Tarea NO Evaluable.

1. **Diseño de videojuegos.**

Estás desarrollando un videojuego de rol (RPG) en el que los personajes pueden ser de distintas clases, como Guerrero, Mago y Arquero. También existen villanos.

Cada **personaje** comparte características comunes como nombre, nivel, puntos de vida (HP) y el arma. Por lo que deben heredar de una clase base llamada Personaje. Un personaje puede ser atacado o no dependiendo de quién le ataque y/o de quién y usando qué arma.

Incluye un método llamado getArma que devuelva:

- Cadena vacía para los villanos.
- Espada para el Caballero
- Flecha para el Arquero
- Hechizo para el Mago.

Considera que el método esAtacado estará sobrecargado. De manera que habrá 2 implementaciones distintas:

- esAtacado(Personaje atacante): boolean
- esAtacado(Personaje atacante, int distancia): boolean

Además, cada tipo de personaje es atacado de forma diferente. Si puede ser atacado devuelve cierto y si no falso.

- A los villanos les ataca todo el mundo y desde cualquier distancia
- Al arquero puedes ser atacado por el caballero si está a menos de 50 mts y el mago siempre.
- Al caballero puedes ser atacado por el arquero si está a más de 100 mts y el mago siempre.
- Al mago no le ataca nadie.

Considera

1. Dibuja el diagrama de clases UML.
2. Crea una clase base llamada Personaje con atributos nombre y arma.
3. Crea subclases Villano, Caballero, Arquero y Mago e implementa los métodos necesarios
4. Crea una clase llamada GestionaJuego que cree un personaje de cada tipo y nos diga si se atacan o no entre ellos con y sin distancia.Invoca también a los métodos que nos dicen qué arma usan

2. Planetas

Define una jerarquía de clases que permita almacenar datos sobre los planetas y satélites (lunas) que forman parte del sistema solar.

Algunos atributos que necesitaremos almacenar son:

- nombre
- masa del cuerpo.
- diámetro medio.
- período de rotación sobre su propio eje (horas)
- período de traslación alrededor del cuerpo que orbitan (horas)
- distancia media a ese cuerpo.
- etc.

Define las clases necesarias conteniendo:

- Constructores.
- Métodos para recuperar y almacenar atributos.
- Método para mostrar la información del objeto.
- Sobreescribe el método toString de forma que devuelva el nombre, la masa y el diámetro de cada astro. Si es un planeta la cadena comenzará por "Planeta:", si es un satélite: por "Satélite:" e igual con Astro.
- Sobreescribe el método el método equals para que devuelva cierto si se llaman igual, están a la misma distancia y tienen la misma masa

Define un método llamado muestraInformación que dado un objeto del sistema solar (astro, planeta o satélite), imprima toda la información que se dispone sobre el mismo (además de su lista de satélites si los tuviera).

Considera que puede haber astros que no sean ni planetas ni satélites.