



UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
FACULTAD DE INGENIERÍA
ESCUELA DE CIENCIAS

MANUAL DE USUARIO

Nombre: Angel Francisco Sique Santos

Carnet:

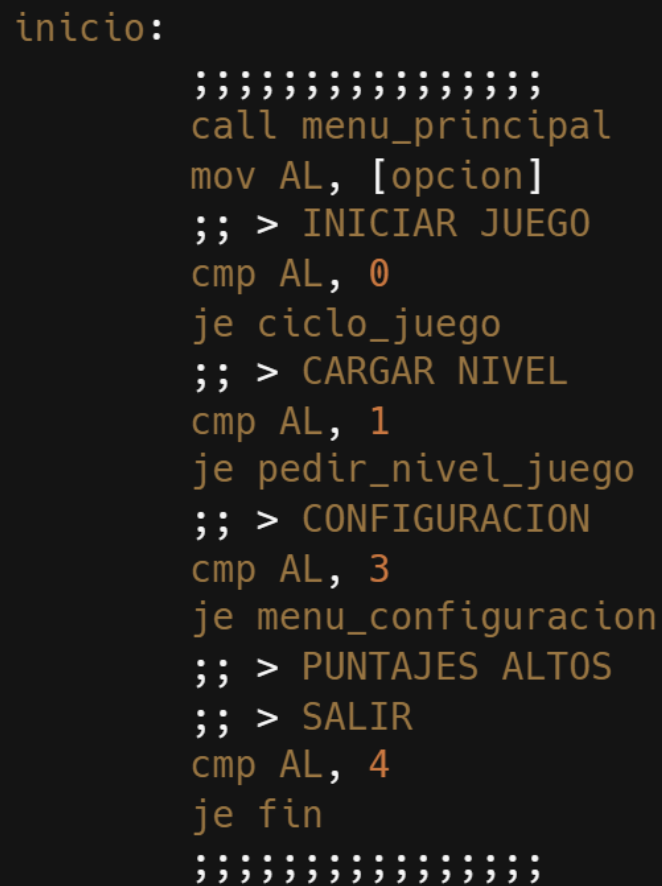
2	0	2	0	–	1	2	0	3	9
---	---	---	---	---	---	---	---	---	---

Uso del programa

1. Lo primero en mostrarse al abrir la pantalla es un mensaje de bienvenida, con los datos del alumno.

```
mensaje_inicial:
    ;; MODO VIDEO ;;
    mov AH, 00
    mov AL, 13
    int 10
    ;;;;;;;;;;;;;;
    mov DL, 0c
    mov DH, 09
    mov BH, 00
    mov AH, 02
    int 10
    ;; <-- posicionar el cursor
    push DX
    mov DX, offset mensaje_bienvenida
    mov AH, 09
    int 21
    pop DX
    ;;
    ;;;; DATOS
    add DH, 02
    mov BH, 00
    mov AH, 02
    int 10
    push DX
    mov DX, offset mensaje_datos_desarrollador
    mov AH, 09
    int 21
    pop DX
    ;; Delay de 5 segundos
    call delay
    ;;;;;;;;;;;;;;
    je inicio
```

