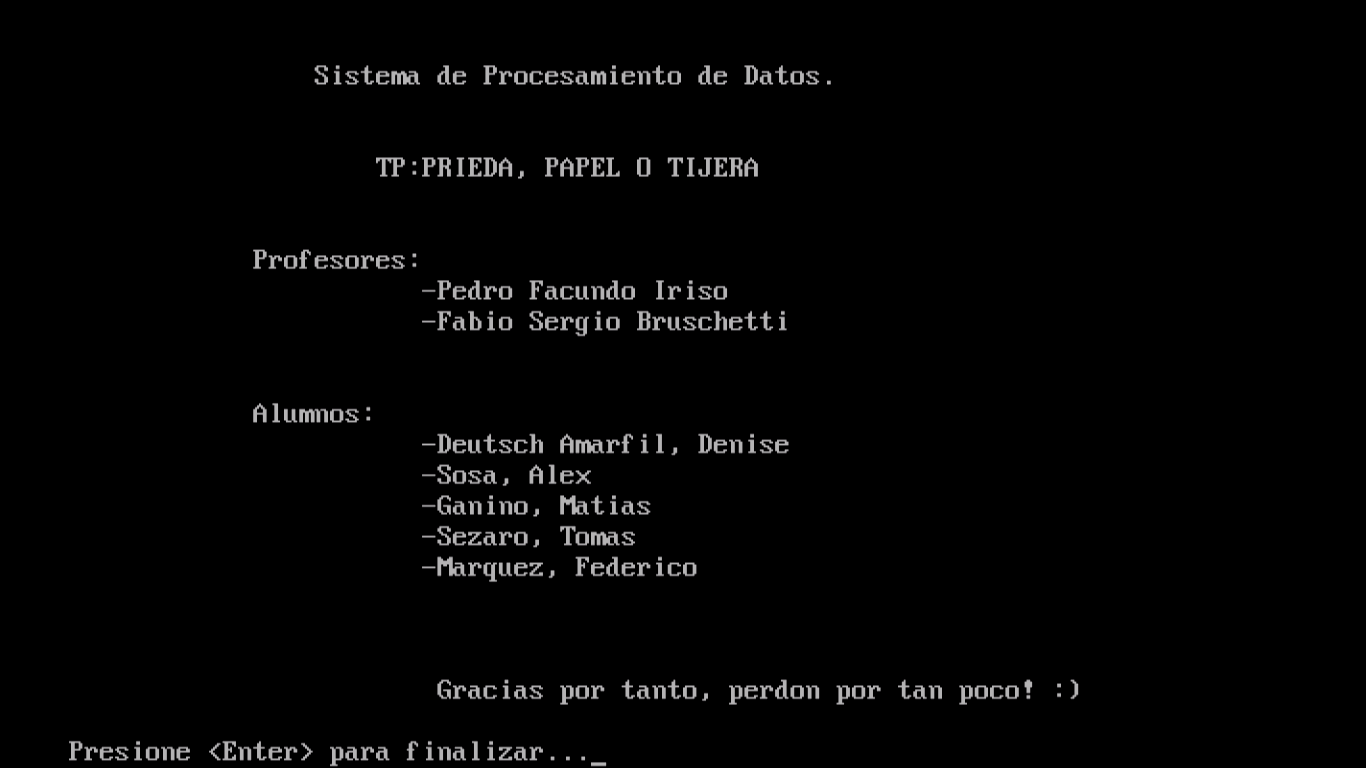
**Pantalla siete:** En el caso de que el usuario pierda, se imprime una imagen con las palabras “Game Over”, usando funciones para abrir el archivo con formato bmp.



**Pantalla ocho:** Al finalizar la partida, se muestra en pantalla una cuenta regresiva de diez segundos. El fondo se oscurece a medida que pasa el tiempo. Si se ingresa una tecla antes de que la cuenta llegue a 0, el juego vuelve a ejecutarse manteniendo el score actual. Si no, se sale del juego.



Pantalla nueve: Si se deja la cuenta regresiva llegar a 0, imprime creditos de un archivo txt



### 

### Instrucciones de juego

* Ingrese una tecla cualquiera para salir de la pantalla de presentación e iniciar el juego.
* Seleccionar la cantidad de partidas a jugar en el menú de opciones.
* elegir piedra, papel o tijera desplazándose con las flechas del teclado y confirmar la opción elegida oprimiendo enter.
* En el caso de que el resultado sea empate o se haya seleccionado jugar múltiples partidas, se juega de nuevo.
* Después de que se haya emitido el resultado final, si se desea volver a jugar, ingresar una tecla antes de que termine la cuenta regresiva.

