Práctica 1 - JavaQuest: Un nuevo comienzo.

Los juegos por turnos son una categoría fascinante de videojuegos que se basan en la toma de decisiones estratégicas y tácticas en cada turno. Estos juegos a menudo emulan situaciones de combate, donde los jugadores deben planificar sus movimientos y acciones cuidadosamente para derrotar a sus oponentes. Un ejemplo icónico de este género es el mundo de Pokémon, donde los entrenadores y sus criaturas se enfrentan en batallas por turnos.

Como todos sabemos el flujo de un juego por turnos sigue estos pasos:

1. Selección de acciones: En cada turno, tanto el jugador como el oponente pueden elegir una acción para sus personajes (por ejemplo, atacar, defender, usar una habilidad especial o cambiar de criatura).
2. Resolución del turno: Una vez que se han seleccionado las acciones, el turno se resuelve. Las acciones pueden tener diferentes efectos, como infligir daño al oponente, curar al propio personaje o aplicar estados especiales como parálisis o quemaduras.
3. Actualización de estados: Después de la resolución del turno, se actualizan los estados de los personajes. Esto incluye la reducción de puntos de vida (HP) por daño sufrido, la aplicación de efectos secundarios y otros cambios en las estadísticas.
4. Turno del oponente: Luego, el oponente selecciona y ejecuta su acción. El proceso se repite hasta que uno de los personajes quede sin HP y se declare al ganador.

Apartado 1: Creación de personajes (2 puntos)

Solicita al usuario que elija un personaje para jugar entre tres opciones: Guerrero, Mago y Arquero. Cada personaje tiene diferentes estadísticas, como puntos de vida, puntos de ataque y puntos de defensa. Almacena estas estadísticas en variables.

Apartado 2: Creación del enemigo (0.5 puntos)

Genera aleatoriamente un enemigo para el jugador con estadísticas similares a las del personaje elegido. Esto incluye puntos de vida, puntos de ataque y puntos de defensa del enemigo. Utiliza números aleatorios para generar estas estadísticas.

Apartado 3: Implementación de la batalla (4 puntos)

Inicia la batalla por turnos entre el jugador y el enemigo. En cada turno, el jugador puede elegir entre atacar o defenderse. Implementa un sistema de ataque y defensa que tome en cuenta las estadísticas de los personajes y enemigos. Actualiza los puntos de vida de cada uno después de cada turno.

Apartado 4: Finalización de la batalla (0.5 puntos)

Continúa la batalla hasta que uno de los dos, el jugador o el enemigo, quede sin puntos de vida. Cuando eso suceda, muestra un mensaje que indique quién ganó la batalla.

Apartado 5: Interfaz de usuario (0.5 puntos)

Todas las interacciones con el usuario deberán seguir una estructura como en las siguientes figuras:

```
**********************
* 1.Guerrero  2.Mago *
* 3.Arquero          *
**********************
```

```
**********************
* 1.Atacar  2.Defender *
* 3.Habilidad Especial *
**********************
```

Apartado 6: Generación de sprites (0.5 puntos)

Genera con asteriscos una imagen de cada tipo de personaje que aparecerá en la selección de personaje y durante el transcurso de la batalla. Además, muestra con una barra de vida tanto la vida del personaje como la del enemigo.

Apartado 7: Enemigo inteligente (1 punto)

En lugar de que el enemigo ataque en cada turno, deberá seguir los siguientes patrones:

Fase 1 (mientras tenga más de un 15% de vida):

- En los turnos que sean número primos hace un ataque normal.
- En los turnos pares que no sean primos bloquea.
- En los turnos impares que no sean primos utilizará su habilidad especial.

Fase 2 (mientras tenga menos de un 15% de vida):

- Seguirá el patrón curar, bloqueo/ataque (50% de probabilidad), ataque especial.

Se deberá entregar en la tarea asignada para ello el fichero .java que contiene el código.

Rúbrica de evaluación:

Apartado 1 (2 puntos):

- El programa permite al usuario elegir entre al menos tres personajes. | 2 puntos
- Las estadísticas de los personajes (puntos de vida, ataque, defensa, etc.) se almacenan correctamente. | 4 puntos
- Se implementa una estructura de control (if/else o switch) para manejar la elección del personaje. | 4 puntos

Apartado 2 (0.5 punto):

- El enemigo se genera aleatoriamente con estadísticas comparables a las del personaje elegido. | 5 puntos
- Las estadísticas del enemigo (puntos de vida, ataque, defensa, etc.) se almacenan correctamente. | 5 puntos

Apartado 3 (4 puntos):

- Se inicia una batalla por turnos entre el jugador y el enemigo. | 2.5 puntos
- En cada turno, el jugador puede elegir entre al menos dos acciones (atacar, defender, etc.). | 2.5 puntos
- Se implementa un sistema de combate que tiene en cuenta las estadísticas de los personajes y enemigos. | 2.5 puntos
- Los puntos de vida de los personajes se actualizan correctamente después de cada turno. | 2.5 puntos

Apartado 4 (0.5 puntos):

- La batalla continúa hasta que uno de los dos, el jugador o el enemigo, quede sin puntos de vida. | 5 puntos
- Se muestra un mensaje que indica quién ganó la batalla. | 5 puntos

Apartado 5 (0.5 puntos):

- Todas las interacciones con el usuario siguen una estructura clara y organizada. | 5 puntos
- El programa proporciona una interfaz de usuario que permite al usuario realizar acciones como seleccionar personajes y ver el estado de la batalla. | 5 puntos

Apartado 6 (0.5 puntos):

- Se generan imágenes con asteriscos para cada tipo de personaje en la selección y durante la batalla. | 5 puntos
- Se muestra una barra de vida que refleja el estado de vida del personaje y del enemigo. | 5 puntos

Apartado 7 (1 punto):

- El enemigo sigue los patrones especificados en las dos fases (Fase 1 y Fase 2) correctamente.

Funcionalidad General (1 punto):

- El código está bien estructurado y sigue buenas prácticas de programación en Java. |2. 5 puntos
- Se utilizan comentarios adecuadamente para explicar el código. | 7.5 puntos