Informe del trabajo

El programa incluye los archivos:

Carta.h

CartaAbstracta.cpp

CartaAbstracta.h

CartaCambiarColorDeUnaFicha.cpp

CartaCambiarColorDeUnaFicha.h

CartaCrearPortal.cpp

CartaCrearPortal.h

CartaEliminarFicha.cpp

CartaEliminarFicha.h

CartaIgnorarMovimiento.cpp

CartaIgnorarMovimiento.h

CartaIntercambioDeFichas.cpp

CartaIntercambioDeFichas.h

CartaRobarCarta.cpp

CartaRobarCarta.h

Celda.h cambios

CuatroEnLinea

CuatroEnLinea.cpp

CuatroEnLinea.h

Declaraciones.h

ExcepcionError.cpp

ExcepcionError.h

Interfaz.cpp

Interfaz.h

Jugador.cpp

Jugador.h

Lista.h

ListaSimple.h

Nodo.h

Tablero.cpp

Tablero.h

main.cpp

Explicación de los archivos usados:

Interfaz: Se encarga de mostrar y pedir todos los datos, por ejemplo, pedir los datos para las dimensiones del tablero y la cantidad de jugadores, mostrar el tablero, las cartas que tiene un jugador,etc.

Tablero: Se encarga de obtener y verificar los datos que ingresa un jugador para poder ingresar una ficha.

Jugador: se encarga de realizar acciones como ingresar una ficha al tablero, utilizar una carta(si este posee una).

Celda: Se encarga de obtener y modificar el color de una ficha usando como referencia el color del jugador

CartaAbstracta: Clase abstracta de cartas, se encarga de modificar y obtener la cantidad de cartas usadas, obtenidas y entregadas, como también de consultar que tipo de carta es, su descripción. Las clases derivadas son:

1. Carta robar carta: Se encarga de quitar una carta existente de un jugador y generarla en el jugador usuario.
2. Carta intercambiar fichas: Se encarga de intercambiar dos fichas consiguientes entre sí, elegidas por el usuario
3. Carta cambiar color: se encarga de cambiar el color de una ficha específica elegida por el usuario

4. Carta crear portal: se encarga de crear un portal temporal en dos posiciones del tablero mientras sean laterales elegidas por el usuario, para seguir la formación de cuatro en línea de las fichas en esas posiciones
5. Carta ignorar anterior: se encarga de eliminar la última ficha ingresada de un jugador
6. Carta eliminar ficha: se encarga de eliminar una ficha en el tablero elegida por el usuario

CuatroEnLinea: Se encarga de realizar todas las acciones del juego interactuando con los archivos jugador, cartas, tablero, celda.

Declaraciones: Se encuentran definidas todas las constantes, clases y estructuras para los demás archivos .cpp

Conclusión:

Se utilizó using namespace solo en los archivos cpp.

Para las cartas se utilizó herencia para crear los poderes de cada carta en una clase distinta.

Para el tablero, se utilizó una lista de listas de listas de puntero a jugador, para un mejor forma de acceder a una celda en especifico

Las celdas son una estructura dinámica parecido a los nodos, solo que en este caso se usan específicamente para las listas que forman la clase tablero

Respuestas del cuestionario:

¿Qué es un SVN?

Apache subversion (SVN) es una herramienta de control de versiones open source basada en un repositorio cuyo funcionamiento se asemeja enormemente al de un sistema de ficheros. Es software libre bajo una licencia de tipo Apache/BSD.

¿Que es “GitHub”?

Git es un software de control de versiones diseñado por Linus Torvalds, pensando en la eficiencia y confiabilidad del mantenimiento de versiones de aplicaciones cuando estas tienen un gran número de archivos de código fuente. Su propósito

es llevar registro de los cambios en archivos de computadora y coordinar el trabajo que varias personas realizan sobre archivos compartidos.

¿Que es “collabnet” y “Tigris”?

CollabNet VersionOne es un proveedor de soluciones de desarrollo y entrega de software con sede en Alpharetta, GA. Los productos y servicios VersionOne pertenecen a las categorías de la industria de administración de flujo de valor, devops, administración ágil, administración del ciclo de vida de la aplicación (ALM) y control de versiones empresariales. Los productos de este tipo son utilizados por compañías y organizaciones gubernamentales para reducir el tiempo que lleva crear y lanzar software.

