|  |
| --- |
| **Paradigmas de programación**  **Informe** |
|  |
| **CRISTIAN EDUARDO ESPINOZA SILVA** |

|  |  |
| --- | --- |
|  | Profesor: |
|  | MIGUEL TRUFFA MONTENEGRO |
|  |  |
|  |  |
|  |  |

|  |  |  |
| --- | --- | --- |
|  | Santiago - Chile |  |
|  | 2016 |  |

Tabla de Contenidos

Tabla de contenido

[CAPÍTULO 1. Introducción 1](#_Toc462437419)

[1.2 Descripción del problemA 1](#_Toc462437420)

[1.3 obejtivos del proyecto 1](#_Toc462437422)

[1.4 aSPECTOS DE IMPLEMENTACION 2](#_Toc462437423)

[1.5 DESCRIPCINO DE LA SOLUCIÓN 2](#_Toc462437424)

[CAPÍTULO 2. Desarrollo de la aplicación 2](#_Toc462437425)

[2.1 Proyecto y paradigma 2](#_Toc462437428)

[2.2 programa o aplicación 2](#_Toc462437429)

[2.2.1 Organización de la aplicación 2](#_Toc462437430)

[La aplicación está distribuida en 3 archivos, los cuales son los siguientes: 2](#_Toc462437431)

[2.2.2 Estructuras implementadas en la aplicación 3](#_Toc462437432)

[2.2.4 Funciones 4](#_Toc462437433)

[CAPÍTULO 3. Conclusiones 9](#_Toc462437434)

[CAPÍTULO 4. REFERENCIAS 9](#_Toc462437435)

[CAPÍTULO 5. ANEXO 10](#_Toc462437436)

[- iMAGENES DE ESTRUCTURAS: 10](#_Toc462437437)

[10](#_Toc462437438)

# Introducción

Durante el pasar los años, los lenguajes de programación han ido cambiando, de ser lenguajes de bajo nivel han ascendido a lenguajes de programación de muy alto nivel, estos grandes cambios que se han producido durante este tiempo pasan por el tema de querer resolver problemas desde diferentes puntos de vistas, con el fin de querer facilitar la resolución de este problema. Todo lo anterior conlleva a un cambio de pensamiento al momento de querer enfrentar un problema y este nuevo pensamiento lo podemos llamar paradigmas. Uno de los principales objetivos que se llegara a cabo es poder implementar el proyecto en el paradigma imperativo.

# Descripción del problemA

El problema consiste en diseñar el juego llamado “BATALLA NAVAL”, el cual tiene una temática de derribar barcos enemigos. Se autogenera un tablero de NxM dimensiones que son comúnmente fijas. Cada jugador debe posicionar sus barcos en su mitad del tablero.

Su principal objetivo es derribar el barco enemigo que se logra atacando con sus barcos en una posición “x” en la mitad del tablero enemigo, el primer usuario que acabe con la totalidad de los barcos del enemigo es el ganador del juego.

# Resultado de imagen para batalla naval

Figura 1: Temática del juego Batalla naval

# obejtivos del proyecto

Los objetivos son realizar un programa que represente el juego batalla naval desarrollado en el lenguaje de programación c, además de ocupar el paradigma de programación imperativo.

Los objetivos del proyecto son implementar ciertas cantidades de funciones las cuales recaen en las siguiente:

* Realizar un tablero de n x m dimensiones, además de tener la opción de poder cargar un tablero y guardar tablero ya creado anteriormente
* Poder imprimir un tablero ya generado, para que el usuario pueda verlo, además de ver y pensar en donde posicionará sus embarcaciones.
* Poder verificar si el tablero cargado o creado es válido para poder comenzar una partida en él.
* Poder posicionar los barcos del oponente en su mitad del tablero y colocar en la otra mitad los barcos que yo deseo ocupar en alguna posición ingresada.
* Poder realizar ataques de barco a barco dentro de la aplicación, dejando marca de cuando se realiza el disparo, además de dar aviso si el disparo fue exitoso.

