Título del Proyecto:

Juego de Texto Interactivo: Combate por Turnos

Integrantes del Equipo:

Nixon Verjel , 192315

Santiago Forero, 192433

Juan David Polanco, 192400

Andre Julián Granados, 192329

Fecha de Presentación:

6 / 12 / 2024

Introducción

Contexto del Problema:

En la actualidad, los juegos de texto interactivo han recuperado popularidad como herramientas educativas y recreativas. Estos juegos combinan narrativas inmersivas con mecánicas de juego estratégicas, desafiando al usuario a tomar decisiones que impactan el desarrollo de la historia. Este proyecto aborda la creación de un juego de combate por turnos con énfasis en roles de personajes y dinámicas de combate.

Objetivos del Proyecto:

Diseñar e implementar un juego interactivo que permita al usuario asumir el rol de un personaje y enfrentarse a enemigos en un sistema de combate por turnos.

Integrar un sistema de roles especializados (guerrero y mago) con habilidades únicas y efectos específicos sobre diferentes enemigos.

Crear un código modular, reutilizable y extensible para futuras mejoras o adiciones.

Alcance del Proyecto:

El juego actual incluye:

Sistema de combate por turnos entre personajes y enemigos.

Clases diferenciadas de personajes y roles con habilidades especializadas.

Eventos aleatorios para aumentar la rejugabilidad y el dinamismo.

Un flujo narrativo básico que guía al jugador en sus decisiones dentro del combate.

Documentación Técnica

Estructura del Proyecto:

El proyecto se divide en los siguientes componentes principales:

Clase Padre: Personaje

Representa la base para todos los personajes del juego.

Incluye atributos básicos como salud (vida) y métodos como estaVivo para determinar el estado del personaje.

Subclases de Personaje:

Jugador:

Se especializa en dos roles:

Guerrero: Inflige daño adicional a enemigos como animales y quimeras.

Mago: Inflige daño adicional a enemigos como fantasmas y esqueletos.

Métodos principales:

atacar: Aplica daño al enemigo según las reglas del rol.

defenderse: Reduce el daño recibido a la mitad.

escapar: Probabilidad aleatoria del 50% para huir del combate.

Enemigo:

Hereda atributos y métodos de la clase Personaje.

Utiliza el método atacar para interactuar en el combate.

Clase Juego:

Administra el flujo del juego, especialmente el sistema de combate por turnos.

Métodos clave:

pelear: Maneja la interacción entre personajes y enemigos, validando condiciones de victoria, derrota o escape.

Estado del Combate: Se muestran los puntos de vida de los personajes al inicio y al final de cada turno para mayor claridad del jugador.

Organización Modular del Código

El código se ha estructurado en archivos separados para mantener la modularidad y facilitar el mantenimiento:

Clase Main:

Se encuentra en un archivo independiente y es responsable de ejecutar el flujo principal del juego.

Carpeta Game:

Contiene los archivos de las clases base y las especializaciones del juego:

Personaje.java: Contiene la clase base con atributos y métodos comunes.

Jugador.java: Incluye las especializaciones del jugador (Guerrero y Mago), cada una con habilidades específicas y bonificaciones.

Enemigo.java: Define los atributos y comportamientos de los enemigos, incluyendo sus tipos y roles en el combate.

Métodos clave:

pelear: Maneja la interacción entre personajes y enemigos, validando condiciones de victoria, derrota o escape.

Estado del Combate: Se muestran los puntos de vida de los personajes al inicio y al final de cada turno para mayor claridad al jugador.

Tecnologías Utilizadas:

Lenguaje de Programación: Java

Bibliotecas:

java.util.Scanner: Captura la entrada del usuario durante el juego.

java.util.Random: Genera eventos aleatorios como la probabilidad de escape.

java.util.HashMap : Gestiona de manera eficiente las bonificaciones y atributos dinámicos en el sistema de roles.

Decisiones Técnicas:

Orientación a Objetos:

Se utilizó herencia y encapsulamiento para estructurar las relaciones entre las clases. Esto asegura modularidad y facilita la extensión del código.

Eventos Aleatorios:

El uso de generación aleatoria en decisiones como escapar mejora la rejugabilidad y aporta dinamismo al juego.

Diseño Reutilizable:

El método pelear está diseñado para ser reutilizable en múltiples escenarios de combate.