**FIUBA - 75.07**

**Algoritmos y programación III**

*Trabajo práctico 2: Al-Go-Oh!*

1er cuatrimestre, 2018

(trabajo grupal de 4 integrantes)

Alumnos:

|  |  |  |
| --- | --- | --- |
| **Nombre** | **Padrón** | **Mail** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Informe

Desarrollamos esta entrega parcial de nuestro trabajo práctico con la metodología TDD.

Examinamos los puntos requeridos para esta entrega y realizamos pruebas unitarias concordando con lo pedido, una o dos pruebas por punto.

Luego, hicimos los diagramas tal que proyectaran lo escrito en las pruebas. Una vez escrito el código debimos actualizar nuestros diagramas.

La idea de este trabajo es realizar la aplicación del juego Yu-Gi-Oh adaptado como Al-Go-Oh.

Decidimos hacer una base de datos de las cartas e implementarlas como ArrayLists con la cantidad definida por el enunciado. Por ejemplo, las cartas Monstruo y efecto deben ser 5, las del Mazo 40, etc.

El el juego habrá un tablero dividido en dos campos, cada uno de cada jugador.

Jugador:

Tendrá los siguientes atributos:

private String nombreJugador;

private int puntosDeVida;

private Campo campoDeJuego;

private Mazo mazo;

El campoDeJuego será el campo correspondiente a cada jugador en el tablero.

Cada jugador iniciará la partida con 8000 puntos de vida. Pierde el jugador que se quede sin puntos, o sin cartas para tomar del mazo, lo que suceda primero.

Además, esta clase tendrá los métodos tal que sea capaz de destruir una carta, es decir mandarla al cementerio, agregar una carta, ponerla en juego en su campo, uno que le devuelve un booleano dependiendo de si la carta está en juego o no pasandole una carta específica y otro método para saber el estado de una carta determinada.

Mazo:

Cuando se agrega una carta al mazo, esta se busca en la base de datos de las cartas. En nuestra base de datos esta carta es buscada por el nombre.

BaseDatosCartas:

Será implementado con un HashMap<> que tendrá como claves los nombres de las cartas: String y como valores las instancias de la clase Carta.

Tendrá los métodos tal que sea capaz de decir si contiene o no una carta, y saber buscar una carta pedida.

EstadoCarta:

Los distintos estados de las cartas heredan de esta clase. Estos son EstadoHorizontalBocaArriba, EstadoVerticalBocaArriba y EstadoVerticalBocaAbajo.

Esto para definir las tácticas y los diferentes efectos que tiene en el puntaje de vida cuando el jugador utiliza las distintas estrategias existentes en el juego.

Zona:

De esta clase heredan las distintas zonas del campo de cada jugador. Contiene el atributo removerCarta que como indica el nombre quita una carta de la zona designada.

Estas son: ZonaCampo, ZonaDeJuego y ZonaCementerio.

ZonaDeJuego:

De esta clase heredan las clases ZonaAtaque y ZonaDefensa.

Implementadas ambas con un atributo casilleros que es un ArrayList y con los métodos agregarCarta y removerCarta y preguntar si la carta está EnJuego.

ZonaCemenerio:

Acá es donde van a parar las cartas que pierden un enfrentamiento.

Implementada con un Stack de instancias de la clase Carta. Con los mismos métodos que las clases anteriores.

Cartas:

Con los atributos nombre y estado. Se inicializa la carta con un estado VerticalBocaAbajo.

Con los métodos cambiarEstado , obtenerNombre y agregarCarta(abstracto).

Las dos primeras nos sirven para poder poder manejar las cartas en juego y poder buscarlas en las base de datos o las distintas zonas respectivamente.

CartaCampo, CartaEfecto, CartaMonstruo heredan de esta clase.

A su vez, CartaMagica y CartaTrampa heredan de CartaEfecto. Tienen un método ejecutarEfecto que hace que la carta active el efecto correspondiente de la carta en juego.

Las distintas clases de cartas son necesarias porque cada una tiene distintas respuestas mientras se va desarrollando el juego.

Campo:

private ZonaAtaque zonaAtaque;

private ZonaDefensa zonaDefensa;

private Mazo mazo;

private ZonaCementerio cementerio;

private ZonaCampo zonaCampo;

Como vemos cada campo tiene sus distintas zonas y un mazo de cartas de donde el jugador levantará.

Lo que hace esta clase es agregar las cartas del jugador a las distintas zonas de su campo.