LEGENDS OF WAR

Implementación de los Requerimientos:

1.POO y Relación entre clases:
Se han utilizado clases como , **Personaje , Region , HeroeAtaque, y HeroeDefensa ,**aplicando herencia y polimorfismo.

Clase Personaje es una clase abstracta que será extendida por clases como HeroeAtaque y HeroeDefensa…
Cada clase que extiende el Personaje tiene un constructor con valores distintos !
En Personaje.java hay metodos abstractos como :

- atacar();
- defender();
- curar();
- pocion();

Cada uno tendrá un uso/efecto diferente dependiendo de la clase que está llamando a método.
Por ejemplo tenemos un sistema de diálogo que será en modo random entre unas frases, cada clase tiene sus propias frases dentro de estos métodos.
O
Por ejemplo mientras HeroeDefensa tiene un función en su método curar(); , HeroeAtaque todavía no tiene ninguna función.

Region también es un clase abstracta extendida por Ionia y noxus , cada región dará a los héroes bonuses de Vida , Ataque , Armadura distintos usando métodos abstractos.

2.Tipos de Personajes:
HeroeAtaque y **HeroeDefensa** tienen atributos y métodos específicos según su rol.
HeroeAtaque tiene mayor daño pero no puede curar mientras que **HeroeDefensa** puede curar aliados y tiene mas defensa
También los pots(pociones) tendrán un efecto diferente dependiendo de cada rol.
(más info en manual de juego o comentarios de códigos)

3.Uso de array:
La clase **Map** contiene una array bidimensional ([3][3]) que representa el mapa del juego y almacena las posiciones de los héroes
Métodos como placePlayer() , PrintMap gestionan el mapa.

4.Menu de main :
Menú hemos aplicado como un bucle dentro de main , una vez que se ejecutan el juego tendrán opciones de :

- Empezar
- Manual de Juego
- Salir

Mientras el jugador no quiera salir , estará dentro de bucle .

5.Sistema de combate:
Implementado en **turno.java ,** donde se manejan los ataques , defensas y turnos de los jugadores.
atack() aplica el daño basado en los valores de ataque y defensa de los personajes.
defender() es mas como recibir el ataque por que en el juego el único momento que vas a defender es cuando recibes un ataque por este método defender(); reduce la vida del personaje si el daño recibido supera su defensa (valor de armadura)
Si el jugador que quiere atacar no localiza el enemigo o ya esta muerte el enemigo en dicha posición , el ataque no será completado. (uso de bucle principal dentro de metodo atack();)

6. Condiciones de Fin del juego:

Se determina el ganador cuando los dos jugadores de un equipo tienen vida igual a 0
Usando método winner(); dentro de bucle

7. Ampliaciones:

Tablero : Uso de array bidimensional para colocar héroes en una mapa y después de cada ataque recibido actualizar el tablero

Opciones de Utensilio : El jugador puede elegir entre 2 diferentes tipos de Utensilio que afectará a su poder o armadura

Equipos: Cambiar a modo de juego a multiplayer (2 equipos de 2 jugador contra 2 equipos de 2 jugador)

Regiones : Hemos dado el opción de elegir el región para tener diferentes bonuses en el resto de juego

Localizar Enemigo: con el uso de tablero , denegar el ataques que seran realizado a posiciones que actualmente no hay ningún enemigo o el enemigo en dicha posición no esta vivo

Curación: A pesar de dar valores diferentes a atributos de cada tipo de personaje , Heroes de defensa tendrán opción de curar!

M.Parsa Pour Shahroodi

Proyecto de programación

Enlace para acceso a repositorio: https://github.com/Parsa-P3/LegendsOfWar.git

26/02/2025