✅

Ejercicio 13 CODIFICACION

13) Se desea implementar la lógica de un dispositivo POSNET que procesa pagos con tarjetas de crédito. Las tarjetas de crédito guardan el nombre de la entidad financiera a la que pertenecen (únicamente Visa, MasterCard o Maestro), el nombre de la entidad bancaria, el número de tarjeta, el saldo disponible y los datos del titular (nombre, apellido, fecha de nacimiento y domicilio). Cada vez que se cree una nueva tarjeta, deberán indicarse todos estos datos.

A la hora de abonar, el POSNET recibiría la tarjeta con la que desea hacerse el pago, junto con el monto que se desea abonar y la cantidad de cuotas (de 1 a 6).

Si el pago es en 1 cuota, no se genera ningún recargo, de lo contrario, el monto se incrementará en un 3% por cada cuota superior a 1. (Ejemplo: Pagar en 4 cuotas representará un 9% de incremento).

El POSNET debe chequear que la tarjeta tenga saldo suficiente para poder efectuar el pago junto con el recargo, si hubiese. En caso de éxito, debe generar y retornar (no mostrar) un ticket donde consten los siguientes datos:

- Nombre y apellido del cliente.
- Monto total a pagar.
- Monto de cada cuota.

Si la operación no tuvo éxito, se retornará `null`.

A) Desarrollar, en la clase `Posnet`, el método `efectuarPago()`, cuyos parámetros, lógica y valor de retorno deben deducirse según lo enunciado. Desarrollar también los métodos derivados que puedan surgir de él para conseguir el objetivo.

B) Desarrollar el método `main` del proyecto y generar las instancias necesarias para poder efectuar un pago de $10000 en 5 cuotas, usando una tarjeta de crédito con saldo disponible de $15000 (el resto de los datos, pueden inventarse a tu gusto).

Detalles del proyecto a destacar:

```
boolean esTarjetaValida = tarjeta != null;
```

```
private boolean datosValidos(TarjetaDeCredito tarjeta,double monto,int cant){
        boolean esTarjetaValida = tarjeta != null; // una expresion booleana puede ser guardada de esta manera
        boolean esMontoValido = monto > 0;
        boolean cantCuotasValidas = cant >= MIN_CANT_CUOTAS && cant <= MAX_CANT_CUOTAS;
        return esTarjetaValida && esMontoValido && cantCuotasValidas ;
    }
```

Clases del proyecto:

```
package ejercicio_13;
public class Principal {
    //PUNTO B
    public static void main(String[] args) {
        Posnet posnet = new Posnet(); // se genera un objeto posnet (para pedirle que efectue un pago)
        Persona p = new Persona("40545665","Pepe","Gomez","1112345678","pepe@fakemail.com"); // persona que será titular de la tarjeta de credito
        TarjetaDeCredito t = new TarjetaDeCredito("FakeBank","1234567890123456",15000,p,EntidadFinanciera.VISA);

        // mostrar la trajeta antes de efectuar pago
        System.out.println("Tarjeta antes del pago");
        System.out.println(t);

        // mostrar la trajeta tras pagar
        System.out.println("Ticket tras pagar...");
        Ticket ticketGenerado = posnet.efectuarPago(t,10000,5);
        System.out.println(ticketGenerado); // verificar si ese ticket es un ticket o null (si el pago no se pudo procesar)

        // mostrar la trajeta despues del pago
        System.out.println("Tarjeta despues del pago"); // se comprueba la modificacion del valor del saldo
        System.out.println(t);
    }
}
```

```
package ejercicio_13;
public class Persona {
    private String DNI;
    private String nombre;
    private String apellido;
    private String telefono;
    private String mail;

    public Persona(String DNI, String nombre, String apellido, String telefono, String mail) {
        this.DNI = DNI;
        this.nombre = nombre;
        this.apellido = apellido;
        this.telefono = telefono;
        this.mail = mail;
    }

    String nombreCompleto() {
        return nombre + " " + apellido;
    }
}
```

```
package ejercicio_13;
class TarjetaDeCredito {
    //atributos "primitivos"
    private String entidadBancaria;
    private String nroTarjeta;
```

```
    private double saldo;
    //atributos donde el tipo de dato es una clase
    private EntidadFinanciera entidadFinanciera;
    private Persona titular;

    public TarjetaDeCredito(String entidadBancaria, String nroTarjeta, double saldo, Persona titular, EntidadFinanciera entidadFinanciera) {
        this.entidadBancaria = entidadBancaria;
        this.nroTarjeta = nroTarjeta;
        this.saldo = saldo;
        this.titular = titular;
        this.entidadFinanciera = entidadFinanciera;
    }

    public boolean tieneSaldoDisponible(double monto) {
        return saldo >= monto;
    }

    public void descontar(double monto) {
        saldo = saldo - monto;
        // saldo -= monto;
    }

    public String nombreCompletoDeTitular() {
        return titular.nombreCompleto();
    }

    @Override
    public String toString() {
        return "TarjetaDeCredito{" + "entidadBancaria=" + entidadBancaria + ", nroTarjeta=" + nroTarjeta + ", saldo=" + saldo + ", entidadFinanciera=" + entidadFinanciera + ", titular=" + titula
    }
}
```

```
package ejercicio_13;

public enum EntidadFinanciera {
    VISA,MASTERCARD;
}
```

```
package ejercicio_13;
class Ticket {
    private String nombreApellido;
    private double montoTotal;
    private double montoPorCuota;

    public Ticket(String nombreApellido, double montoTotal, double montoPorCuota) {
        this.nombreApellido = nombreApellido;
        this.montoTotal = montoTotal;
        this.montoPorCuota = montoPorCuota;
    }

    @Override
    public String toString() {
        return "Ticket{" + "nombreApellido=" + nombreApellido + ", montoTotal=" + montoTotal + ", montoPorCuota=" + montoPorCuota + '}';
    }
}
```

```
package ejercicio_13;
public class Posnet {
    public static final double RECARGO_POR_CUOTA = 0.03;
    public static final int MIN_CANT_CUOTAS = 1;
    public static final int MAX_CANT_CUOTAS = 6;
    //atributos de clase --> estáticos
    // final --> son constantes

    public Ticket efectuarPago(TarjetaDeCredito tarjeta, double montoAAbonar, int cantCuotas ){
        Ticket elTicket = null;
        // se analiza que los datos sean validos, si no son validos, devuelve null
        if(datosValidos(tarjeta,montoAAbonar,cantCuotas)){
            double montoFinal = montoAAbonar + montoAAbonar*recargaoSegunCuotas(cantCuotas);
            if(tarjeta.tieneSaldoDisponible(montoFinal)){
                tarjeta.descontar(montoFinal);
                String nomApe= tarjeta.nombreCompletoDeTitular();
                double montoPorCuota=montoFinal/cantCuotas;
                elTicket= new Ticket(nomApe,montoFinal,montoPorCuota);
            }

        }
        return elTicket;
    }

    private boolean datosValidos(TarjetaDeCredito tarjeta,double monto,int cant){
        boolean esTarjetaValida = tarjeta != null; // una expresion booleana puede ser guardada de esta manera
        boolean esMontoValido = monto > 0;
        boolean cantCuotasValidas = cant >= MIN_CANT_CUOTAS && cant <= MAX_CANT_CUOTAS;
        return esTarjetaValida && esMontoValido && cantCuotasValidas ;
    }

    private double recargaoSegunCuotas(int cantCuotas){
        return (cantCuotas-1)*RECARGO_POR_CUOTA;
    }
}
```