**INFORME PARCIAL II**

**Por: Natalia Polo Peña**

**Cristian Florez Grisales**

1. **Contextualización del problema:**

El desafío que debemos abordar consiste en la implementación del juego de mesa Othello en el lenguaje de programación C++, con un enfoque principal en la programación orientada a objetos (POO) para resolverlo. Para lograrlo, es esencial que dividamos el análisis del problema en varios pasos, el primero de ellos es estudiar y analizar las reglas y la lógica del juego, lo que nos permitirá realizar una implementación exitosa en C++.

1. **Análisis**

Ya analizadas las reglas del juego, podemos darnos cuenta de que este juego de mesa se compone principalmente de fichas negras y blancas, y de una cuadrícula en la realización de la implementación. Las fichas negras serán asteriscos (\*) y las fichas blancas guiones (-). El tablero es de 8x8, por lo tanto, tiene 64 casillas. Es importante destacar que, al principio del juego, las cuatro casillas del centro están ocupadas por dos filas de fichas blancas y dos filas de fichas negras. Cabe resaltar que las fichas son diferente color porque estamos hablando de dos jugadores diferentes un jugador para las negras y uno para las blancas.

Othello tiene ciertos movimientos reglas y condiciones para ganar las cuales resumiremos de manera breve:

Movimientos:

-El juego comienza con las cuatro fichas iniciales en el centro del tablero, dos de cada color.

-En los turnos, cada jugador debe colocar una ficha de su color en el tablero en una casilla vacía.

-Para poner una ficha, se requiere rodear al menos una ficha del oponente con las fichas propias en una línea horizontal, vertical o diagonal. Al hacerlo, las fichas del oponente atrapadas entre las fichas del jugador se voltean y cambian de color.

-Es obligatorio realizar al menos una captura en cada turno si existe una jugada válida disponible.

Ganar el Juego:

-La partida finaliza cuando no hay más movimientos disponibles para ninguno de los jugadores o cuando todas las casillas del tablero están ocupadas.

-El ganador se determina contando las fichas de cada color en el tablero al final del juego. El jugador con más fichas de su color es declarado vencedor.

Después de completar este paso crucial, que es de fundamental importancia para comprender el funcionamiento del juego, procederemos a analizar las metodologías y estrategias necesarias para llevar a cabo la implementación de este juego en el lenguaje C++. Este análisis nos permitirá abordar el desarrollo del programa con un enfoque informado y estratégico.

1. **Diseño de la solución:**

Clase Tablero:

* Atributos:
  + tablero - Una matriz de 8x8 para representar las fichas.
  + jugador\_turno - El jugador cuyo turno es actualmente.
* Métodos:
  + inicializar() - Inicializa el tablero con un nuevo juego.
  + colocar\_ficha() - Coloca una ficha en el tablero.
  + es\_movimiento\_valido() - Determina si un movimiento es válido.
  + revisar\_tablero() - Revisa el tablero para determinar si hay un ganador.
  + obtener\_jugador\_ganador() - Devuelve el jugador ganador, si lo hay.

Clase Jugador:

* Atributos:
  + nombre - El nombre del jugador.
  + color - El color de las fichas del jugador.
* Métodos:
  + \_\_init\_\_(nombre, color) - Constructor de la clase Jugador.

Clase Juego:

* Atributos:
  + tablero - Una referencia a la clase Tablero.
  + jugadores - Una lista de los dos jugadores.
  + turno - El turno actual del juego.
* Métodos:
  + iniciar() - Inicializa el juego.
  + turno\_siguiente() - Avanza al siguiente turno.
  + revisar\_resultado() - Determina el resultado del juego.
  + guardar\_historial() - Guarda el historial del juego.

Clase Ficha:

* Atributos:
  + color - El color de la ficha.
  + posicion - La posición de la ficha en el tablero.
* Métodos:
  + \_\_init\_\_(color, posicion) - Constructor de la clase Ficha.

Clase HistorialPartidas:

* Atributos:
  + partidas - Una lista de las partidas jugadas.
* Métodos:
  + agregar\_partida() - Agrega una partida al historial.
  + obtener\_partidas() - Devuelve una lista de las partidas jugadas.

Clase OthelloApp:

* Atributos:
  + tablero - Una referencia a la clase Tablero.
  + jugadores - Una lista de los dos jugadores.
* Métodos:
  + iniciar() - Inicializa el juego.
  + mostrar\_tablero() - Muestra el tablero en la consola.
  + obtener\_jugada\_jugador() - Obtiene la jugada del jugador actual.
  + jugar\_partida() - Juega una partida de Othello.

Atributos adicionales:

* Clase Tablero:
  + dimensiones - Las dimensiones del tablero (8x8 por defecto).
* Clase Ficha:
  + orientacion - La orientación de la ficha (horizontal o vertical).

Métodos adicionales:

* Clase Tablero:
  + obtener\_ficha\_en\_posicion() - Devuelve la ficha en la posición especificada.
  + obtener\_fichas\_en\_direccion() - Devuelve una lista de las fichas en una dirección especificada desde una posición especificada.
* Clase Jugador:
  + obtener\_fichas() - Devuelve una lista de las fichas del jugador.
* Clase Juego:
  + realizar\_jugada() - Realiza una jugada para el jugador actual.
  + obtener\_ficha\_en\_turno() - Devuelve la ficha del jugador en turno.
* Clase HistorialPartidas:
  + obtener\_partida\_por\_id() - Devuelve una partida con el ID especificado.