Modelos

Entrenador

- id -> int
- nombre-> String
- apellidos-> String
- nacimiento-> LocalDate
- tiposPreferidos -> List<TipoPokemon>
- equipoPokemon -> List<Pokemon>

Pokemon

- numPokedex-> int
- generacion-> Integer
- nombre-> String
- nivel -> Integer
- tipos -> List<TipoPokemon>
- stats -> Stats
- lineaEvolutiva -> List<LineaEvolutiva>
- movimientos -> List<Movimiento>

LineaEvolutiva

- pokemon-> Pokemon
- nivelParaEvolucionar -> Integer
- orden-> Integer

Stats

- id-> int
- ps -> Integer
- at-> Integer
- df-> Integer
- spa-> Integer
- spdf-> Integer
- spd -> Integer

Movimiento

- id-> int
- nombre ->String
- tipo -> TipoPokemon
- tipoAtaque -> TipoAtaque
- potencia-> Integer

TipoPokemon (Enum)

FUEGO, AGUA, PLANTA, ELECTRICO, NORMAL, LUCHA , VOLADOR, PSÍQUICO, SINIESTRO, BICHO, FANTASMA, HIELO, ACERO, DRAGÓN, HADA,ROCA,TIERRA;

TipoAtaque (Enum)

FÍSICO, ESPECIAL ;

Examen 2ª Evaluación Java

Profesor: Luis Javier López López

EJERCICIO 1 - (3 puntos)

1. (1,5 punto): Realiza un método en UtilidadesPokemon:
    a. public List<Pokemon> obtenerPokemonConTipos(List<Pokemon> pokemons, List<TipoPokemon> tipos)
        i. El método devuelve la lista de pokémon de los pasados como parámetro que cumplan la condición de que en sus tipos, contenga alguno de los tipos que se pasa como parámetro.

2. (1,5 punto): Realiza un método en UtilidadesPokemon:
    a. public Map<TipoPokemon, List<Pokemon>> obtenerPokemonPurosPorTipo(List<Pokemon> pokemons)
        i. Que devuelve un mapa con los pokémon agrupados por TipoPokemon siempre y cuando su lista de tipos contenga un único tipo.

EJERCICIO 3 - (2 puntos)

3. (2 punto): Realiza un método en UtilidadesPokemon:
    a. public List<Pokemon> leerPokemonConAtaques()
        i. Lee los pokémon del csv “pokemon.csv”
        ii. Lee los movimientos del csv “movimientos.csv”
        iii. Asigna los movimientos a los pokemon
        iv. Devuelve los pokemon.

EJERCICIO 4 - (5 puntos)

Realiza los siguientes métodos en Utilidades Combate:

4. (2,5 puntos): public Map<Entrenador, Integer> repartirPokemon(List<Entrenador> entrenadores, List<Pokemon> pokemon)
    a. El método reparte los pokémon por orden y equitativamente entre la lista de entrenadores, teniendo en cuenta que ambas listas contienen un número par de elementos, y que siempre el número de pokemons es divisible entre el número de entrenadores.
    b. Una vez repartidos los pokémon se comprueban cuantos tipos de entre los tipos favoritos del entrenador tienen los pokémon que se le han repartido, y se suma un punto por cada tipo que coincida.
    c. Al final devuelve el mapa de jugadores y puntos totales del reparto.

5. (1 puntos):public void subirAlNivel(Pokemon pokemon, Integer nivel)
    a. El método setea el nivel del pokémon al nivel pasado y aumenta las stats de la siguiente manera:
        i. A cada stat(ps,at,def,spa,spdf,spd) le suma 2* nivel. Ejemplo:
        ii. ps = ps + (2 * nivel)

6. (1,5 puntos): public boolean puedeEvolucionar(Pokemon pokemon)
    a. El método comprueba si el pokémon pasado como parámetro puede evolucionar para ello:
        i. Mira el nivel del pokémon que se le pasa y tiene que hacer dos comprobaciones:
            1. Que en la línea evolutiva del pokémon haya algún pokémon por encima en orden.
            2. Que el alguno de los pokémon que está por arriba en índice, tenga de nivel requerido uno menor o igual que el del pokémon actual.