# aSPECTOS DE IMPLEMENTACION

Para abordar cada uno de los problemas encontrados al momento de implementar la solución al problema se ocupó: búsqueda de espacio, similitud débil, entre otros.

Estos conceptos mencionados se llevaron a hechos mediante el compilador Mingw y el editor de texto sublime text 3.0

El principal alcance que se deseó alcanzar en el proyecto era cumplir con todas las funcionalidades pedidas en el enunciado entregado.

# DESCRIPCINO DE LA SOLUCIÓN

La solución que le daremos al problema tiene que ser pensada mediante el paradigma de programación emperativo e implementada en el lenguaje de programación c. El cual los resultados de cada función que serán declaradas vendrán siendo mostrados por la “cmd”

Se hizo una abstracción de la siguiente manera: El usuario dispondrá de un menú en el cual se encontrarán 8 acciones que puede realizar dentro del juego, que estas cumplirán con los objetivos expuestos anteriormente.

# Desarrollo de la aplicación



## Proyecto y paradigma

**2.1.1 El paradigma:**

El proycto que se llevó acabo fue implementado en un lenguaje de programación llamado “C”. En esos momentos es cuando entra el paradigma de programación que existen variedades, en este proyecto se verá implementado el paradigma de programación imperativo (o procedual). Principalmente, este paradigma significa mandar e ordenar y eso es exactamente lo que ocupamos al momento de programar.

## programa o aplicación

Ahora, se procederá a explicar el funcionamiento del programa, primero se explicarán las estructuras definidas dentro del programa, a continuación de lo anterior definiré el resto de las funciones.

### Organización de la aplicación

### La aplicación está distribuida en 3 archivos, los cuales son los siguientes:

Archivo 1 -> Contiene el menú principal de la aplicación y el llamado a las distintas funciones que son las encargadas de realizar las operaciones que el usuario solicita. Es un archivo de tipo .c

Archivo 2 -> Contiene definido las estructuras que se ocupan dentro del programa, además del “statusCode” que es el encargado de avisar ciertos tipos de errores, cabe mencionar que este archivo es de tipo .h

Archivo 3 -> Por ultimo tenemos el archivo donde tenemos declarada todo el algoritmo de cada una de las funciones que contiene el proyecto al igual que el archivo anterior es de tipo .h

### Estructuras implementadas en la aplicación

Como base, se ha indicado que las estructuras necesarias para el correcto funcionamiento del programa serían las siguientes:

* **Board:** Estructura que se encargada de guardar un tablero que será generado por dimensiones ingresadas por el usuario. Además de esa funcionalidad cumple con almacenar las filas, columnas y nivel del tablero, barcos del usuario. Una parte importante que cumple es almacenar la estructura “Dueno” que describiremos en el transcurso del informe.
* **Game:** Estructura encargada de guardar dos variables que son fundamentales para llevar acabo la batalla que se genera dentro del juego.
* **Params:** Esta estructura tendrá en su interior almacenada una de la estructura más importante del juego, llamada “Ship”.
* **Position:** Estructura encargada de recibir las posiciones que se van requiriendo dentro del funcionamiento del juego, ademas de cumplir un almacenamiento unico al momento de posicionar los barcos del usuario en el tablero, ya va indicando la cantidad de barcos que se tiene que posicionar en el tablero.
* **Ship:** Como anteriormente lo mencione esta es una de las estructuras que hace el manejo de información de una manera mucho mas factible dentro del juego, ya que es la encargada de almacenar todos los datos que contienen un barco dentro del juego.

Las siguientes funciones que se describirán están implementadas por la abstracción del problema hecha:

* **tm:** Estructura que facilita el manejo de tiempo dentro del juego, teniendo un rol fundamental al momento de guardar un tablero.
* **Dueno:** Estructura que se encarga de guardar la posición donde se coloca una embarcación, su vida y por último el nivel de esta embarcación.
  + 1. **Bibliotecas utilizadas:**

Las bibliotecas utilizadas en el programa fueron las siguientes:

#include <stdio.h> → Para poder utilizar funciones que puedan mostrar resultado al usuario por la consola, ej: (printr( ),scanf( ) ).

