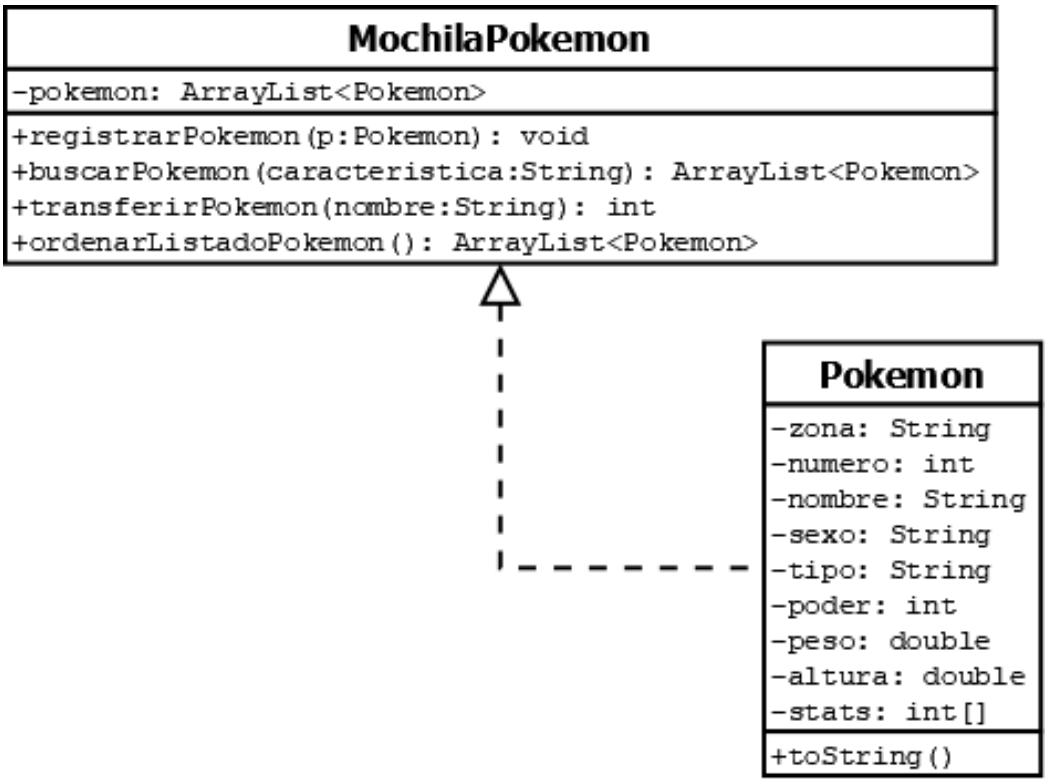
Ejercicio: POO – Colecciones ArrayList

Juguemos un poquito a programar un juego o algo que se parece a Pokémon Go.

A continuación, se entrega el modelo de clases que representa en términos generales lo que se necesita codificar.



CODIFICANDO LAS CLASES

Se solicita que para cada clase se definan constructor, accesadores y mutadores según sea necesario. Junto a ello se necesita que modifique el método toString de cada clase para que entregue la información cada clase de una manera clara y sencilla.

Respecto a las clases y sus métodos CUSTOM, analicemos la clase MochilaPokemon que posee los siguientes métodos:

registrarPokemon: este método permitirá agregar un Pokémon a la mochila. Simplemente debe agregarse a la mochila. No se debe evaluar si existe dicho Pokémon en la mochila, en pocas palabras pueden existir más de un Pokémon con los mismos datos.

buscarPokemon: este método recibirá un parámetro a través del cual deberá buscar en todas las propiedades, por ejemplo, si el método recibe Bulbasaur, el método deberá retornar un ArrayList con todos los Pokémon que tengan dicho nombre. Otro ejemplo es si el parámetro entregado es “Tierra”, el método retornará un ArrayList con todos los Pokémon de tipo Tierra. Los posibles parámetros serán asociados a la zona, nombre, sexo o tipo. El método deberá buscar en todas las propiedades indicadas.

transferirPokemon: este método convertirá en caramelos (triturar bien triturados) todos los Pokémon que tengan el nombre ingresado como parámetro. Debe retornar la cantidad de Pokémon que fueron triturados.

ordenarListadoPokemon: este método nos permitirá ordenar según los distintos tipos de criterios o propiedades, considerando que se puede ordenar por número, nombre, sexo, tipo y poder. El método solo debe ordenar en primera instancia por Número, en caso de igualdad, se deben ordenar por Poder. De persistir la igualdad por Sexo (Femenino luego Masculino), peso, altura, promedio de las stats. Si aún existen igualdades, deberá darse dos vueltas sobre su propio eje y rezar 10 ave maría.

El método toString de la clase Pokémon deberá entregar la siguiente información:

Número: 1 | Nombre: Bulbasaur | Sexo: Masculino | Tipo: Planta / Veneno | Poder: 860 | Peso: 5.99 kg. |

Altura: 0.6m | Estadísticas: A15 – D15 – S15

Para probar el funcionamiento de su código se requiere que en la clase Main realice las siguientes operaciones

1. Crear una MochilaPokemon.
2. Crear unos 10 Pokémon de distintos tipos
3. Registrarlos en la mochila.
4. Ejecutar los métodos que permiten buscar, transferir y ordenar. Se recomienda que cada vez que ejecute un método imprima los elementos de la Mochila para ver los resultados.
5. Recuerde documentar todo su código mediante el uso de JAVADOC y general la documentación del proyecto.

Utilice la nomenclatura del lenguaje para la definición de clases, atributos y encapsule todo su código.