### 

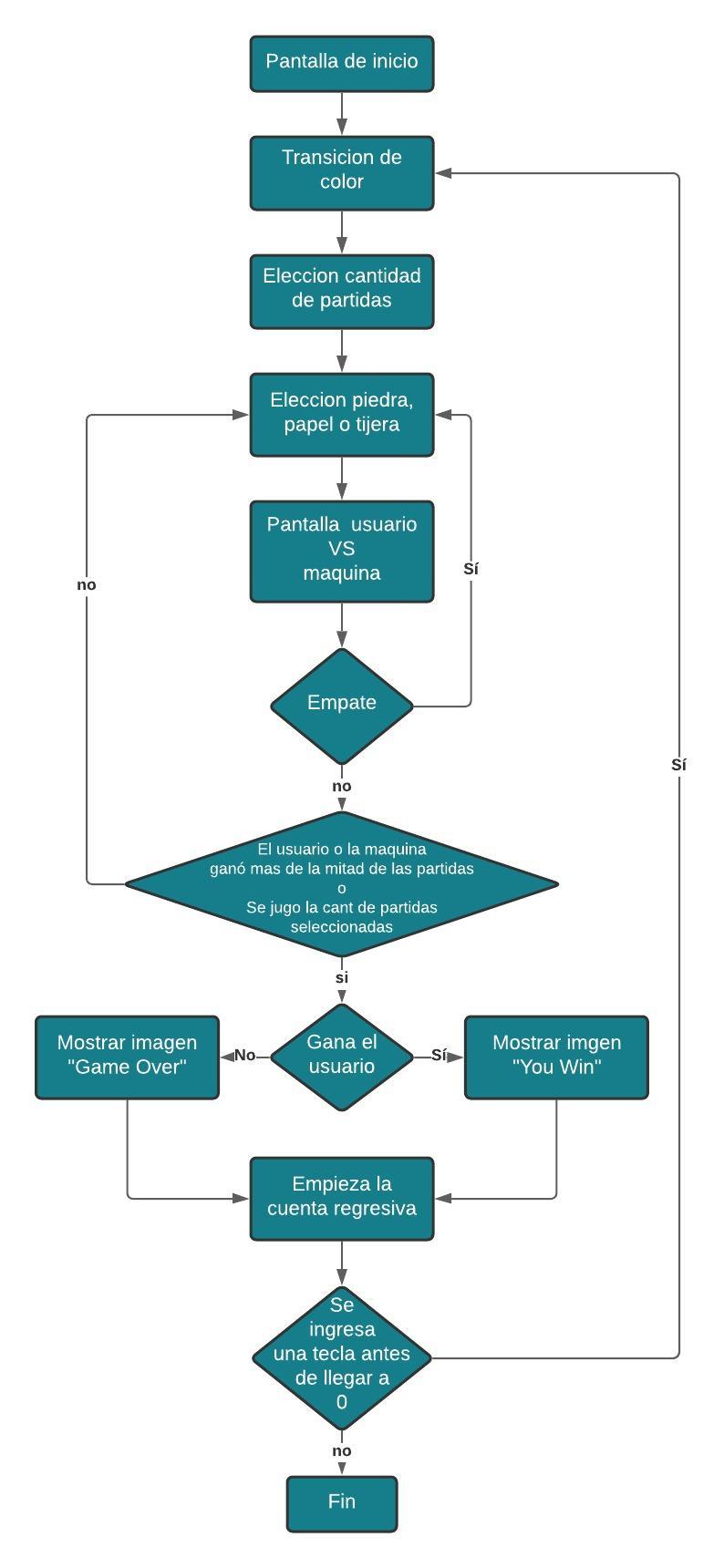
### Compilación, Instalación y Ejecución

Descomprimir el PiedraPapelTijera.rar.

Ejecutar el “juego.bat” teniendo los siguientes archivos en la carpeta TASM:

* ppt.bmp
* nw.bmp
* youWin.bmp
* gameOver.bmp
* var.inc
* lib.asm
* main.asm
* creditos.txt

**Diagrama de Flujo:**



### 

### Funciones implementadas

Desarrollo de cada una de las funciones indicando cómo reciben los datos y cómo los envía

1. **Cuenta Regresiva**

Función: Muestra una cuenta regresiva en pantalla de 9 a 0 y a medida que avanza la pantalla se oscurece

Recibe: Las siguientes variables tienen que estar declaradas en la librería

* texto Cuenta db " Desea volver a jugar?",0dh,0ah,24h
* textoCuenta2 db "Ingrese una tecla",0dh,0ah,24h

