Ejercicio 4

Codigo original:

```
public class Pedido

{
private Cliente cliente;
private List<Producto> productos;
private String formaPago;

public Pedido(Cliente cliente, List<Producto> productos, String formaPago) {
if (!"efectivo".equals(formaPago)
&& !"6 cuotas".equals(formaPago)
&& !"12 cuotas".equals(formaPago)) {
throw new Error("Forma de pago incorrecta");
}
this.cliente = cliente;
this.productos = productos;
this.formaPago = formaPago;
}

public double getCostoTotal() {
double costoProductos = 0;
for (Producto producto : this.productos) {
costoProductos += producto.getPrecio();
}
double extraFormaPago = 0;

if ("efectivo".equals(this.formaPago)) {
extraFormaPago = 0;
} else if ("6 cuotas".equals(this.formaPago)) {
extraFormaPago = costoProductos * 0.2;
} else if ("12 cuotas".equals(this.formaPago)) {
extraFormaPago = costoProductos * 0.5;
}
int añosDesdeFechaAlta = Period.between(this.cliente.getFechaAlta(),
LocalDate.now()).getYears();
// Aplicar descuento del 10% si el cliente tiene más de 5 años de antigüedad
if (añosDesdeFechaAlta > 5) {
return (costoProductos + extraFormaPago) * 0.9;
}
```

```
return costoProductos + extraFormaPago;
}
}
```

Aplicando refactoring SOLO a las lineas donde nos dicen:

- Replace Loop with Pipeline (líneas 16 a 19)

```
// refactor realizado
public double getCostoTotal() {
double costoProductos = this.productos.stream()
.mapToDouble(p -> p.getPrecio())
.sum();
```

-
- Replace Conditional with Polymorphism (líneas 21 a 27)

```
// Refactor 2 realizado : Se calcula el extra de manera polimorfica.
double extraFormaPagoPolimorfica =
this.formaPago.calcularExtraPago(costoProductos);
```

- Extract method y move method (línea 28)

```
package ar.edu.info.unlp.ejercicioDemo;
import java.time.LocalDate;
import java.time.Period;

public class Cliente {
private LocalDate fechaAlta;
public Cliente ()
{
this.fechaAlta = LocalDate.now();
}

public LocalDate getFechaAlta()
{
return this.fechaAlta;
}

// Refactoring 3
public int calcularAnhosFechaAlta()
{
return Period.between(fechaAlta, LocalDate.now()).getYears();
```

```
}
}
```

- Extract method y replace temp with query (líneas 28 a 33)

```
public double aplicarDescuento (double costoProducto, double extraFormaPago)

{

if (cliente.calcularAnhosFechaAlta() > 5)
    return (costoProducto + extraFormaPago)* 0.9;
return costoProducto + extraFormaPago;
```

Codigo FINAL

Clase Pedido:

```
package ar.edu.info.unlp.ejercicioDemo;
import java.time.LocalDate;
import java.time.Period;
import java.util.List;

public class Pedido
{
private Cliente cliente;
private List<Producto> productos;
private FormaDePago formaPago;

public Pedido(Cliente cliente, List<Producto> productos, FormaDePago
formaPago) {
if (!"efectivo".equals(formaPago)
&& !"6 cuotas".equals(formaPago)
&& !"12 cuotas".equals(formaPago)) {
throw new Error("Forma de pago incorrecta");
}
this.cliente = cliente;
this.productos = productos;
this.formaPago = formaPago;
}



// Refactor 2 realizado
```

```
public double getCostoTotal() {
double costoProductos = this.productos.stream()/
.mapToDouble(p -> p.getPrecio())
.sum();

// Refactor 2 realizado : Se calcula el extra de manera polimorfica.

double extraFormaPago = this.formaPago.calcularExtraPago(costoProductos);
// Regactor 4
return aplicarDescuento(costoProductos, extraFormaPago);
}

public double aplicarDescuento (double costoProducto, double extraFormaPago)
{
if (cliente.calcularAnhosFechaAlta() > 5)
    return (costoProducto + extraFormaPago)* 0.9;
return costoProducto + extraFormaPago;
}

}
```

Clase Producto:

package ar.edu.info.unlp.ejercicioDemo;

```
import java.time.LocalDate;



public class Producto {

private double precio;

public Producto (double precio)

{

this.precio = precio;

}

public double getPrecio()

{
```

```
return this.precio;

}

}
```

Clase Cliente:

```
package ar.edu.info.unlp.ejercicioDemo;
import java.time.LocalDate;
import java.time.Period;

public class Cliente {
private LocalDate fechaAlta;

public Cliente ()

{

this.fechaAlta = LocalDate.now();

}

public LocalDate getFechaAlta()

{

return this.fechaAlta;

}

public int calcularAnhosFechaAlta()

{

return Period.between(fechaAlta, LocalDate.now()).getYears();

}

}
```

Interfaz FormaDePago

```java
package ar.edu.info.unlp.ejercicioDemo;

public interface FormaDePago {



public double calcularExtraPago(double costoProducto);

}
```

Clase Efectivo:

```java
package ar.edu.info.unlp.ejercicioDemo;

public class Efectivo implements FormaDePago{

@Override

public double calcularExtraPago(double costoProducto)
{
return costoProducto * 0;

}

}
```

Clase Cuotas_6

```java
package ar.edu.info.unlp.ejercicioDemo;


public class Cuotas_6 implements FormaDePago {


@Override

public double calcularExtraPago(double costoProducto)
{
return costoProducto * 0.2;
}

}
```

Clase Cuotas_12

```
package ar.edu.info.unlp.ejercicioDemo;

public class Cuotas_12 implements FormaDePago {
@Override

public double calcularExtraPago(double costoProducto)
{
return costoProducto * 0.5;
}

}
```