**Instituto Tecnológico y de Estudios Superiores de Occidente**

**PROGRAMACIÓN CON MEMORIA DINÁMICA**



**EVALUACIÓN I - BATALLA NAVAL**

Presentan

**Enoch Joaquín Álvarez Goñi 739917**

**Jorge Ramón Figueroa Maya 739924**

Profesor: Francisco Cervantes

Fecha: <21/09/2022>

Contenido

[Introducción 3](#_Toc114486552)

[Análisis del problema 3](#_Toc114486553)

[Propuesta de solución 3](#_Toc114486554)

[Pruebas 3](#_Toc114486555)

[Conclusiones 3](#_Toc114486556)

[Fuentes de información 3](#_Toc114486557)

# Introducción

En este proyecto fue realizado utilizando el lenguaje de programación C y empleando el uso de matrices, direcciones de memoria y apuntadores, consta de un juego de tipo “battleship” o “batalla naval”. Este proyecto tiene como objetivo el asemejarse lo más posible al juego físico de batalla naval en cuestión de modo de juego usuario contra maquina y el cumplimiento los requerimientos especificados por el profesor en torno al criterio de evaluación proporcionado por el mismo.

El uso y funcionamiento de estas estructuras (matrices) será explicado de manera detallada mas adelante en este documento al igual que el de sus respectivas funciones y estructuras.

# Análisis del problema

El problema que se dio a resolver consta en crear un juego “Battleship” o “batalla naval” en lenguaje C, este juego consiste en dos tableros (matrices de tamaño determinado por el jugador) los cuales contienen distintos barcos de distintos tamaños, estos son: portaviones, buque, submarino, crucero y lancha. El 30% de las celdas totales de los tableros deben de estar llenas con barcos, es decir, si se tiene una matriz de 10x10 30 celdas deben de contener barcos. Entre otras especificaciones del problema se pidió una serie de estructuras que contienen diferentes funcionalidades del juego las cuales son CELDA y NAVE, además de distintas funciones básicas del programa como lo es el mostrar el menú, modificar las dimensiones del tablero, seleccionar el modo de juego, iniciarlo o salir.

Cada estructura tiene sus especificaciones de funcionamiento, un ejemplo de esto es la estructura CELDA, esta define el estado de la celda que se encarga de indicar si está ocupado o vacío, le da un identificador único a cada tipo de nave y se encarga de definir si una nave ha sido impactada o no. Además, se pidió la creación de funciones como lo es la función TABLERO la cual funciona con la matriz de tipo CELDA y tiene como finalidad asignar las naves de manera aleatoria con base los criterios de espacio y variedad de naves anteriormente mencionados con un margen de error definido por el profesor (+5%, -5%).

# Propuesta de solución

La primera solución que se implemento fue una función llamada “mostrarMenu” como cumplimiento de uno de los primeros criterios



La funcionalidad principal de esta función es imprimir el menú de opciones e inicialización del juego especificando las dimensiones de tablero y dificultad preestablecidas.

Texto

Descripción generada automáticamente

Después se implementaron las distintas estructuras pedidas por el profesor las cuales son:

Estructura tipo CELDA

Texto

Descripción generada automáticamente

Esta estructura tiene como finalidad almacenar los siguientes datos: el estado de las celdas (estadoCelda) que determina si una celda en el tablero de juego esta ocupada por un barco o no, el identificador (idBarco) el cual se encarga de almacenar un valor único que identifica a un tipo de barco e impacto (impactoBarco) el cual se encarga de almacenar un valor entero que determina si la nave fue impactada (1) o no (0).

Estructura tipo NAVE

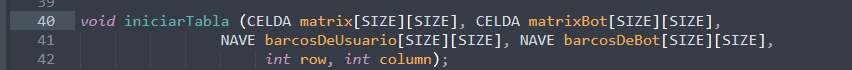
Texto

Descripción generada automáticamente

Esta estructura tiene como finalidad almacenar los siguientes datos: el tipo de nave (tipoBarco) que almacena el identificador generado de manera aleatoria único de una nave para después asociarla al tipo de nave al que pertenece (es decir si el identificador generado fue 5 el tipo de barco lo almacenará como “p” ya que pertenece al portaviones), orientación (orientacionBarco) el cual se encarga de almacenar si una nave esta orientada de manera horizontal (“H”) o vertical (“V”), hundida (sunk) se encarga de almacenar un valor verdadero o falso el cual define si una nave ha sido completamente hundida o no.

El siguiente paso fue implementar las distintas funciones necesarias para el funcionamiento del programa, las cuales son:

Void iniciarTabla:



Esta función se encarga de inicializar las matrices (tableros) del usuario y de la computadora con los datos necesarios para posteriormente hacer la asignación de barcos.

Su funcionamiento es el siguiente:

Texto

Descripción generada automáticamente

Tanto la matriz del usuario como de la maquina funcionan de la misma manera, primero se establecen las variables necesarias para los ciclos for y luego se navega en los datos deseados dentro de las estructuras (\*celda y \*nave) a manera de apuntador, para posterior mente inicializar un ciclo for el cual comienza a crear la matriz (por filas y columnas) y llenarla con los datos mencionados anteriormente mencionados (ver estructuras) y sus valores preestablecidos.

Void iniciarJuego

Texto

Descripción generada automáticamente

Esta función se encarga principalmente de inicializar el juego como su nombre lo sugiere.

Su funcionamiento es el siguiente:

Texto

Descripción generada automáticamente

Se encarga de mostrar los datos necesarios y de invocar las funciones necesarias para el funcionamiento básico del juego, funciona inicializando un switch case el cual contiene tres respuestas posibles basadas en el nivel de dificultad elegido.