Manual de usuario:

Al iniciar el programa, se debe ingresar las dimensiones del tablero(fila, columna y profundidad), estos datos deben ser al menos dos mayores o iguales a 4.

Luego se ingresará la cantidad de jugadores que habrá en el juego, que estará entre 2 y una cantidad máxima de jugadores que determinara el programa.

La cantidad de fichas por jugador depende de las dimensiones del tablero y de la dificultad elegida.

Para cado turno de un jugador:

Se mostrará por consola, el tablero, que está dividido por capas, las cartas que tiene disponible, al principio se mostrará que está vacío.

Se pedirá si quiere usar una carta, si lo usa , se realizará las acciones necesarias que afectarán al tablero y/o a un jugador elegido.

Se pedirá que ingrese la fila y la columna en donde quiere que ingrese su ficha.

Todo este proceso se repetirá hasta que uno de los jugadores haya formado 4 en línea en cualquier dimensión, o hasta que el tablero este lleno de fichas.

Al final se mostrará por pantalla el jugador que haya ganado o empate (si ningun jugador logró ganar).

Manual del programador:

Para este programa usamos los siguientes útiles

- Lenguaje C++
- Librería estándar de C++
- Libreria cstdlib
- librería ctime
- Las siguientes clases
    a. Tablero
    b. Celda
    c. Jugador
    d. Cartas
    e. CuatroEnLinea
- Una biblioteca
    a. Interfaz.hpp
    b. Interfaz.cpp
- programa principal
    a. main.cpp

Para hacer código en este programa nos adherimos a las siguientes normas:

Apéndice A

1. Usar las siguientes convenciones para nombrar identificadores.
   Clases y structs: Los nombres de clases y structs siempre deben comenzar con la primera letra en mayúscula en cada palabra, deben ser simples y descriptivos.
   Se concatenan todas las palabras. Ejemplo: Coche, Vehiculo, CentralTelefonica.
2. Métodos y funciones: Deben comenzar con letra minúscula, y si está compuesta por 2 o más palabras, la primera letra de la segunda palabra debe comenzar con mayúscula. De preferencia que sean verbos. Ejemplo: arrancarCoche(), sumar().
3. Variables y objetos: las variables siguen la misma convención que los métodos. Por Ejemplo: jugadorBuscado, padronElectoral.
4. Constantes: Las variables constantes o finales, las cuales no cambian su valor durante todo el programa se deben escribir en mayúsculas, concatenadas por "_". Ejemplo: ANCHO, VACIO, COLOR_BASE.

5. No usar sentencias 'using namespace' en los .h, solo en los .cpp. Por ejemplo, para referenciar el tipo string en el .h se pone std::string.
6. No usar 'and' y 'or', utilizar los operadores '&&' y '||' respectivamente.
7. Compilar en forma ANSI. Debe estar desarrollado en linux con eclipse y g++. Utilizamos el estándar C++98.
8. Comentar el código. Todos los tipos, métodos y funciones deberían tener sus comentarios en el .h que los declara.
9. No inicializar valores dentro del struct o .h.

Aclaraciones:

Para Cartas:

Se utilizó herencia para los poderes de cada carta,para poder usar solamente el método activar

Para Tablero:

El tablero crea las filas y las columnas, pero las celdas se crean progresivamente cada vez que un jugador ingresa una ficha.

Las celdas están en una lista doblemente enlazada, para poder obtener una verificación mas rapida y eficiente usando los punteros anterior y siguiente.

Las celdas contienen un puntero a jugador para poder obtener el color que lo representa.

Para verificación:

Se utilizó una lista que guarda todas las direcciones de las celdas adyacentes a la última ficha ingresada y una función recursiva que verifica si la celda siguiente tiene el mismo color de ficha que la última ficha ingresada

Para los niveles:

Se decidió que para cada nivel, se ofrecerá una cierta cantidad de cartas y un cierto tipo de cartas. Ej: en el nivel fácil los jugadores obtendrán cartas regularmente y una gran posibilidad de obtener las mejores cartas (ej:robar carta)