Universidad de Antioquia



Parcial 2

Jhoan Sneyder Portillo Anaya

Informática II

UdeA – Facultad de Ingeniería

Tabla de contenido

[A. Contextualización breve del problema. 3](#_Toc150201223)

[B. Análisis: consideraciones para el abordaje del problema y desarrollo de propuesta de la estrategia de solución 3](#_Toc150201224)

[Consideraciones para el abordaje del problema 3](#_Toc150201225)

[Diseño de clases 3](#_Toc150201226)

[Representación del tablero 3](#_Toc150201227)

[Reglas del juego 3](#_Toc150201228)

[Desarrollo de propuesta 3](#_Toc150201229)

[Diseño de clases 3](#_Toc150201230)

[Representación del Tablero 4](#_Toc150201231)

[Reglas del Juego 4](#_Toc150201232)

[C. Diseño 4](#_Toc150201233)

[Justificaciones y Lógica Detrás de las Decisiones de Diseño 5](#_Toc150201234)

[Clase Tablero 5](#_Toc150201235)

[Clase Ficha 5](#_Toc150201236)

[Clase Partida 5](#_Toc150201237)

# Contextualización breve del problema.

El problema consiste en implementar el juego Othello en C++ utilizando la programación orientada a objetos. El juego Othello también conocido como Reversi, es un juego de mesa para dos jugadores que se juega en un tablero cuadrado de 8x8. El objetivo del juego es controlar la mayor cantidad de casillas en el tablero al final del juego. Los jugadores se turnan para colocar sus piezas, que son blancas o negras, en las casillas al lado de las piezas de su oponente. Si un jugador coloca fichas al lado de dos o más fichas del oponente, las fichas del oponente se voltean al color del jugador que hizo el movimiento. El juego termina cuando no quedan más movimientos posibles y gana el jugador que tenga más piezas en el tablero.

# Análisis: consideraciones para el abordaje del problema y desarrollo de propuesta de la estrategia de solución

Consideraciones para el abordaje del problema:

Diseño de clases: Diseñar un conjunto de clases que refleje los elementos principales del juego, como el tablero, las fichas, los jugadores y las reglas del juego.

Representación del tablero: Decidir cómo se representará el tablero. Una matriz bidimensional de celdas (8x8)

Reglas del juego: Esto incluye la lógica para validar movimientos legales, cambiar fichas de color cuando se atrapan entre dos fichas del oponente y determinar cuándo termina el juego.

Desarrollo de propuesta:

### Diseño de clases

Crear una serie de clases que representen los elementos clave del juego, como ‘**Tablero’**, ‘**Jugador’**, y ‘**Partida’**.

* La clase ‘**Tablero’** se encargará de representar el estado del tablero, mantener las fichas y realizar operaciones en el tablero.
* La clase ‘**Jugador’** representará las fichas del juego con un atributo de color (blanca o negra).

La clase ‘**Partida’** coordinará la partida en sí, administrando los turnos de los jugadores y supervisando el estado del juego, esto es si el juego ya finalizó o no.

### ***Representación del Tablero***

Se utilizará una matriz bidimensional para representar el tablero en la clase Tablero, donde cada celda contendrá una instancia de Ficha para indicar su estado (blanca, negra o vacía).

### Reglas del Juego

Implementar las reglas del juego de Othello en la clase ReglasOthello. Esto incluye:

* Validar si un movimiento es legal, o sea, si esta produce un encierro tipo "sandwich".
* Detectar si el juego ha terminado y quién es el ganador.

# TablaDiseño

## Justificaciones y Lógica Detrás de las Decisiones de Diseño

Clase Tablero: Esta clase representa el estado del tablero de juego. Utiliza una matriz de objetos Ficha para representar cada celda. El método validarMovimiento verifica si un movimiento es legal al comprobar si se capturan fichas del oponente.

Clase Ficha: Cada celda del tablero contiene una instancia de esta clase para representar su estado. El atributo color almacena si la celda está vacía, o contiene una ficha blanca o negra. Los métodos obtenerColor y establecerColor permiten acceder y modificar el estado de la ficha de manera controlada.

Clase Partida: Coordinación del juego. Mantiene un registro del tablero y de los jugadores, controla los turnos y finaliza el juego cuando se cumplen ciertas condiciones. El método ejecutarTurno coordina el flujo de juego, y anunciarGanador muestra el resultado de la partida.