Devuelve: DX: 1 si se desea jugar otra vez, 0 si no

1. **delayTrans:**

Funcion: Espera una cantidad muy corta de tiepo

Recibe: BL: Cant de ticks que se quiere esperar

Devuelve: Nada

1. **usVsMaq**

Funcion: Devuelve el resultado de la ronda

Recibe: BH: Eleccion del usuario  
 BL: “Eleccion” maquina

Devuelve: DH: Un 1 si ganó el usuario  
 DL: Un 1 si ganó la maquina  
 DX: 0 si fue empate

1. **impScore:**

Función: Muestra el score en la pantalla

Recibe: BX: El score del usuario

DX: El score de la maquina

En la librería tiene que estar declarada la variable

score db "000",24h

textoUsuario db "Usuario",0dh,0ah,24h

textoMaquina db "Computadora",0dh,0ah,24h

Tiene que estar activado el modo grafico

Devuelve: Nada, solo imprime el score en la pantalla

1. **selec\_maquina**:

FUNCION: Devuelve un numero aleatorio entre 1,2 o 3

Recibe: NADA

Devuelve: AH = número que "eligió" la maquina

1. **leerScanCode**:

Función: Lee el scand code ingresado por teclado

Recibe: Nada

Devuelve: AH el valor del scan code leido

7. **ImPPT**:

Funcion: Imprime piedra, papel o tijera segun parametro

Recibe: BL: 1 (Piedra), 2 (Papel) o 3 (Tijera)

AL: Color

DL: Desplazamiento en y

DI; Desplazamiento en x

Devuelve: Nada, imprime la mano que le enviaron según los parametros indicados

8. **clearRosa**:

FUNCION: Limpia la pantalla en el color que este, si se quiere limpiar menos pantalla, primero

posicionar el cursor y luego llamarla.

Recibe: Nada

Devuelve: Nada, solo limpia la pantalla.

9. **CopyPal:**

Función: Copiar la paleta de colores a la memoria de video

El número del primer color debe enviarse al puerto 3C8h

La paleta se envía al puerto 3C9h

Recibe: nada

Devuelve: nada

10. **ReadPalette:**

Función: Lee la paleta de colores, usando la variable palete del archivo var .inc

Recibe: nada

Devuelve: nada

11 **ReadHeader:**

Función: leer encabezado del archivo .bmp

Recibe: SI (1,2,3) para ver en que variable guardar el encabezado, esta para poder usar 3 imágenes

Devuelve: Nada

12. **OpenFile:**

Función: Funciones de BMP para imprimir la foto de bienvenida

Recibe: CX (1,2,3) dependiendo del archivo que se quiera abrir, esta para abrir 3 archivos.

Devuelve: Nada

13. **impImagen:**

Función: Imprime una imagen pixel por pixel pasándole las coordenadas como una serie variables que representan las filas

las cuales están compuestas por los números de columnas a pintar

Ej.: fila0 db 1,2,3,4,24h

fila1 db 1,4,24h

Recibe: BX: Offset de la primera variable de las coordenadas

AL: Color.

DL: Desplazamiento en y.

CL; Desplazamiento en x.

Devuelve: Nada, imprime pixeles en pantalla siguiendo las coordenadas de las variables.

14. **cambiarColor**:

Función: Busca un color en la pantalla y lo cambia

Recibe: BX: Color a cambiar

DH: Rojo 0-60

CH: Verde 0-60

CL: Azul 0-60

Devuelve: Nada

15. **modoGrafico**:

Función: Pasa a modo Grafico 320x200 pixeles ,39x24 columnas imprimibles Recibe: Nada

Devuelve: Nada, solamente cambia el modo de pantalla

16. **posCursor:**

Función: posiciona el cursor en la parte de nuestra ventana que queramos, recordar que el cursor queda posicionado, por lo cual hay que volver a posicionarlo luego

Recibe: CL fila en la cual queremos trabajar (de 0 a 24 en decimal son las filas)

CH columna en la cual queremos trabajar (de 0 a 39 decimal son las columnas)

cl = 0 y ch = 0 son punta superior izquierda de la pantalla

Devuelve: Nada, simplemente posiciona el cursor para poder imprimir con la interrupción 21h

17. **Clearscreen**:

Función: limpia pantalla y vuelve al modo texto

Recibe: nada

Devuelve: nada, simplemente limpia la pantalla y vuelve al modo texto original.

18. **vaciarBuffer:**

Funcion: Vacía el Buffer sin importar la cantidad de teclas apretadas.

Recibe: Nada

Devuelve: Nada, el buffer vacío