**Refactoring**

**Problema**: el visitor en si no respetaba el principio SOLID open/closed. Cada vez que agregara una prenda a mi proyecto, esto implicaba tener que modificar tanto la interfaz como la implementación concreta del visitor.

**Solución**: el visitor originalmente tenía un método específico para cada prenda (por esto surge el problema).

En la solución propongo que se supriman todos estos métodos “específicos” de las prendas y se trabaje con lo más general de estas que sería la clase “Prenda”.

Por ende el visitor tendrá un solo método donde visita a una “Prenda” y luego cada implementación específica de este decidirá qué hacer con esta.

Además anteriormente el proceso de armar el puntaje de la prenda lo realizaba el visitor, lo que propongo es considerar el “proceso” de seteado de puntaje como un proceso específico de la prenda, donde cada una a partir de una serie de datos que le brindara el visitor decidirá que puntaje se estaría estableciendo a sí misma.

Por lo tanto ahora esos “métodos específicos” que poseía la interfaz del visitor se delegan a cada prenda específica y se suprimen de la interfaz.

**Conclusión**:

Antes agregar una prenda a mi proyecto implicaba una modificación tanto de la interfaz como de las implementaciones (si es que hay más de una en el futuro) del visitor.

Ahora agregar una prenda implica que esta prenda defina su proceso de “punteado” sin necesidad de modificar las interfaces.

**La solución se encuentra ya implementada en el proyecto.**