

Análisis y Diseño de la Solución

Parcial II

Informática II

Robinson Correa Morales

CC: 1001471074

Juan José Díaz Zuluaga

CC: 1001456540

Docente

Aníbal José Guerra Soler

2023-2

UNIVERSIDAD DE ANTIOQUIA

DEPARTAMENTO DE I. ELECTRÓNICA Y TELECOMUNICACIONES

FACULTAD DE INGENIERÍA

SEDE MEDELLÍN

Octubre 2023

Medellín, Colombia

**Segundo examen parcial**

**1 - Planteamiento del problema:**

Se debe realizar una aplicación de consola que permita jugar el juego de mesa *Othello.* El juego requiere de dos jugadores y se juega sobre un tablero de 8 x 8 aunque también existen versiones de otras dimensiones.

El juego consiste en colocar fichas blancas o negras (según el jugador) e ir cambiando el color de las fichas del oponente según la siguiente regla:

El jugador puede poner una ficha en una posición si y solo si el movimiento encierra de manera lineal (vertical, horizontal y diagonal) una o más fichas del oponente, provocando que las fichas encerradas cambien al color del jugador (se le añaden al total de ficas en el tablero).

Si el jugador no puede hacer un movimiento, pierde el turno.

El juego termina cuando la partida se bloquea (no existen movimientos para ningún jugador) o el tablero está completamente lleno. Gana quien tiene la mayor cantidad de fichas.

Para desarrollar el juego, no se permite el uso de contenedores de la STL, además se deben modelar las distintas clases que representen los objetos y la posibilidad de manejar archivos de texto para tener un historial de partidas.

**2 - Análisis del problema:**

Para realizar el modelado de una partida, se utilizará un enfoque de programación orientada a objetos, en la cual se crearán 4 clases diferentes:

* Clase Juego: La clase juego permite controlar todos los métodos y atributos asociados con la manipulación de la partida, es decir el iniciar una partida, crear un tablero de juego, verificar cuando finaliza la partida y finalmente almacenar los resultados del juego en un archivo de texto.
* Clase tablero: Esta clase permitirá modelar el tablero de juego, con la posibilidad de generar tableros de tamaños y dimensiones pares y mayores a dos.

Parte de esta clase se puede modelar como una matriz compuesta de arreglos dinámicos los cuales contienen los diferentes objetos ficha que participan en el juego.

Estos objetos pueden cambiar de acuerdo con las reglas y además pueden crearse nuevos durante la partida.

Además, el tablero tendrá como atributo, un arreglo con dos objetos jugador, el cual le permitirá saber quiénes están jugando y sus fichas.

* Clase jugador: Esta clase permite modelar los jugadores, sus atributos serán la cantidad de fichas, el color de las fichas y el nombre del jugador.
* Clase ficha: Esta clase será la clase más simple de todas, permitirá modelar una ficha, conociendo su color, siendo blanca (\*), negra (-) o nula (0). La ficha nula será la ficha sin color generada en el inicio del juego, esta ficha es “invisible” y solo puede transformarse en una ficha con color, es decir, una ficha con color no puede transformarse en una ficha nula. A través de un atributo estático será posible conocer cuántos objetos ficha de cada color se han creado.

La clase tablero y ficha tendrán interacción directa. La clase ficha y jugador no tendrán interacción directa, esta ocurrirá a través de la clase tablero.

Para iniciar el juego se ejecuta una función que crea la matriz de la partida que se va a desarrollar siendo esta de tamaño y se inicializan los objetos en la posición cero de la partida, es decir, dos fichas negras y dos fichas blancas en diagonal ubicadas en el centro del tablero.

En cada turno se debe verificar si el jugador tiene movimientos disponibles e indicar cuales son estos, ya que, de no haber, ocurrirá un cambio de turno. Si existen movimientos posibles el jugador debe indicar la posición en la que debe ir la ficha y si la jugada es válida, pasa al siguiente turno y continúa la partida.

En caso de que la posición indicada no sea válida, se volverán a pedir las coordenadas hasta que esta sea correcta.

El juego terminará de dos maneras:

* Si no existen más movimientos posibles para ninguno de los jugadores
* Si el tablero se ha llenado por completo.

Cualquiera de estos dos casos se verificará en la clase tablero y dará por finalizada la partida; el jugador con más fichas sobre el tablero será el ganador.

Al finalizar se deben mostrar los resultados de la partida y almacenarlos el historial de todos los juegos que se han dado. Para ello se deben guardar lo nombres de los jugadores y con cuantas fichas finalizó cada uno en un archivo de texto.

**3 - Diagrama de Clases y estructuras de datos:**

Diagrama

Descripción generada automáticamente

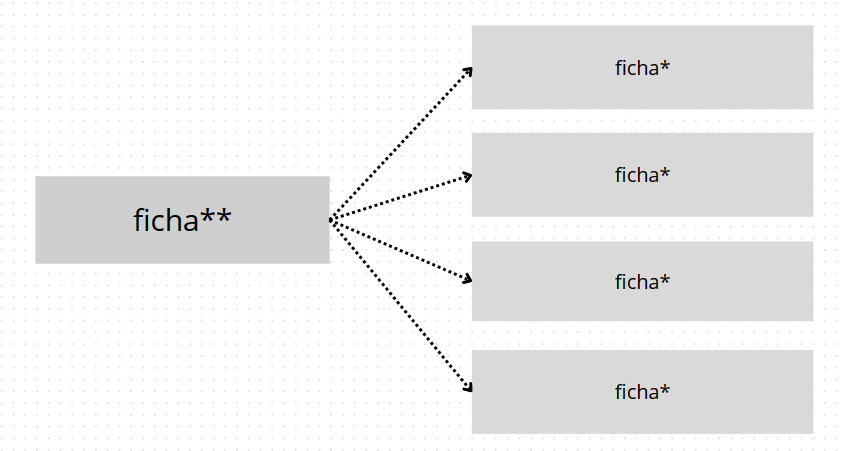
**Estructuras de datos:**

Para seleccionar la estructura de datos del programa se analizaron los diferentes requerimientos de la aplicación entre los cuales estaban el no uso de contenedores STL y el aprovechamiento de la memoria dinámica.

En este caso el tablero estará formado por una matriz dinámica, la cual básicamente se compone de filas de arreglos dinámicos de objetos tipo ficha y un arreglo de apuntadores a las filas mencionadas anteriormente (Concepto de apuntador doble)

Además, la clase tablero poseerá un arreglo estático en la que se almacenarán los objetos jugadores de la partida.

Finalmente, la clase juego utilizará una matriz dinámica para manejar el histórico de partidas. Y esta se borrará tan pronto como la información se pase al archivo de texto, ya que, al tener esta información almacenada en una sección de memoria no volátil, no hace falta conservarla directamente en el programa.



Representación del tablero como arreglo de arreglos utilizando apuntadores.