# Trabajo Práctico
# Patrón Strategy

1. Una empresa que se dedica a la comercialización de productos informáticos a través de internet ofrece a sus clientes la posibilidad de optar entre diferentes formas de envío de los productos. El cliente va almacenando productos en su carrito de compras y finalmente el sistema calcula el costo total incluyendo el envío. El costo total será la suma de precio de cada producto del carrito, más el envío que cada compañía ofrece su forma de cálculo específica.

Las posibilidades de envío que ofrece son a través de la empresa de

- **Colectivos Sur**: Si el destino es Capital Federal hay un costo fijo de 1000 pesos. Si el destino es gran buenos aires el monto fijo es de 1500 pesos. Cualquier otro destino el monto fijo es 3000 pesos. Además, si el peso total de los productos superan los 5kg (hasta 30kg), se le agrega un adicional de 500 pesos. Pasados los 30kg el adicional es de 2000 pesos.

- **Correo Argentino**: Si el destino es Capital Federal se cobra un monto fijo de 500 pesos. Cualquier otro destino, se cobra un fijo de 800, más un monto que sale de calcular 5$ multiplicado por la cantidad de kilómetros entre Capital Federal y el destino. Ésta distancia la brinda un servicio externo Web (http://distancia.ar?orgen=capital&destino=xxx). El sistema debe permitir al cliente optar por cualquier forma de envío e informarle el costo asociado a la opción elegida.

a) Aplicando el patrón Strategy diseñe una posible solución al problema planteado. Implemente en Java la solución propuesta y dos casos de test.

2. Implemente en Java una clase Persona que responda al mensaje fechaNacimiento(). Este mensaje devuelve un String con la fecha de nacimiento de la persona. La fecha de nacimiento puede ser:

- Corta: 3-06-1986
- Larga: 3 de Junio de 1986

Implemente utilizando el patrón Strategy. Implemente dos casos de test.

3. La siguiente clase Producto calcula el precio de un producto teniendo en cuenta impuestos, descuentos y envío. Luego se presenta un Main para mostrar cómo se utiliza. Se pide:

1. Refactorizar para remover los IFs sobre los tipos de producto aplicando el patrón Strategy (creando la jerarquía polimórfica con un CalculadorDePrecios, no sobre Producto. Producto delega en la estrategia de forma polimorfica).

2. Modifique el Main para que funcione de acuerdo al refactor realizado.

```
enum TipoProducto {
   LIBRO,
   ALIMENTO,
```

```
    MEDICINA,
    OTRO
}

class Producto {
    public TipoProducto tipo;
    public double precio;

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

    public double precioFinal() {
        double impuestos = 0;
        double descuentos = 0;
        boolean envioGratis = false;

        if (tipo == TipoProducto.LIBRO) {
            impuestos = 0.1;
            descuentos = 0.1;
            if (precio > 100) {
                envioGratis = true;
            }
        } else if (tipo == TipoProducto.ALIMENTO) {
            impuestos = 0.05;
            if (precio > 100) {
                descuentos = 0.15;
            }
            if (precio > 200) {
                envioGratis = true;
            }
        } else if (tipo == TipoProducto.MEDICINA) {
```

```
            impuestos = 0;
            if (precio > 50) {
                descuentos = 0.1;
            }
            if (precio > 100) {
                envioGratis = true;
            }
        } else {
            impuestos = 0.15;
            if (precio > 50) {
                descuentos = 0.05;
            }
            if (precio > 200) {
                envioGratis = true;
            }
        }

        double total = precio * (1 + impuestos) * (1 - descuentos);
        if (envioGratis) {
            total -= 10;
        }

        return total;
    }
}


public class Main {
    public static void main(String[] args) {
        var p1 = new Producto(TipoProducto.LIBRO, 30);
        var p2 = new Producto(TipoProducto.MEDICINA, 330);
        var p3 = new Producto(TipoProducto.ALIMENTO, 130);
        var p4 = new Producto(TipoProducto.OTRO, 130);
```

```
        System.out.println(p1.precioFinal());
        System.out.println(p2.precioFinal());
        System.out.println(p3.precioFinal());
        System.out.println(p4.precioFinal());
    }
}
```