#include <time.h> →Consiste en poder obtener la hora y fecha exacta al momento que se esté realizando una acción. (Ocupada en saveBoard()).

#include <stdlib.h> → Contiene dos funcionalidades útiles que consisten en el ordenamiento y búsqueda de datos.

#include <string.h> → Permite el manejo de cadenas de caracteres infinitos, con el fin de facilitar los métodos de resolución ocupados.

Archivos incluidos en el programa:

- #include "nombre.c" → Donde se encuentra el desglose total de las funciones que se utilizan en el desarrollo del proyecto.

- #include "struct.h" → Donde se almacenan las estructuras implementadas para el desarrollo del proyecto

### Funciones

#### **Función main() :**

Es la función principal del programa, la que se encarga de llamar a cada una de las funciones como se le indique. Además, contiene un menú en cual le permite al usuario poder realizar las distintas funcionalidades implementadas en la aplicación.

#### **Funciones createBoard() , checkBoard() y print():**

En estas funciones se genera el terreno que se verá ocupado durante lo que dure la partida, por una parte, tenemos la función:

createBoard() → Es la encargada de generar el tablero con las dimensiones que el usuario estime conveniente e ingrese a la consola, además de ingresar el nivel de dificultad con el cual desea desarrollar el encuentro.   
Teniendo lo anterior, utilizamos la estructura Board con el fin de solicitar memoria para poder crear el tablero solicitado, luego se procede a guardar los datos ingresados en la estructura Board para ser utilizadas durante el transcurso del juego.   
Según la dificultad ingresada se auto genera la cantidad de barcos que tendrá la maquina en su territorio, además de ser guardados en la estructura para ser utilizados posteriormente.

checkBoard() → Es la encargada de ver la veracidad de un tablero generado cumpla con las condiciones para que se pueda desarrollar el juego sin problemas, una de las condiciones importantes que debe cumplir es que en una mitad del tablero se deben poder colocar la cantidad de barcos generados correspondiente al nivel. Al verificar lo anterior, se procede a llamar a la función posicionesCp(), que se encarga de colocar los barcos de la computadora en forma ramdon tanto su posición como el tipo de barco .

print() → La función print() revise como parámetro un Board, el cual es el tablero que se desea imprimir, accediendo a la estructura podemos obtener la cantidad de filas y columnas que contiene este tablero, para tener un mejor manejo al momento de declarar los dos ciclos para poder recorrer la matriz, así mostrando el tablero, teniendo en cuenta que se auto genera una separación para tener mejor visualización de su tablero y del oponente, además recibe como un parámetro showComplete, el cual indica si mostrar el tablero por completo al momento de ser 1 y en caso de ser 0 ocultara los barcos enemigos.

#### **Función putShip()**

putShip() -> Esta función está encargada de posicionar los barcos del usuario en una mitad del tablero, la cual revise como parámetro un Board. el cuál es el tablero que se está ocupando en el juego, además de recibir pos y ship. Estos dos parámetros mencionados anteriormente son el barco el cual será colocado en el tablero y en “pos” viene la posición principal del barco (que se obtiene accediendo a la estructura).

Luego se procede a realizar una validación en el tablero para ver la disponibilidad de casillas adyacentes de la posición ingresada dependiendo del largo del barco, este proceso está puesto dentro de un ciclo for (Se conoce el número de barcos a colocar, accediendo a la estructura Board), para colocar la totalidad de los barcos del usuario en la mitad de su tablero.

#### **Funciones saveBoard() y loadBoard():**

Estas funciones tiene cierta similitud entre ellas, ya que como su nombre lo indica se encargan de guardar o cargar algún tablero que haya sido generado y pasado por el filtro de la función checkBoard().   
  
