4.A. Refactorización

2. Patrones de refactorización más habituales.

Algunos de los **patrones más habituales de refactorización**, que vienen ya integrados en la mayoría de los entornos de desarrollos, son los siguientes:

- **Renombrar**. Cambiar el nombre de un paquete, clase, método o campo, por un nombre más significativo.
- **Encapsular campos**. Crear métodos de asignación y de consulta (**getters y setters**) para los campos de la clase, que permitan un control sobre el acceso de estos campos, debiendo hacerse siempre mediante el uso de estos métodos.
- **Sustituir bloques de código por un método**. En ocasiones se observa que un bloque de código puede constituir el cuerpo de un método, dado que implementa una función por si mismo o aparece repetido en múltiples sitios. De esta forma, cada vez que queramos acceder a ese bloque de código, bastaría con invocar al método.
- **Modificar la extensión del código**. Hacer un código más extenso si se gana en claridad o menos extenso sólo si con eso se gana eficiencia.
- **Reorganizar código condicional complejo.** Patrón aplicable cuando existen varios if o condiciones anidadas o complejas.
- **Crear código común** (en una clase o método) para evitar el código repetido.
- **Mover la clase**. Mover una clase de un paquete a otro, o de un proyecto a otro. Esto implica la actualización en todo el código fuente de las referencias a la clase en su nueva localización.
- **Borrado seguro**. Garantizar que cuando un elemento del código ya no es necesario, se borran todas la referencias a él que había en cualquier parte del proyecto.
- **Cambiar los parámetros del método**. Permite añadir/modificar/eliminar los parámetros en un método y cambiar los modificadores de acceso.
- **Extraer la interfaz**. Crea un nueva interfaz de los métodos public non-static seleccionados en una clase o interfaz.