2. Luego se muestra el menú principal con las opciones principales. En este menú se puede mover con las flechas del teclado y para seleccionar una opción se debe presionar la tecla F1.



```
inicio:
    ;;;;;;;;;;;;;;;;;;
    call menu_principal
    mov AL, [opcion]
    ;; > INICIAR JUEGO
    cmp AL, 0
    je ciclo_juego
    ;; > CARGAR NIVEL
    cmp AL, 1
    je pedir_nivel_juego
    ;; > CONFIGURACION
    cmp AL, 3
    je menu_configuracion
    ;; > PUNTAJES ALTOS
    ;; > SALIR
    cmp AL, 4
    je fin
    ;;;;;;;;;;;;;;;;;;
```

```

menu_principal:
    call clear_pantalla
    mov AL, 0
    mov [opcion], AL    ;; reinicio de la variable de salida
    mov AL, 5
    mov [maximo], AL
    mov AX, 50
    mov BX, 28
    mov [xFlecha], AX
    mov [yFlecha], BX
    ;; IMPRIMIR OPCIONES ;;
    ;;;; INICIAR JUEGO
    mov DL, 0c
    mov DH, 05
    mov BH, 00
    mov AH, 02
    int 10
    ;; <<-- posicionar el cursor
    push DX
    mov DX, offset iniciar_juego
    mov AH, 09
    int 21
    pop DX
    ;;
    ;;;; CARGAR NIVEL
    add DH, 02
    mov BH, 00
    mov AH, 02
    int 10
    push DX
    mov DX, offset cargar_nivel
    mov AH, 09
    int 21
    pop DX
    ;;
    ;;;; PUNTAJES ALTOS
    add DH, 02
    mov BH, 00
    mov AH, 02
    int 10
    push DX
    mov DX, offset puntajes
    mov AH, 09
    int 21
    pop DX
    ;;
    ;;;; CONFIGURACION
    add DH, 02
    mov BH, 00
    mov AH, 02
    int 10
    push DX
    mov DX, offset configuracion
    mov AH, 09
    int 21
    pop DX
    ;;
    ;;;; SALIR
    add DH, 02
    mov BH, 00
    mov AH, 02
    int 10
    push DX
    mov DX, offset salir
    mov AH, 09
    int 21
    pop DX
    ;;;;
    call pintar_flecha
    ;;;;
    ;; LEER TECLA
    ;;;;

```

3. La opción de cargar nivel pide al usuario que escriba el nombre del mapa que desea cargar.

```
pedir_nivel_juego:

    call clear_pantalla
    mov AL, 0
    mov DL, 08
    mov DH, 05
    mov BH, 00
    mov AH, 02
    int 10
    ;;
    mov DX, offset mensaje_cargar
    mov AH, 09
    int 21
    mov DX, offset nueva_lin
    mov AH, 09
    int 21
    mov DX, offset nueva_lin
    mov AH, 09
    int 21

    mov DL, 08
    mov DH, 08
    mov BH, 00
    mov AH, 02
    int 10

    mov DX, offset mensaje_indicar
    mov AH, 09
    int 21
    ;; PEDIR NOMBRE
pedir_de_nuevo_nombre:
    mov DL, 08
    mov DH, 08
    add DH, 03
    mov BH, 00
    mov AH, 02
    int 10
    ;;
    mov DX, offset prompt_nombre
    mov AH, 09
    int 21
    mov DX, offset buffer_entrada
    mov AH, 0a
    int 21
    mov DI, offset buffer_entrada
    inc DI
    mov AL, [DI]
    cmp AL, 00
    je pedir_de_nuevo_nombre
    cmp AL, 20
    jb aceptar_tam_nom
    mov DX, offset nueva_lin
    mov AH, 09
    int 21
    jmp pedir_de_nuevo_nombre
```

4. La función "pintar_mapa" se encarga de pintar un mapa. Utiliza bucles anidados para iterar a través de cada fila y columna del mapa. Dentro de los bucles anidados, llama a otra subrutina llamada "obtener_de_mapa" para obtener el valor en la posición actual en el mapa. En función del valor obtenido, realiza distintas acciones para pintar el elemento correspondiente en el mapa, como espacio vacío, jugador, pared, caja, objetivo o suelo.

```
pintar_mapa:
    mov AL, 1    ;; fila
    ;;
ciclo_v:
    cmp AL, 18
    je fin_pintar_mapa
    mov AH, 00    ;; columna
    ;;
ciclo_h:
    cmp AH, 28
    je continuar_v
    push AX
    call obtener_de_mapa
    pop AX
    ;;
    cmp DL, NADA
    je pintar_vacio_mapa
    ;;
    cmp DL, JUGADOR
    je pintar_jugador_mapa
    ;;
    cmp DL, PARED
    je pintar_pared_mapa
    ;;
    cmp DL, CAJA
    je pintar_caja_mapa
    ;;
    cmp DL, OBJETIVO
    je pintar_objetivo_mapa
    ;;
    cmp DL, SUELO
    je pintar_suelo_mapa
    ;;
    jmp continuar_h
```

