**ESCUELA SUPERIOR POLITÉCNICA**

**PROGRAMACIÓN ORIENTADA A OBJETOS**

**Objetivo:**

* Diseñar una clase basándose en las características propias de los objetos para la asociación con el mundo real.
* Aplicar los conceptos de encapsulamiento para la protección de las propiedades en un programa.

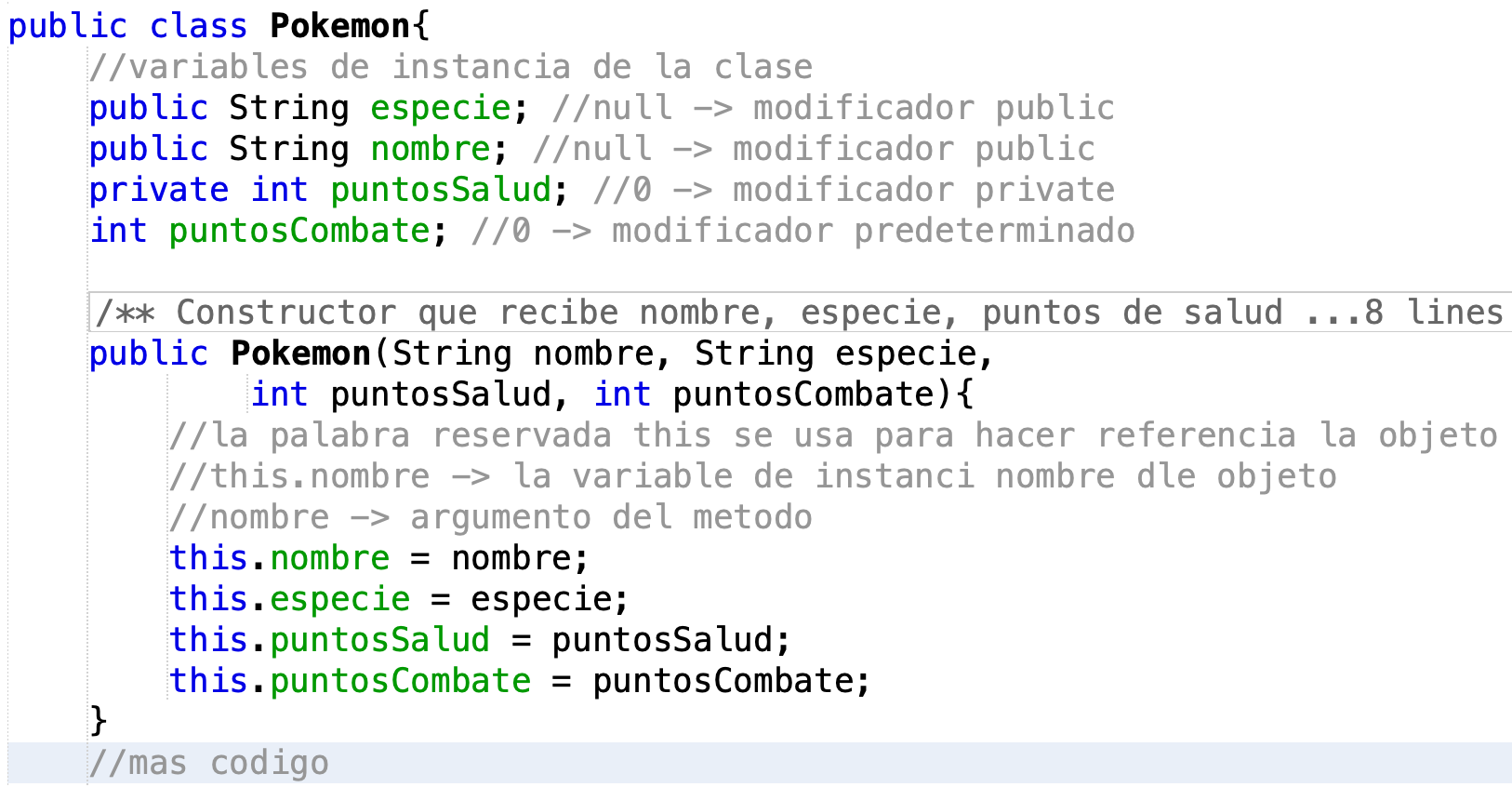
**Instrucciones:**

* **Descargue el proyecto AppPokemonAlimento y descomprimalo**
* **Abra el proyecto en el IDE**
* **Realice las tareas indicadas en la Descripción**

**Descripción:**

**PARTE 1 – Probar modificadores de acceso**

* Revise la clase Pokemon.java

****

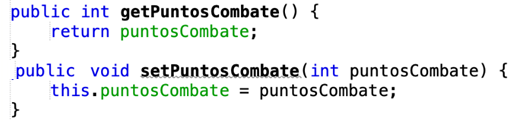
* + Las variables de instancia especie y nombre tienen modificador de acceso public -> son accesibles desde cualquier clase.
  + La variable de instancia puntosSalud tiene modificador de acceso **private**, solo puede ser accedido dentro de la clase.
  + La variable de instancia puntosCombate no declara modificador de acceso por lo tanto tiene modificador de acceso **default**, puede ser accedido desde dentro de la clase y otras clases en el mismo paquete.
* Ejecute la clase **TestAccesoMismoPaquete.java y observe la salida.** Descomente las líneas con comentarios y observe que pasa.
* Ejecute la clase **TestAccesoOtroPaquete.java y observe la salida. .** Descomente las líneas con comentarios y observe que pasa.

