1. Bad smell se encuentra en distintas partes en este código:

* En el primer método de la clase VideoJuego, hay un espacio entre la palabra personaje y ConMasDaño(), lo cual produce un error.
* Falta un corchete para cerrar el método imprimirInfo() y esto genera un error al compilar.
* Dentro del método imprimirInfo() hay código duplicado, ya que en vez de usar System.out.printLn se puede tener un método en cada subclase que retorne su descripción.
* double daño = p.getTipoHabilidad().calcularDaño(p.getDaño()) es una línea de código que presenta mucha confusión porque llama a muchos métodos a la vez. Lo que se puede hacer, es crear un método en la clase personaje que realice la lógica interna de cuánto daño hace.
* En el método imprimirInfo() se puede extraer partes en métodos más pequeños ya que chequea tipos aparte de mostrar descripciones.

1. Creemos que los malos olores detectados representan un problema para la mantenibilidad y escalabilidad del software ya que:

* Hacen que el código sea difícil de entender debido a que, cada vez que se agrega un nuevo tipo de Habilidad, hay que modificar el if para contemplar ese nuevo tipo y eso va a terminar generando código repetido porque va a haber muchos else if.
* Es más difícil encontrar bugs en el programa ya que no va a tener una organización o estructura definida.
* Para las próximas personas que interactúen con el código, les va a ser muy difícil entenderlo y modificarlo.
* El código mezcla responsabilidades (mostrar nombre, calcular danio, imprimir descripción). Esto hace que sea difícil de leer, mantener y extender.

1. El tipo de refactoring que se recomienda aplicar para corregir el mal olor es **extract method** ya que:

* Se puede extraer el código duplicado (por ejemplo, los printLn) en un método común dentro de cada clase correspondiente.
* Se puede dividir las múltiples responsabilidades que tiene un método en métodos más pequeños y comprensibles.