Ejercicio 2 - Refactoring

OO2 -1er recuperatorio- 29/06/2024

Para el siguiente código, realice las siguientes tareas:

(i) indique que mal olor presenta

(ii) indique el refactoring que lo corrige

(iii) aplique el refactoring (modifique el código)

Si vuelve a encontrar un mal olor, retorne al pasc (i).

Nota: Haga los cambios que considere necesarios.

```
1. public class Pago {
3.    private List<Producto> productos;
4.    private String tipo;
5.    private static final double ADICIONAL_TARJETA = 1000.0;
6.    private static final double DESCUENTO_EFECTIVO = 2000.0;

7.    public Pago(String tipo, List<Producto> productos) {
8.       this.productos = productos;
9.       this.tipo = tipo;
10.   }

12.   public double calcularMontoFinal() {
13.      double total = 0.0;
14.      if (this.tipo == "EFECTIVO"){
15.         for (Producto producto: this.productos){
16.            total = total + producto.getPrecio() + (producto.getPrecio() * producto.getIVA());
17.         }
18.         if (total > 100000){
19.            total = total - DESCUENTO_EFECTIVO;
20.         }
21.      }
22.      else if (this.tipo == "TARJETA"){
23.         for (Producto producto: this.productos){
24.            total = total + producto.getPrecio() + (producto.getPrecio() * producto.getIVA())
25.         }
26.         total = total + ADICIONAL_TARJETA;
27.      }
28.      return total;
29.   }
30. }
```

```
1.  public class Producto {
2.     private double precio;
3.     private double IVA;
4.
5.     public Producto(double precio, double IVA) {
6.        this.precio = precio;
7.        this.IVA = IVA;
8.     }
9.
10.    public double getPrecio() {
11.       return this.precio;
12.    }
```

```
13.   public double getIVA() {
14.      return this.IVA;
15.   }
16. }
```

-
-
-
-

```
total = productos.stream()
                 .mapToDouble(producto -> producto.getPrecio() +
                     producto.getPrecio() * producto.getIva())
                 .sum();
```

```
// Dentro de la clase pago
public double calcularTotal() {
    return this.precio + (this.precio * this.IVA);
}

// pipeline correcido
// .mapToDouble(Producto::calcularTotal).sum();
```

```java
public class Pago {
    private List<Producto> productos;
    private IMetodoPago metodoPago;

    public Pago(IMetodoPago metodoPago, List<Producto> productos) {
        this.productos = productos;
        this.metodoPago = metodoPago;
    }

    public void setMetodoPago(IMetodoPago metodoPago) {
        this.metodoPago = metodoPago;
    }

    public double calcularMontoFinal() {
        double total = 0.0;
        total = productos.stream()
                    .mapToDouble(Producto::calcularTotal)
                    .sum();
        return metodoPago.aplicarDescuento(total);
    }


}
```

```java
public class Producto {
    private double precio;
    private double IVA;

    public Producto(double precio, double IVA) {
        this.precio = precio;
        this.IVA = IVA;
    }

    public double calcularTotal() {
        return this.precio + (this.precio * this.IVA);
    }
}


public interface IMetodoPago {
    public double aplicarDescuento(double total);
}


public class Efectivo extends IMetodoPago {

    public double aplicarDescuento(double total) {
        if (total > 100000) {
            return total - 2000;
        }
        return total;
    }

}

public class Tarjeta extends IMetodoPago {
    public double aplicarDescuento(double total) {
        return total + 1000
    }
}
```