Tarea 1 - Heart Stone

Profesor: Alexandre Bergel
Auxiliar: Javier Espinoza, Eduardo Riveros

En esta tarea se le pedirá que implemente un modelo de batallas entre cartas. Su aplicación debe estar debidamente testeada y documentada y debe utilizar buenas prácticas de diseño. Especificaciones serán dadas más adelante.

Requisitos

Una carta tiene un nombre, un número que indica sus puntos de vida totales, un número a modo de contador de daño recibido, y un número que indica cuánto daño hace al atacar (considere que solo tiene un ataque o acción). Considere que existen solo 9 tipos de cartas: asesino (Assassin), druida (Druid), curador (Healer), cazador (Hunter), caballero (Knight), mago (Mage), paladin (Paladin), shaman (Shaman), y brujo (Warlock).

Una carta debe ser capaz de hacer una acción sobre otra:

- Assassin, Warlock, Knight, Mage y Hunter deben atacar a una carta normalmente.
- Healer debe restaurar el daño de una carta (no puede resucitar), pero usando la mitad de sus puntos de ataque en vez de todo..
- Druid aumenta los puntos de ataque de una carta usando todos sus puntos de ataque, pero la daña por la mitad de sus puntos de ataque del druida.
- Paladin debe incrementar el ataque y restaurar el daño de una carta, pero solo un tercio del ataque del Paladín para ambas acciones.
- Shaman disminuye el ataque y daña una carta usando un tercio del ataque del Shaman para ambas acciones.

Sin embargo, antes de realizar una acción, se deben considerar las interacciones especiales entre cartas, que son:

- **Assassin**: Recibe el doble de daño por Warlock, y recibe la bonificación de aumento de puntos de ataque del Druida sin recibir daño, pero solamente aumenta la mitad de los puntos de ataque del druida, a diferencia de cuando se aumenta la cantidad completa.
- **Druid**: Recibe el doble de los efectos del Shaman y del Paladin.
- **Healer**: Recibe doble de daño por parte de Assassin, y doble de los efectos del Paladin.

- **Hunter**: Solo recibe la porción del daño del Druid, y recibe el doble de ataque por parte de Mage.
- **Knight**: Recibe doble de ataque por Hunter, pero $\frac{1}{2}$ de daño por Assassin.
- **Mage**: Recibe el doble de daño por Assassin, pero $\frac{1}{2}$ por un Warlock.
- **Paladin**: Recibe la mitad del daño por un Knight, pero el doble de daño por un Mage.
- **Shaman**: Recibe el doble de daño por un Hunter.
- **Warlock**: El Healer, en vez de curarlo, lo ataca usando todo su ataque, y recibe el efecto doble del Shaman.

Tenga presente que si el contador de daño de una carta iguala o supera sus puntos de vida, la carta queda fuera de competencia (y no puede desarrollar su acción).

Requerimientos Adicionales

Además de una implementación basada en las buenas prácticas y patrones de diseño vistos en clases, usted además debe considerar:

- **Cobertura:** Cree los tests unitarios, en JUnit, que sean necesarios para alcanzar un 90% de coverage por paquete. Estos tests deben estar en el paquete test de su proyecto.
- **Javadoc:** Cada clase pública y cada método público deben estar debidamente documentados siguiendo las convenciones de Javadoc.
- **Resumen:** Debe entregar un archivo pdf que contenga su nombre y un diagrama UML que muestre las clases y métodos que usted definió en su proyecto.
- **Coding Style:** El código y documentación deben seguir las convenciones de Google (Click en Google).
- Git: Debe subir su código a GitHub.

Todos estos ítemes serán considerados para la nota final.

Evaluación

- **Código fuente (3.5 puntos):** Se analizará que su código provea la funcionalidad descrita y esté implementada utilizando un buen diseño.
- **Tests (1 punto):** Sus casos de prueba deben crear diversos escenarios y contar con un coverage de 90% por paquete.

- **Javadoc (1 punto):** Cada clase y método público debe ser documentado. Se descontará por cada falta.
- **Resumen (0.5 puntos):** El resumen mencionado en la sección anterior. De no enviarlo, se considerará que no utilizó un buen diseño y será penalizado. Debe incluir su link de GitHub en el resumen.

Entrega

Usted debe subir a U-Cursos o bien el proyecto de Eclipse realizado o bien las carpetas src y test de su proyecto. El plazo de entrega es hasta el 3 de septiembre a las 23:59 hrs. No se aceptarán peticiones de extensión del plazo.