Tomando en cuenta que en código para generar las estas dos funciones se ocupó un concepto visto llamado similitud débil, ya que tiene una pequeña modificación dentro de su algoritmo.  
  
 Estas funciones reciben un parámetro llamado “id”, el cual se encarga de auto generar el nombre con el cual será guardado o cargado el tablero.

#### **Funciones play():**

Una de las principales funciones y la que lleva acabo la dinámica del juego es la función play(), la cual es la encargada de realizar los ataques dentro del juego, esta función recibe 4 parámetros los cuales son los siguientes :   
  
Board → En donde viene el tablero que se generó en la partida , sea tanto por la función createBoard() o loadBoard(), permitirá ir almacenando cada ataque realizado por el usuario y la respuesta de la computadora, esto conlleva a que se va ir modificando el tablero a cada momento que se realice una acción..   
 ° Position → En donde viene las posiciones donde se van a realizar el ataque (fila y columna), esta se obtienen accediendo a su estructura.  
  
Ship → Es el barco con el cual se va a realizar el ataque al terreno enemigo, teniendo en cuenta que estos tienen un límite de ataques el cual depende del nivel que sea el barco.

El funcionamiento de la función consta de realizar el ataque al oponente y de inmediato se obtiene la respuesta de la computadora, teniendo en consideración que el barco que ataca como la posición donde se realizara este se obtienen de manera random. La manera de ataque que se implementa, se basa en verificar el nivel del barco que fue seleccionado para realizar el ataque, luego se pasa a validar la posición donde se generó el ataque (teniendo en cuentas que estos datos se obtienen a través de la estructura “Position”, que contiene la fila y columna donde se realiza el ataque), que puede tener 3 respuestas las cuales son:

* Estar ocupado por “mar” → En este caso solamente se cambia el valor que anteriormente tenía el tablero la dicha posición.
* Estar ocupada por “barco” → Este es el caso exitoso, en el cual se procede a tener nuevamente 2 eventos que pueden ser activados:
  + Quitar vida→ En este caso se procede a descontar un punto de vida al barco alcanzado por el disparo, accediendo a la estructura “Ship” y disminuyendo su vida en una cantidad.
  + Eliminar barco → Este evento ocurre al momento de que su vida llegue al valor de 0, de inmediato se elimina del juego y se procede a descontar en una cantidad los barcos del oponentes que sobreviven.  
    Luego, como anteriormente se mencionó viene la respuesta de la computadora, como el problema que se presenta tiene el mismo objetivo que el anterior, se procede a realizar un procedimiento con cierta similitud.

#### **Diseño de la aplicación:**

La aplicación tiene un diseño relativamente didáctico con el usuario, para poder crear un tablero se procede a solicitar las dimensiones de este, que corresponden a las filas y columnas que tendrá al momento de representarlo en el juego, además de eso se pide la dificultad con la cual va a realizar la partida.

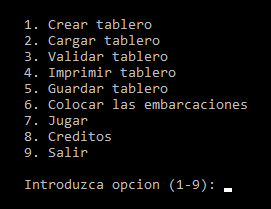
Luego se procede a validar el tablero, con el fin de verificar si las filas e columnas son un numero positivo y columnas es un numero par y por último se verifica que el largo de los barcos que se van a colocar en el barco sea menor que la cantidad de celdas que contiene una mitad del tablero.

Al terminar los procesos anteriores, se procede a seleccionar los barcos que va a ocupar en esta partida, los cuales pueden ser universidaddechile(ude), universidadcatolica(udc), unionespañola(ues), colocolo(cc), deportesiquique(dp), deportestemuco(dp), serena(s), valdivia(v).

