TRABAJO PRÁCTICO: El Camino de Gondolf

Integrantes del grupo:

- Frias, Roman.

Legajo:47.095.626
Mail: romanfriasyf@gmail.com

- Obregón, Lucas.

Legajo:45.612.732
Mail: lucasoobregon04@gmail.com

- Sanchez, Lara Malen.

Legajo: 45.992.743
Mail: mmsanchez743@gmail.com

- Villaverde, Giuliana Ailén.

Legajo: 46.830.077
Mail: giulianvillaverde@gmail.com

El trabajo consistia en recrear un videojuego que contenia a un mago y a unos murcielagos tratando de matarlo, el mago tenia que tener sus hechizos de diferentes tamaños y gastos de energia y esquivar piedras en el camino.

Implementamos 5 clases ademas de Juego.java, (mago, hechizos,murcielago,piedra y pocion.)

CLASE MAGO: En esta clase el mago recibe todos los valores para su funcionamiento, como altura, las coordenadas en x, y, ancho y alto, etc

CLASE MURCIELAGO: Esta clase contiene lo mismo casi que la del mago pero con la implementacion de calcular la distancia entre el murcielago y tal objetivo

CLASE HECHIZOS: No contiene mucho, solo inicia el costo de energia y el efecto del radio del hechizo.

CLASE PIEDRA: No contiene mucho tampoco, le ajustamos los bordes, dibujamos la imagen y le damos valores de coordenada.

CLASE POCION: Esta clase contiene el metodo recoger para saber si el mago recogio o no la pocion,calcula la distancia entre el mago y la pocion y se le dan valores de coordenada.

Variables más importantes en la clase Juego:

1. **Entorno entorno**
    - **Resumen:** Esta variable representa el entorno gráfico donde se dibuja todo y se manejan las interacciones del usuario (teclado, mouse). Es la interfaz principal con el motor del juego.

2. `mago mago`
    - **Resumen:** Representa al personaje principal (el jugador). Contiene su posición (`x`, `y`), vida, y otros atributos relacionados con el jugador.
3. `murcielago[] murcielagos`
    - **Resumen:** Un array de objetos de tipo `murcielago` (los enemigos). Es vital para la lógica de los enemigos, su movimiento, su dibujo en pantalla y la detección de colisiones.
4. `piedra[] piedras`
    - **Resumen:** Un array que contiene los objetos `piedra`, actuan como obstaculos dentro del mapa
5. `pocion[] pociones`
    - **Resumen:** Un array para guardar objetos `pocion`. Las pociones son ítems recolectables que probablemente otorgan magia al mago

6. `Image fondo`, `Image panelHechizos`, `Image corazon`, `Image energia`, `etc` **(y los gifs)**
    - **Resumen:** las variables `Image` son cruciales porque cargan y almacenan todas las imagenes, las pantallas de fin de juego, las animaciones de los hechizos, etc). Sin ellas, el juego no tendría representación gráfica.

7. `Hechizo hechizoCrystalExplosion`, `Hechizo hechizoFireball`, `Hechizo hechizoSilverLining`

    - **Resumen:** Estas variables representan los diferentes tipos de hechizos que el mago puede usar. Almacenan información sobre cada hechizo como su nombre, costo de energía, y radio de efecto
8. `boolean estaEnGameOver`

    - **Resumen:** Un booleano que controla el estado del juego. Si es `true`, el juego está en la pantalla de "Game Over" y la lógica de juego normal se detiene.
9. `boolean juegoCompletado`

    - **Resumen:** Similar a `estaEnGameOver`, esta indica si el jugador ha completado todas las oleadas del juego y, por lo tanto, ha ganado.
10. `String hechizoActivo` y `String hechizoaLanzar`

    - **Resumen:** Estas variables (`String`) modifican el estado actual de los hechizos. `hechizoaLanzar` indica qué hechizo ha sido seleccionado por el jugador, mientras que `hechizoActivo` indica qué hechizo se selecciono
11. `int ticksHechizo`

    - **Resumen:** Un contador de ticks que controla la duración de la animación y el efecto de un hechizo activo.
12. `Point posLanzamiento`

- **Resumen:** Almacena las coordenadas (x, y) donde el hechizo fue lanzado, lo que es crucial para determinar el centro de efecto de los hechizos de área.

13. `int energiaMagica`

    - **Resumen:** Representa la cantidad actual de energía mágica del mago, que se consume al lanzar hechizos. es vital para el mago y para el combate

CODIGO BASE:

```
public class Juego extends InterfaceJuego {
    private Entorno entorno;

    // Variables y metodos propios de cada grupo
    // ...

    public Juego() {
        // Inicializa el objeto entorno
        entorno = new Entorno(this, "El camino de Gondolf - Grupo 3 - v1", 800, 600);

        // Inicializar lo que haga falta para el juego
        // ...

        // Inicia el juego
        entorno.iniciar();
    }

    public void tick() {
        // Procesamiento de un instante de tiempo
        // ...
    }

    public static void main(String[] args) {
       Juego juego = new Juego();
    }

}
```

En conclusion, coincidimos en que el trabajo fue muy dificil, tuvimos que aprender muchas cosas para poder implementarlas al trabajo, en relacion a eso, hay muchas cosas que probabamos en el codigo y no funcionaban, tuvimos problemas para cargar ciertas imagenes pero de a poco fuimos terminando todo, nos gusto como nos quedo el juego, por ahi se podria mejorar esteticamente pero pudimos hacer que funcione correctamente.