El primer caso se trata de la dificultad didáctica el cual funciona por medio de un ciclo do while primero se limpia el sistema para subsecuentemente invocar las funciones necesarias para imprimir los tableros tanto del usuario como de la computadora, se piden las coordenadas a atacar para subsecuentemente guardarlas en la función correspondiente y hacer los “bombardeos” del usuario a la máquina.

El segundo ciclo do se encarga de hacer los “bombardeos” de la maquina hacia el usuario.

El segundo caso se trata de la dificultad fácil, hace lo mismo que el primer caso exceptuando la impresión del tablero de la máquina

Y un default el cual se encarga de, en el caso que el usuario seleccione una dificultad del fuera del rango le pida a este que ingrese una opción valida

Void hacerBarco:



Se encarga de crear el tipo de barco y asignar su celda de manera aleatoria

Su funcionamiento es el siguiente:

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Primera mente se utiliza la librería rand para poder crear los barcos de manera aleatoria y de la misma manera asignar su lugar en el tablero de ambos jugadores, además de definir la orientación de los barcos por medio de un switch case, para subsecuentemente entrar en un ciclo do-while y repetir las mismas funciones para la máquina.

Void printUser:



Se encarga de imprimir la matriz del usuario en el juego principal.

Su funcionamiento es:

Texto

Descripción generada automáticamente

Primero se acceden a las estructuras necesarias para subsecuentemente utilizar ciclos for para imprimir el tablero por filas y columnas por medio de los apuntadores de las estructuras necesarias.

Void printBot:



Se encarga de imprimir el tablero de la maquina en el juego principal

Su funcionamiento es:

Texto

Descripción generada automáticamente

Funciona de la misma manera que la función de usuario, pero imprime los datos para el tablero de la maquina por medio del mismo tipo de ciclos for anteriormente mostrados.

Void bombardeoAUsuario:



Se encarga de definir la forma en la que se registran los golpes o las fallas por parte de la maquina

Su funcionamiento es:

Texto

Descripción generada automáticamente

Se generan de manera aleatoria dos números que funcionan como coordenadas a atacar, luego se acceden a los datos necesarios dentro de las estructuras por medio de apuntadores para subsecuente mente utilizar un ciclo if para reconocer si la maquina acertó o fallo.

Void bombardeoABot:



Se encarga de definir el ataque del usuario a la maquina.

Su funcionamiento es:

Texto

Descripción generada automáticamente

Funciona case de la misma manera que la función designada para la máquina, la única diferencia es que en lugar de generar de manera aleatoria las coordenadas a atacar el usuario las ingresa y se leen, para subsecuentemente definir si acertó o falló.

Main:

Su funcionamiento es: Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Funciona primordialmente con un ciclo do-while el cual se encarga de invocar las funciones anteriormente mencionadas para el correcto funcionamiento del juego y utiliza un switch-case para poder leer las elecciones de tamaño de tablero deo usuario y dificultad.

# Pruebas

*Se espera;*

* *La descripción un conjunto de pruebas que permita validar que su programa funciona de forma correcta acorde a lo descrito en la actividad.*

*Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente*

Al momento de iniciar el programa, nos muestra el menú principal, el cual, podremos ajustar el tamaño de la matriz (Cada opción tiene su función).

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza media

Dependiendo de la dificultad en la que se quiera jugar el juego, mostrará las matrices, ya sea del usuario y la de la pc, o solo la del usuario, con un total de 30% de las cacillas utilizadas para los barcos

Forma, Flecha

Descripción generada automáticamente

Como se puede observar, también se genera una tabla para la matriz de la máquina.

Texto

Descripción generada automáticamente

Muestra el turno del jugador o de la computadora (Los turnos son aleatorios)

Forma, Flecha

Descripción generada automáticamente

Si se dispara a un barco, en la tabla del oponente (Si es que se está mostrando), cambia de valor a una “X”, en cambio, si no se le dispara a un barco, se cambia el valor a un “+”.

Imagen que contiene Flecha

Descripción generada automáticamente

Cuando es el turno de la computadora, escoge coordenadas al azar, con las mismas condiciones que en la tabla del oponente, si da a un barco, cambia el valor a un “X”, en cambio, si no da a una nave, solo se cambia a un “+”.

Y el juego termina cuando el usuario o el oponente dispara en todos los barcos.*.*

# Conclusiones

Enoch Joaquín Álvarez Goñi: este proyecto honestamente estuvo bastante complicado, ya que por mi parte fue una introducción completamente nueva al funcionamiento especifico del lenguaje C

Jorge Ramón Figueroa Maya: Para ser sinceros, este proyecto me costó mucho trabajo, ya que, apenas comencé a recordar la syntaxis del lenguaje, pero a fin y al cabo, se pudo lograr un gran avance para el proyecto. Cuando comencé con el proyecto, me di cuenta de que prácticamente no había aprendido de los apuntadores, sin embargo, con el paso del tiempo, repetí 2 veces el proyecto, y aprendí a utilizar los apuntadores de la manera intencionada.

# Fuentes de información

* *Este proyecto lo desarrollamos solos, tratamos de no utilizar tutoriales, ejemplos de proyectos, entre otro tipo de apoyo, ya sea audiovisual o de texto, como códigos y ejercicios, todos los problemas los solucionamos en archivos externos con nombres como “prueba1.c”, los cuales, en nuestro repositorio de GitHub está repleto de estos archivos.*
* *Sin embargo, con el poco avance que teníamos a través de los días, pudimos concluir con el proyecto de manera efectiva y concisa.*