Teniendo en cuenta que las longitudes de sus abreviaciones representan la cantidad de vidas que contiene cada uno de estos barcos, el usuario debe ingresar la cantidad de barcos que va a utilizar en esta partida pudiendo ser +/-1 la cantidad de los barcos que tenga el oponente puesto en su tablero. Luego tiene que proceder a escoger los barcos de los cuales, si o si 1 tiene que escoger un barco de nivel 3, que son los barcos que contienen 3 cifras, y teniendo en cuenta que solo puede tener uno en juego, los barcos que le queden por posicionar pueden ser del nivel que el estime conveniente.

Terminando el proceso anterior tenemos que procedemos a comenzar el ataque que se basa primero en escoger el barco con el cual desea realizar el disparo, para luego ingresar la posición de donde realizara el ataque y de inmediato la computadora devuelve su ataque, este proceso se repetirá hasta que uno de los dos jugadores quede sin embarcaciones en su terreno, pero tenemos que tener en cuenta que no se pueden repetir los disparos a la misma posición que anteriormente se atacó. Al momento que uno de los oponentes quede sin embarcaciones en su terreno el juego se acaba siendo victorioso el jugador que tenga mayor cantidad de barcos en su terreno.

* + 1. **Resultados:**



Los resultados que se obtienen en la aplicación se pueden ver reflejados en el menú principal que se le muestra al usuario:

El usuario tiene la libertad de escoger cualquiera de las siguientes opciones, teniendo en cuenta que no puede validar tablero, colocar las embarcaciones, imprimir tablero, jugar, si no tiene creado o cargado un tablero**.** Para controlar el error se declara una variable Board = NULL, con el fin de saber en el momento que tenemos un tablero.

Figura 2: Menú principal

Opción 1 -> Esta función permite crear un tablero de NxM dimensiones, además de solicitar la dificultad con la cual se va a operar en esta partida.



Figura 3: Salida de la función createBoard()

Opción 2 -> Esta función reemplaza a la función de crear el tablero, ya que podemos cargar uno ya creado anteriormente y comenzar a jugar.

Opción 3 -> Esta función permite validar un tablero, si cumple con las condiciones para que sea un tablero validado.



Figura 5: Salida de la función checkBoard()

Opción 4 -> Función que muestra el tablero creado y permite ir actualizando el tablero cada ves que se realiza un ataque.

Opción 5 -> Función que permite guardar un tablero ya creado para que pueda ser reutilizado cuando sea conveniente.

Opción 6 -> Función que permite colocar los barcos al usuario.

Opción 7 -> Función que permite jugar a los usuarios

Opción 8 -> Créditos del proyecto

Opción 9 -> Permite salir de la aplicación.

# Conclusiones

- La resolución de este problema utilizando el paradigma imperativo, fue relativamente alcanzada se cumplieron con los requisitos obligatorios tanto como funcionales como no funcionales, se hizo un poco arduo el trabajo, ya que anteriormente no había tenido experiencias con este lenguaje de programación.

Una de las características que ayudo bastante a la realización del proyecto fue el llamado procedual que tiene este paradigma, ya que no se aleja de la realidad en la manera de ver la solución de ciertos problemas, cosa que en otros paradigmas las soluciones son un poco más abstractas.

Al tener la ventaja que el paradigma de programación tenía cierta cercanía con la realidad no trajo gran problema con la solución que se le daba a cada requerimiento que se estaba pidiendo, sin embargo, hay que tener en cuenta que para los siguientes laboratorios cuando tengamos que ocupar nuevos paradigmas tales como funcional o lógico la tarea se verá un poco más ardua pero no imposible. Se deja de lado el paradigma orientado a objetos por el motivo que ya se tuvo una experiencia programando en ese paradigma.

Una dificultad que se tuvo que ir enfrentar al transcurso de la solución, fue el gran manejo de punteros que se tuvo que aplicar, fue una complicación al momento de querer empezar a plasmar las ideas en código por el hecho que primera ves que se veía este tipo de paradigmas. Sin dejar de lado el manejo de estructuras que al momento de aprender usarlas como corresponden, conllevan a facilitar la solución del problema.

# REFERENCIAS

# ANEXO

# iMAGENES DE ESTRUCTURAS:

# 

