75.41 - Algoritmos y Programación II

Cátedra Ing. Patricia Calvo - 2do cuatrimestre 2023

Trabajo Práctico 2: TESORO BINARIO V2.0

Objetivo

Generar una pieza de software que simule el funcionamiento del juego Tesoro Binario en su versión multijugador Jugador.

Enunciado

Tesoro Binario es un juego de mesa para N jugadores en el que se introducen M tesoros por cada jugador en un tablero de X por Y por Z con el fin de que los tesoros no sean encontrados hasta el final. En cada turno el jugador el jugador puede:

a) Sacar una carta del mazo, para jugarla o guardarla. Ver las opciones de cartas.
b) Atacar una posición del tablero con una tesoro mina, eliminando si hay un tesoro en ella y dejando la casilla inactiva por tantos turnos dependiendo el poder de la mina, o dejándolo a la espera de ser descubierto y el jugador que lo descubre, pierde el turno.
c) En cada turno el jugador deja un espía en un casillero del tablero, recuperando el tesoro si hay uno en ella o dejando la casilla con el espía esperando que se mueva un tesoro. Cuando el espía encuentra el tesoro, tarda 5 turnos en recuperarlo dejando la casilla inactiva por esos turnos. Si el jugador pone un espía en una casilla con un espía contrario, ambos espías son eliminados del juego.
d) Mover un tesoro de lugar. Si un tesoro se mueve a una casilla con un tesoro, el juego le

avisa al jugador para que mande un espía y lo pueda recuperar. El jugador no puede mover el tesoro a una casilla en proceso de que el espía está recuperando el tesoro.

El juego tiene que tener 6 cartas disponibles, 3 dadas por el enunciado y 3 definidas por el grupo. Las cartas dadas por el enunciado son:

1) Blindaje: bloquea la captura del tesoro por una cantidad de turnos.
2) Radar: colocado en una posición, detecta 3 casilleros a la redonda si hay o no tesoros.
3) Partir tesoro: permite partir un tesoro en 2 casilleros.

El juego termina cuando todos los jugadores se quedan sin tesoros salvo uno, ganando este jugador.

Interfaz de usuario

Toda la interfaz de usuario debe estar basada en texto, la entrada y salida y el estado del tablero tiene que mostrarse utilizando un Bitmap (ver librería) de una manera ideada por el grupo.

No es necesario que se limpie la pantalla, simplemente escribir el estado del tablero luego de cada jugada.

Cuestionario

Responder el siguiente Cuestionario:

1) ¿Qué es un svn?
2) ¿Qué es git?
3) ¿Qué es Github?
4) ¿Qué es un valgrind?

Normas de entrega

Trabajo práctico grupal: 6 personas. Se deberá elegir un nombre de Grupo.

Reglas generales: respetar el Apéndice A.

El tablero se debe implementar utilizando la clase Lista. Todos los objetos deben estar en memoria dinámica.

Se deberá subir un único archivo comprimido al campus, en un link que se habilitará para esta entrega. Este archivo deberá tener un nombre formado de la siguiente manera:

Nombre de Grupo-TP2.zip

Deberá contener los archivos fuentes (no los binarios), el informe del trabajo realizado, las respuestas al cuestionario, el manual del usuario y el manual del programador (Todo en el mismo PDF).

La fecha de entrega vence el día viernes 03/11/23 a las 23.59hs.

Se evaluará: funcionalidad, eficiencia, algoritmos utilizados, buenas prácticas de programación, modularización, documentación, gestión de memoria y estructuras de datos.

Apéndice A

1) Usar las siguientes convenciones para nombrar identificadores.
    a) 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.
    b) 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().
    c) Variables y objetos: las variables siguen la misma convención que los métodos. Por Ejemplo: alumno, padronElectoral.
    d) 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.

2) El lenguaje utilizado es C++, esto quiere decir que se debe utilizar siempre C++ y no C, por lo tanto una forma de darse cuenta de esto es no incluir nada que tenga .h, por ejemplo #include <iostream> .
3) 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.
4) No usar 'and' y 'or', utilizar los operadores '&&' y '||' respectivamente.
5) Compilar en forma ANSI. Debe estar desarrollado en linux con eclipse y g++. Utilizamos el estándar C++98.
6) Chequear memoria antes de entregar. No tener accesos fuera de rango ni memoria colgada.
7) Si el trabajo práctico requiere archivos para procesar, entregar los archivos de prueba en la entrega del TP. Utilizar siempre rutas relativas y no absolutas.
8) Entregar el informe explicando el TP realizado, manual de usuario y manual del programador.
9) Comentar el código. Todos los tipos, métodos y funciones deberían tener sus comentarios en el .h que los declara.
10) Modularizar el código. No entregar 1 o 2 archivos, separar cada clase o struct con sus funcionalidades en un .h y .cpp
11) No inicializar valores dentro del struct o .h.
12) El tablero en el TP 2 tiene que ser listas de listas.
13) Si cualquier estructura de control tiene 1 línea, utilizar {} siempre, por ejemplo:

```
for(int i = 0; i < 10; i++) {
      std::cout << i;
}
```