INFORME PARCIAL 2 INFORMATICA-2 2023-2

REVERSI.

GABRIEL MENDOZA BARRIOS

UNIVERSIDAD DE ANTIOQUIA

OCTUBRE 2023

**Contextualización:**

Se pide al estudiante elaborar un modelo virtual basado en el juego de mesa reversi, que sea funcional en el lenguaje de C++ donde este cumpla y siga todas las reglas del juego original.

**Análisis:**

Lo primero es pensar en que elementos componen al juego de mesa en sí y agregarle uno que otro elemento extra ya que en si este va a ejecutará en un ambiente virtual.

Los elementos que primero se vienen a mi cabeza son:

* **Fichas:** Estos elementos son los que se modifican dentro del tablero después de cada jugada.

Las fichas tienen sus atributos:

* Color: en este caso representadas por caracteres ‘-‘, ‘\*’ para negro y blanco respectivamente.
* Coordenadas: un vector (i,j) que la ayudará a posicionarse espacialmente dentro del tablero.

Las fichas tienen sus métodos:

* Cambio de color.
* **Tablero:** es de donde se van a manejar la cantidad de fichas que puede haber en el tablero y los movimientos y desplazamientos a lo largo de el basado en las reglas del juego.  
    
  Tiene atributos como:
* La matriz contenedora de fichas.
* Fichas.
* Puntajes para ambas fichas.

Métodos:

* Mover ficha/agregar ficha.
* Cambiar color de fichas.

**Nota:** tener en cuenta que para poder modificar de alguna forma las fichas del tablero, los movimientos deben pasar por el filtro de las reglas.

* **Reglas:** Es donde estarán contenidas todas las reglas de desplazamiento de las fichas. Reglas como el fin del juego, el cambio de turno, movimientos validos etc. Las reglas no se si manejarlas como una clase aparte o como una cabecera en la que invoque funciones, eso está por verse más adelante.

Contiene un pack de funciones para autorizar movimientos/cambios de color y fin de juego.

* **Menú:** esta clase contendrá al tablero y se encargará de administrarlo.

Por ahora este es el diseño de solución que se plantea al inicio, este puede y lo más seguro va a estar sujeto a cambios internos, pero me parece por ahora una aproximación general de lo que se puede seguir para armar la solución.

Armé la clase fichas y con presentaba ningún inconveniente por el momento, pero al empezar a trabajar con el tablero me di cuenta de que era quizás poco practico pues en la clase tablero debía generar una matriz de 8x8 que guardara las fichas pero al generar la matriz me pregunté con qué inicializaría dicha matriz en los espacios que no estuviesen ocupando las piezas/fichas de color blanco o negro por lo que ahora pienso añadir una pseudo ficha que contengo como “color” el carácter vacío ‘\0’ de forma que ahora pueda inicializar la matriz con estas pseudo fichas y las originalmente planeadas, pero esto también me hace plantearme si entonces no sería mejor generar una matriz de tipo char, porque finalmente en esencia el “color” es el único atributo de mi clase fichas.

En el proceso también me percate de que no es necesario asignarle como atributo las coordenadas de las fichas, pues al estar cada ficha en una posición de la matriz la coordenada corresponderá a la de su ubicación en la misma.

Ahora hablando un poco más acerca de la matriz que contiene al tablero, pensé en usar un doble puntero para que en caso de que se quiera amplificar el tamaño del tablero solo sea cuestión de generar la matriz nxn que el usuario quiera.

El día 31/10 tuve problemas con el github, ya que al tratar e cambiar la ubicación de la carpeta donde se ejecutan los proyectos en Qt creé conflictos en la rama principal y me tocó borrar todo, gracias a que pude copiar un respaldo antiguo de los archivos pude recuperar 95% de material y el 5% restante era código que había entrado en conflicto.

Lo que se adelantó ayer fue el dibujar la matriz de igual forma al pdf que se proporciona y el movimiento válido, pero olvidé incluir la condición del sándwich donde la ficha del centro debe tener color distinto, por lo que usaré un nuevo condicional antes de desarrollar todos los casos.

04/11 hace unos dos días aproximadamente haciendo los métodos de búsqueda para cambiar color me di cuenta que lo que había definido como movimientos validos estaba mal por lo que teniendo eso en cuenta seguí con los métodos de búsqueda(pues es muy similar a los movimientos validos) y ya con eso terminado hoy terminé lo que seria la estructura para los movimientos válidos y métodos de búsqueda para cambiar de color, en el repositorio se encuentran las imágenes, estas estructuras están sujetas a cambio en el código por lo que siempre la versión del código va a ser la más actual.