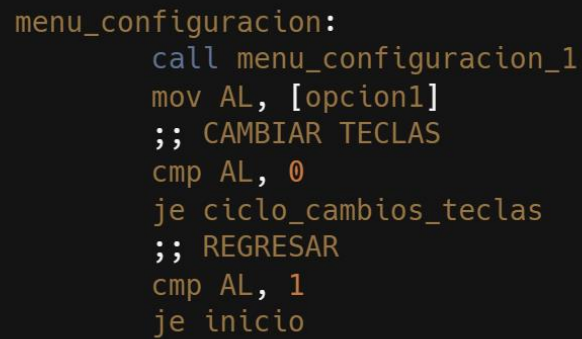
5. Este código llama a la subrutina "entrada_juego" que maneja la entrada del usuario durante un juego.

```
entrada_juego:
    mov AH, 01
    int 16
    jz fin_entrada_juego ;; nada en el buffer de entrada
    mov AH, 00
    int 16
    ;; AH <- scan code
    cmp AH, [control_arriba]
    je mover_jugador_arr
    cmp AH, [control_abajo]
    je mover_jugador_aba
    cmp AH, [control_izquierda]
    je mover_jugador_izq
    cmp AH, [control_derecha]
    je mover_jugador_der
    cmp AH, 3c
    je menu_pausa
    ret
```

6. El código define un menú llamado "menu_pausa". Llama a una función llamada "pintar_menu_pausa" para mostrar el menú. Luego verifica el valor de una variable llamada "opcion" y realiza diferentes acciones en función de su valor. Si "opcion" es 0, continúa con el programa. Si "opcion" es 1, salta a una etiqueta llamada "inicio" para abandonar el programa.

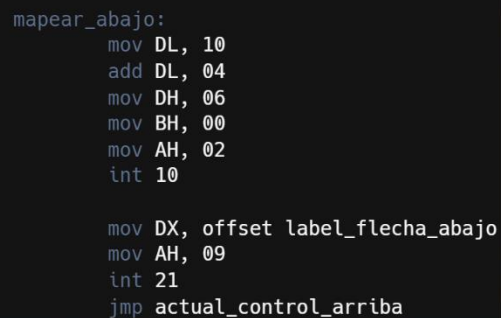
```
menu_pausa:
    call pintar_menu_pausa
    mov AL, [opcion]
    cmp AL, 0
    ;; Continuar
    cmp AL, 1
    ;; Abandonar
    je inicio
```

7. El código es parte de una configuración de menú. Primero llama a un submenú llamado "menu_configuracion_1". Luego, comprueba el valor de la variable "opcion1". Si es 0 entra en un bucle llamado "ciclo_cambios_teclas" para cambiar las claves. Si es 1, vuelve al menú principal llamado "inicio".



```
menu_configuracion:
    call menu_configuracion_1
    mov AL, [opcion1]
    ;; CAMBIAR TECLAS
    cmp AL, 0
    je ciclo_cambios_teclas
    ;; REGRESAR
    cmp AL, 1
    je inicio
```

8. En este código se definen cuatro funciones diferentes: mapear_abajo, mapear_arriba, mapear_derecha y mapear_izquierda. Cada función establece los valores de los registros DL y DH en valores específicos, luego llama a la interrupción 10h para mover el cursor a una posición específica en la pantalla. Después de mover el cursor, la función muestra una etiqueta específica usando la interrupción 21h. Finalmente, la función salta a una parte diferente del código para continuar con la ejecución.



```
mapear_abajo:
    mov DL, 10
    add DL, 04
    mov DH, 06
    mov BH, 00
    mov AH, 02
    int 10

    mov DX, offset label_flecha_abajo
    mov AH, 09
    int 21
    jmp actual_control_arriba
```


mapear_arriba:

```
    mov DL, 10
    add DL, 04
    mov DH, 08
    mov BH, 00
    mov AH, 02
    int 10
    ;; Si es la flecha hacia arriba
    mov DX, offset label_flecha_arriba
    mov AH, 09
    int 21
    jmp actual_control_derecha
```

mapear_derecha:

```
    mov DL, 10
    add DL, 04
    mov DH, 08
    add DH, 02
    mov BH, 00
    mov AH, 02
    int 10

    mov DX, offset label_flecha_derecha
    mov AH, 09
    int 21
    jmp actual_control_izquierda
```

mapear_izquierda:

```
    mov DX, offset label_flecha_izquierda
    mov AH, 09
    int 21
    jmp seleccionar_opcion_configuracion
```

9. La última opción es la de salir, esta opción cierra el juego.



```
;; > SALIR  
cmp AL, 4  
je fin
```



```
fin:  
.EXIT  
END
```