



## Trabajo Final: Gimnasio Pokémon



Se desea crear un simulador de un Gimnasio Pokémon donde se enfrentan en un torneo varios Entrenadores con sus Pokémon. El torneo tendrá un número fijo de entrenadores y luego de varias rondas se definirá al ganador (definir sistema de rondas y eliminación) .

Cada entrenador posee una cantidad de pokemones, y en cada ronda presentará uno de ellos para que se enfrente a duelo con el pokemón de otro entrenador.




- Cada Entrenador tiene un Nombre y una lista de Pokémones
- Cada Pokémon tiene un nombre, puntos de experiencia, un nivel de escudo, nivel de vitalidad, y nivel de fuerza de ataque.

Una vez en la arena, se debe decidir que pokemón comienza a atacar, y luego el otro realiza el contraaunque.

Cada ataque consta de la siguiente secuencia:

1. Golpe inicial
2. Recarga (solo algunos pokemones pueden hacerlo)
3. Golpe final

Existen varias clases de Pokemones:

	Tipo	Vitalidad	Escudo	Fuerza
	Agua	500	100	120
	Hielo	500	120	100
	Fuego	530	200	80
	Etc (mínimo 5 tipos)	Definirlo	Definirlo	Definirlo

El modo de atacar y absorber el daño será el siguiente:

Tipo	Golpe Inicial	Recarga	Golpe Final	Recibe daño
Fuego	Todos los tipos realizan el mismo ataque inicial, infligiendo al adversario un daño igual a su fuerza de ataque, luego su fuerza de ataque se reduce a la mitad (cansancio)	Incrementa un 10% la fuerza y la vitalidad.	Provoca al adversario un daño igual a su fuerza mas un 25% y luego la fuerza se agota por completo (queda en cero)	El escudo y la vitalidad absorben la mitad del daño cada uno (decrementándose)
Agua			Provoca al adversario un daño igual a su fuerza y luego su fuerza se reduce a la mitad	El escudo absorbe todo el daño, solo cuando éste se agota comienza a decrementarse la vitalidad
Hielo		Si el Pokemon posee capacidad de " <i>gran recarga</i> " su fuerza tomará un valor de 400 pero no recargará su vitalidad, si no posee dicha capacidad, tendrá el comportamiento habitual.	Provoca al adversario un daño igual a su fuerza menos un 10% pero conserva la misma fuerza	El escudo absorbe el 75% del daño y la vitalidad el otro 25%
Otros tipos		Definirlo	Definirlo	Definirlo

Para modelar esta funcionalidad aplicar el patrón correspondiente.

Una vez terminada la ronda se define al ganador mediante un sistema de puntajes que aplicará un cálculo sobre el estado final de la fuerza, el escudo y la vitalidad (definirlo).

Al ganar una batalla el pokemón gana 3 puntos de experiencia, si pierde la batalla obtendrá un punto. Por su parte, los Entrenadores podrán recibir algún tipo de premio / castigo, (El pokemon perdedor quedará en poder del entrenador Ganador? Habrá créditos o algún otro tipo de premio?, definirlo)

Cuando un entrenador presenta a su pokemón para la batalla podrá opcionalmente presentar una carta hechizo que repercutirá en forma negativa en los atributos del pokemón de su adversario. Existen varios tipos de hechizos **Niebla**, **Viento** y **Tormenta**. Cada uno de estos hechizos afectará de forma diferente al Pokémon oponente dependiendo de su tipo (definirlo). Aplicar DoubleDispatch.

Durante el torneo los entrenadores sólo podrán utilizar un número limitado de hechizos (definirlo) si se pretende exceder ese límite se deberá administrar el error de la forma que considere conveniente (Excepciones)

Tanto los pokemones como los entrenadores son clasificables. La categoría de un pokemón está dada por su nivel de experiencia, y la de cada entrenador, por la sumatoria de las categorías de sus pokemones (aplicar interfaces).

Debe tenerse en cuenta que algunos tipos de pokemón deberán ser siempre clonables y otros no.

Se debe poder realizar, cuando sea posible, la clonación de un Entrenador.

## **Funcionalidades requeridas**

El sistema debe poder realizar la gestión de todos los datos del mundo del problema. El sistema debe poder generar un reporte de todos los enfrentamientos.

## **Consideraciones generales**

Deben documentarse los métodos que considere necesario aclarando las precondiciones establecidas y la lista de excepciones que podría lanzar.

Debe determinar en qué casos conviene lanzar una excepción, establecer precondiciones, aplicar patrones de diseño.

Debe tener en cuenta un diseño mínimo pero completo de cada clase, con las características y comportamiento necesarios.

El sistema debe entregarse funcionando, libre de errores de compilación. Debe presentar una clase principal con el método `main()` que comience la ejecución del programa. Dentro del `main()` deben presentarse los casos posibles que se consideren necesarios para comprobar la funcionalidad del sistema.

Puede incorporar funcionalidades extra si lo considera beneficioso al sistema.

## Segunda Parte (entregado el 26 de mayo de 2020)

Nuevas funcionalidades:

- Se deberá incorporar una interfaz gráfica (unificada o varias ventanas) para poder gestionar el sistema. Para ello se deberá aplicar el patrón MVC.
- Se debe poder realizar el alta de los entrenadores y de sus correspondientes pokemones.
- Para agilizar el desarrollo del torneo se podrán realizar varios enfrentamientos en forma simultánea (conurrencia). Sin embargo, en el torneo habrá una cantidad limitada (definirlo) de arenas (lugar donde se desarrolla la batalla) de manera que en una misma arena no puede más de un enfrentamiento a la vez (recurso compartido)
- Las acciones desarrolladas en las arenas deberá verse reflejada en la interfaz gráfica. Aplicar el patrón Observer / Observable.
- Cada arena pasará por varios estados, como mínimo deberán figurar
  - **Preliminar:** Se presentan los entrenadores y los pokemones correspondientes
  - **Batalla:** Durante este período se llevarán a cabo las batallas en si misma, lanzamiento de golpes, hechizos, etc.
  - **Definición:** Se define el ganador y se informa los resultados finales, premios, eliminaciones etc.

Opcionalmente se pueden agregar más estados, como por ejemplo **limpieza** (luego de la finalización antes de la nueva Preparación) o **CierreDefinitivo** (si ya no se llevarán a cabo más batallas en la arena). Aplicar el patrón State.

- Se deberá poder persistir el sistema con sus entrenadores, pokemones y estado del torneo.
  - Al finalizar cada etapa se debe persistir
  - Al iniciar la jornada se debe recuperar

Se sugiere separar el torneo en etapas para implementar esta funcionalidad, por ejemplo:

- Etapa 1: Alta de Entrenadores
- Etapa 2: Alta de Pokemones
- Etapa 3: Fase de grupos
- Etapa 4: Octavos de Final
- Etapa 5: Cuartos de Final
- Etc.