## Manual Técnico

Para la realización de la práctica 4 de Arquitectura de Computadores y Ensambladores 1 se hizo uso de lenguaje Ensamblador en DOSBox para simular. Se solicito realizar el juego GO en este.

Para la realización de este y el ordenamiento del código se hizo uso de 2 archivos. Uno llamado main.asm y otro llamado macros.asm.

En el main se coloco el código principal del juego, el flujo de la aplicación en sí. Para este se siguió lo siguiente:

Segmento de Data:

```
; DATA SEGMENT
; ON THIS SEGMENT WE CREATE THE "VARIABLES" THAT WE ARE GOING TO USE IN THE PROGRAM
.data

; SPECIAL CHARACTERS···
; END SPECIAL CHARACTER

; TESTING···
; END TESTING

; HEADER AND MENU···
; END HEADER AND MENU

; HTML···
; END HTML

; TURN (15 POSITIONS)···
; END OF TURN

; POSITION (VARIABLES FOR THE POSITION WHERE WE ARE GOING TO PUT A COIN)···
; END POSITION

; COMMAND···
; ROUTES···
; END ROUTES

; ERRORS···
; END ERRORS

; TABLE···
; END OF TABLE

; MATRIX. 56h -> Empty. 55h -> Neutral Territory | 57h -> White. 58h -> White Territory | 42h -> Black. 43h -> Black Territoy···
; END OF MATRIX
```

Este segmento se dividió en secciones para poder llevar un control y orden mejor de los datos a utilizar. Entre estas secciones están:

- Caracteres especiales: Donde se tienen caracteres como salto de línea, cadena para limpiar, error de comando invalido, errores del juego en sí, carácter para remover moneda y contador de turnos.
- Testeo: Este solo contiene la cadena ”Prueba”. Esta se utilizó para probar poco a poco el código que se iba implementando si hubiera algún error lógico.
- Encabezado y Menú. Este contiene el encabezado, el menú, el mensaje de ingreso de ruta y el mensaje de finalización del juego.
- HTML: Esta sección contiene todas las cadenas necesarias para realizar el reporte HTML del estado del tablero.

- Turno: Este contiene cadenas que indican quien está jugando. Además, contiene la cadena para la moneda de cada jugador. También uno que nos va a servir más adelante para verificar quien es el que está jugando.
- Posición: Esta contiene el valor de la fila y la columna donde se quiere ingresar algo. Además, dos auxiliares que se usan para la lectura del archivo.
- Comando: Esta sección solo tiene la cadena que guardará el comando que ingresa el usuario.
- Rutas: Esta contiene la cadena donde el usuario guardará una ruta para guardar una partida. Además, tiene los handler que se usarán para el html y el archivo de entrada.
- Errores: Esta sección contiene mensajes de error por si ocurre alguno mientras se manejan los archivos.
- Tabla: Contiene la tabla que se imprime en la consola del DOSBox.
- Matriz: Esta contiene 9 arreglos y uno que se llama fileContent. Estos describen donde están hay monedas o esta vació.

Luego del segmento de data se encuentra el segmento de código que igualmente fue separado por “secciones” dependiendo de que acciones se realicen en esas secciones.

```
; CODE SEGMENT
; ON THIS SEGMENT WE START TO WRITE THE CODE
.code

main proc

    ; BEGGINIG OF THE PROGRAM
    Start: ...
    ; PRINCIPAL MENU
    PrincipalMenu: ...
    Game:          ...
    ; Probado
    LoadGame: ...
    Playing:                  ...
    PutBlackCoin:          ...
    PutWhiteCoin:          ...
    INVALIDCOMMAND: ...
    SHOWGAME: ...
    SAVEGAME: ...
    PASSTURN: ...
    EndGame: ...
    EXITGAME: ...
    Exit: ...
    ; Errors ...
main endp

end
```

En donde:

- Start: Contiene únicamente un salto al menú principal.
- Menú Principal: Este contiene la parte donde se le solicitara al usuario que desea hacer. Si ingresar a una partida normal, cargar partida o salir.
- Juego: En este se imprimirá el tablero y se mandará a la sección de Jugando
- Cargar juego: En esta se le pedirá al usuario que ingrese una ruta para guardar un juego. Se dibuja el tablero y se analiza lo que esta en el archivo para imprimir las monedas en sus respectivas posiciones. Al finalizar se manda a la sección jugando.
- Jugando: Se mueve el cursor a la posición donde recibirá el comando. Se lee el comando y se analiza para ver que ingresa.

```
; EXIT: ROW = 4fh COLUMN = 4fh
; PASS: ROW = 54 COLUMN = 54
; SAVE: ROW = 45 COLUMN = 45
; SHOW: ROW = 53 COLUMN = 53
; ERROR: ROW = 43 COLUMN = 43
```

Se tiene valores de fila y columna especiales para ver que comando ingreso. Si no ingreso alguno de los comandos establecidos se le manda a un error con 43 en la fila. Sino cumple con ninguno de los valores anteriores entonces se mueve a la posición donde se encuentra la nueva moneda y se pasa a la sección de PonerColorFicha.

- Poner ficha negra: Pinta la ficha y le pasa el turno a las blancas.
- Poner ficha blanca: Pinta la ficha y le pasa el turno a las negras.
- Comando invalido: Muestra el mensaje de comando invalido.
- Mostrar juego: Genera el HTML.
- Guardar juego: Pide al usuario una ruta y crea un archivo con el estado de la partida actual
- Pasar turno: Pasa de turno, si el contador de pasar turno llega a 2 termina el juego.
- Fin de juego: Genera el HTML final y termina el juego.
- Salir del juego: Sale de la partida actual
- Salir: Termina la ejecución de la aplicación.
- Errores: Este contiene la muestra de errores para archivos y cuando se juega

Para el archivo de los macros se tienen macros que hacen validaciones, generan archivos, y realizan situaciones específicas. De los más importantes están:

- Print: Imprime algo que se encuentre en la sección de data del main.
- GetText: Consigue el texto que escriba el usuario hasta que escriba un salto de línea.
- GetCommand: Analiza carácter por carácter la entrada para ver que comando realizará o que posición quiere poner el usuario.
- PutCoinMacro: Verifica que en la posición donde desean ingresar no haya una ya colocada. Si no hay una ficha, imprime la ficha y la coloca en el arreglo de la fila correspondiente.
- MoveCursor: Macro para mover el cursor a alguna posición de la pantalla.
- ConcatenateRows: Concatena las filas en un solo arreglo para la creación del archivo de guardado.
- AnalizeText: Analiza un juego guardado para imprimirlo en pantalla.
- GenerateHTML: Concatena todo lo que requiere el archivo HTML a generar.
- getDateAndHour: Macro para conseguir la fecha. Esta se concatena en una cadena.
- NumberToString – getNumber: Realiza una división entre 10 para poder convertir lo que devuelven las interrupciones de fecha y hora a su código ASCII.
- Pushear y Popear: Macros para recuperar lo que se encuentre en los registros.

Nota: No muestro screenshots de este porque son casi 2000 lineas de código :’v.