**PARTE 2 – Modifique la clase Pokemon para ocultar su información**

En la clase **Pokemon** realice lo siguiente:

* Modifique las variables de instancia de la clase para que sean privadas.
* Cree los métodos get de las variables de instancia.
* Cree el método set de las variables de instancia **nombre**, **puntosSalud** y **puntosCombate.**

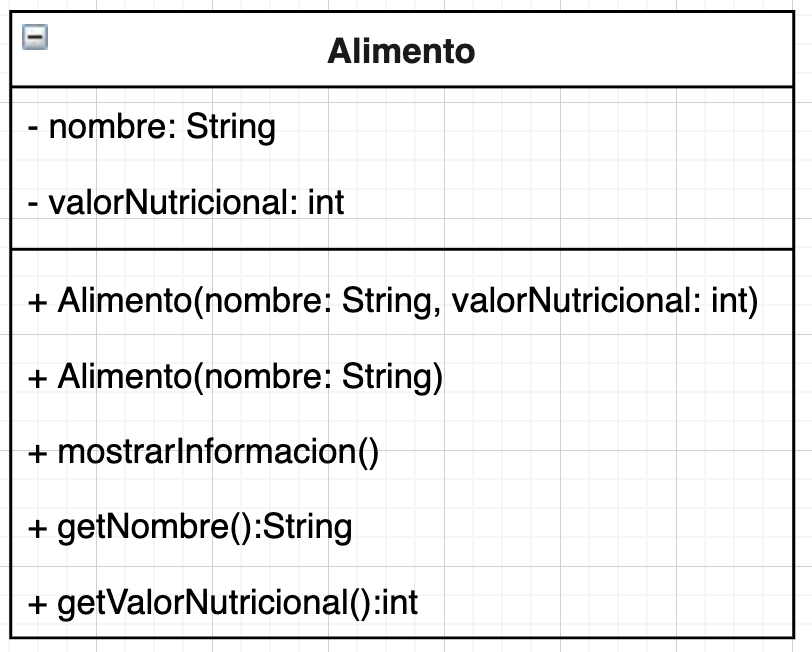
Ejemplo del get y ser de la variable puntosCombate



**PARTE 3**

En el paquete espol.poo.**modelo** cree la clase **Alimento**.

Esta clase representa la comida que se le da a un pokemon.



* La clase tiene dos variables de instancia que almacenan el nombre del alimento y su valor nutricional.
* Estas variables de instancia deben tener modificador de acceso privado.
* La clase debe tener dos constructores públicos
  + Uno que reciba un nombre y un valor nutricional. Inicialice las variables de instancia del objeto con estos valores.
  + Uno que reciba el nombre del Alimento. Inicialice la variable nombre con el valor recibido, y fije el valor nutricional a 10.
* Cree los métodos get de las variables de instancia.
* La clase debe tener un método público mostrarAlimento que imprima en pantalla el nombre del alimento y su valor nutricional

**PARTE 4 – Implemente método comer.**

* En la clase **Pokemon** defina el método **comer** que reciba un objeto de tipo Alimento. Este método incrementa los puntos de salud del pokemon en el valor nutricional del alimento.

Graphical user interface, text, application

Description automatically generated

**PARTE 4**

* En la clase **AppPokemonAlimento.java** realice lo que indica el método main.

**PARTE 5 – CONTESTE LAS SIGUIENTES PREGUNTAS**

* **¿Qué implicación tendría en el programa que las variables de instancia de la clase Alimento tengan modificador de acceso privado y qué no se hayan creado método set para estas?**

Esto implica que los objetos de la clase Alimento no pueden ser modificadas una vez inicializado el objeto, sin embargo si puede se puede leer los atributos gracias a los getters de la clase.

* **Para cada una de las sentencias a continuación trate de predecir cuál será la salida en pantalla. Si la ejecución de la sentencia produce un error indique cuál sería y por qué. Luego compruebe sus respuestas ejecutando el código en el método main.**

|  |  |  |
| --- | --- | --- |
| 1 | Alimento a = new Alimento();  a.nombre="fresa"; | **Mostraría un error ya que la clase Alimento ya tiene definido constructores y no se especifica la clase al cre.** |
| 2 | Pokemon p1 = new Pokemon("Pikachu","Sparky",300,1200);  Pokemon p2 = p1;  p1.setNombre("Gigante")  System.out.println(p1.getNombre());  System.out.println(p2.getNombre()); | **Mostraría un error de sintaxis porque falta un ; al final de la tercera línea, ignorando aquello, al ejecutar mostraría “Gigante” dos veces ya que la segunda línea no crea una copia.** |
| 3 | Pokemon p3 = new Pokemon("Charizard","MyDragon");  System.out.println(p3.mostrarInformacion()); | **Mostraría un error ya que el método mostrarInformacion() no retorna un valor a imprimir.** |
| 4 | Pokemon p4 = new Pokemon("Chikorita","Chiko");  p4.setPuntosSalud(100);  System.out.println(p4.mostrarInformacion()); | **Mostraría un error ya que en la segunda línea no se especifica la clase y el método mostrarInformacion() no retorna un valor a imprimir.** |