Explicación de la solución - Simulador de Batalla de Robots

Para este laboratorio, desarrollé un programa en Java que simula una batalla entre robots. La solución está dividida principalmente en dos clases: Robot y JuegoBatalla. El objetivo era poner en práctica el uso de arrays, clases y lógica básica de programación orientada a objetos. A continuación explico cómo resolví cada parte.

Clase Robot

La clase Robot representa a cada personaje que va a pelear en la simulación. Cada robot tiene cuatro atributos: el nombre, los puntos de vida, el poder de ataque y un valor de defensa.

Vida: está entre 50 y 100, y se va reduciendo durante las batallas.

Ataque: indica cuánto daño puede causar un robot (entre 10 y 20).

Defensa: actúa como un escudo que se suma temporalmente a los puntos de vida antes del ataque. Después del primer golpe, se vuelve 0.

También tiene un método Atacar que permite a un robot atacar a otro, y un método EstaVivo() que verifica si el robot sigue en pie. Además, sobreescribí el método toString para poder mostrar fácilmente la información del robot en consola.

Clase JuegoBatalla

Esta clase contiene toda la lógica del juego y la interacción con el usuario.

Usé un ArrayList para guardar hasta 10 robots que se crean manualmente por el usuario desde consola.

El programa tiene un menú principal que permite: agregar robots, iniciar la batalla, ver los robots registrados o salir del programa.

Antes de registrar un robot, el programa valida que los valores ingresados estén dentro del rango permitido (por ejemplo, vida entre 50-100, ataque entre 10-20, etc.).

La batalla

La parte más interesante es la simulación de la batalla. Cuando hay al menos dos robots, se puede iniciar el combate. En cada ronda:

1. Cada robot ataca a otro elegido aleatoriamente (pero nunca a sí mismo).

2. Se ejecuta el método Atacar para reducir la vida del objetivo.

3. Si un robot llega a 0 o menos puntos de vida, se elimina del ArrayList.

4. El proceso se repite hasta que solo queda uno en pie o el usuario decide detener la batalla.

También agregué una opción extra para que el usuario pueda decidir si quiere seguir después de cada ronda, lo que permite ver el estado de los robots vivos y parar si se desea.

Mostrar al ganador

Al final, si queda un solo robot, el programa muestra su nombre y cuánta vida le quedó. Si todos fueron eliminados al mismo tiempo, se indica que no hubo ganador.

En general, fue una práctica muy útil para reforzar el uso de clases, arrays y estructuras de control. Además, el sistema de menús y validaciones le da una experiencia más completa al usuario.

Christopher Chacón Mora
Carné: C32037