EJERCICIO 2.- SOLUCIÓN

Construir las siguientes clases:

- Clase **Vivienda**, con atributos tipo (*enumerado* TipoVivienda), número de habitaciones (*entero*), metros cuadrados (*doble*), precio (*doble*), ciudad (*String*) y zona (*String*).
  La clase debe tener dos constructores, uno sin parámetros y otro parametrizado, para todos los atributos.
  Escribir algún método setter y getter para alguno de los atributos.
  Definir un método toString para obtener un String con todos los datos.
  Definir asimismo un método "comisión", que nos devuelva el importe de la comisión por venta, que será el 3 por ciento del precio de la vivienda.

```
1  public class Vivienda{
2      protected TipoVivienda tipo;
3      protected int numHabitaciones;
4      protected double metrosCuadrados;
5      protected double precio;
6      protected String ciudad;
7      protected String zona;
8
9      public Vivienda(){
10     }
11
12     public Vivienda( TipoVivienda tipo,int numHabitaciones, double metrosCuadrados,
13     double precio, String ciudad, String zona){
14         this.tipo =tipo;
15         this.numHabitaciones=numHabitaciones;
16         this.metrosCuadrados=metrosCuadrados;
17         this.ciudad=ciudad;
18         this.zona=zona;
19         this.precio=precio;
20     }
21
22     public String toString(){
23         return "Tipo: "+tipo + " - Num Habitaciones: "+numHabitaciones + " - m2: "+metrosCuadrados+
24         "\n Precio: "+precio+ " - Ciudad: "+ciudad+ " - Zona: "+zona;
25     }
26
27     public double comision(){
28          return precio*0.03;
29     }
30 }
```

- Clase **Casa**, derivada de la clase Vivienda, añadiendo los atributos privados "parcela" (*doble*) y "piscina" (*lógico*).
  Debe incluir también un constructor parametrizado que use el constructor de la clase base y también los datos para los nuevos atributos de esta clase.
  Sobrescribir el método toString de la clase base para añadir los nuevos datos.

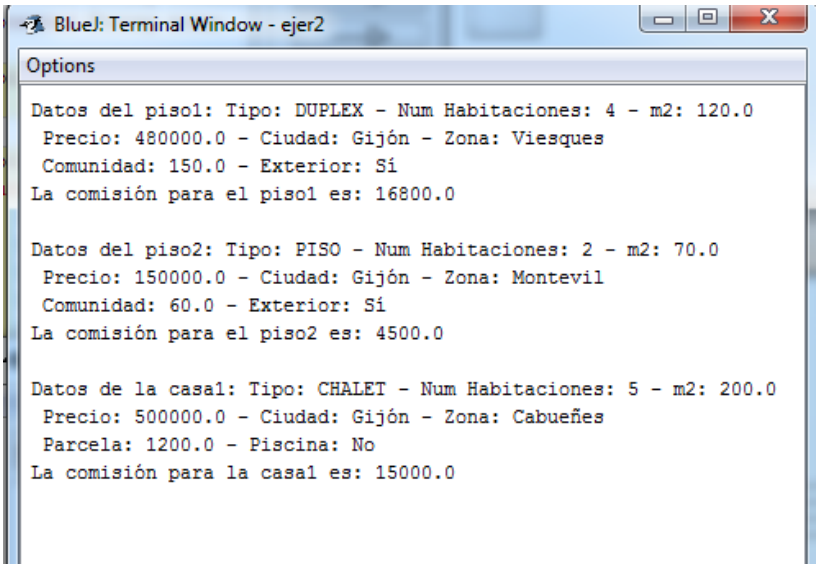
```
1  public class Casa extends Vivienda{
2      private double parcela;
3      private boolean piscina;
4
5      public Casa(TipoVivienda tipo,int numHabitaciones, double metrosCuadrados,
6          double precio, String ciudad, String zona, double parcela, boolean piscina){
7          super(tipo,numHabitaciones, metrosCuadrados, precio, ciudad, zona);
8          this.parcela=parcela;
9          this.piscina=piscina;
10     }
11
12     public String toString(){
13             String tienePiscina="No";
14             if (piscina)
15                 tienePiscina="Sí";
16             return super.toString()+ "\n Parcela: "+parcela+ " - Piscina: "+tienePiscina;
17     }
18 }
```

- Clase **Piso**, derivada de la clase Vivienda, añadiendo los atributos privados "comunidad", tipo *doble*, y "exterior", que será de tipo *lógico*.
  Debe incluir también un constructor parametrizado que use el constructor de la clase base y añada valores para los nuevos atributos.
  Sobrescribir el método "comisión" de forma que si los metros cuadrados del piso son menores o iguales que 100, la comisión sigue siendo la misma que en Vivienda, pero si esos metros cuadrados son mayores que 100, la comisión será del 3.5 por ciento de precio.

```
1  public class Piso extends Vivienda{
2      private double comunidad;
3      private boolean exterior;
4
5      public Piso(TipoVivienda tipo,int numHabitaciones, double metrosCuadrados,
6          double precio, String ciudad, String zona, double comunidad, boolean exterior){
7          super(tipo,numHabitaciones, metrosCuadrados, precio , ciudad, zona);
8          this.comunidad = comunidad;
9          this.exterior=exterior;
10     }
11
12     public String toString(){
13         String esExterior="No";
14            if (exterior)
15                esExterior="Sí";
16         return super.toString()+ "\n Comunidad: "+comunidad+ " - Exterior: "+esExterior;
17     }
18
19     public double comision(){
20         if (metrosCuadrados <=100)
21            return super.comision();
22         else
23            return precio*0.035;
24     }
25
26 }
```

- Clase con función principal, **TestViviendas**, en la que se declaren varios objetos de la clase Piso y de la clase Casa utilizando diferentes constructores, para posteriormente ver los datos de los objetos creados a través del método sobrescrito toString, con las correspondientes comisiones.

```
public class TestViviendas {

    public static void main(String args[]){
        Piso piso1, piso2;
        Casa casa1;
        piso1 = new Piso(TipoVivienda.DUPLEX,4,120,480000,"Gijón", "Viesques",150,true);
        piso2 = new Piso(TipoVivienda.PISO,2,70,150000,"Gijón", "Montevil",60,true);
        casa1 = new Casa(TipoVivienda.CHALET, 5, 200, 500000, "Gijón", "Cabueñes", 1200, false);

        //ver datos de los pisos
        System.out.println("Datos del piso1: "+piso1.toString());
        System.out.println("La comisión para el piso1 es: "+piso1.comision()+"\n");

        System.out.println("Datos del piso2: "+piso2.toString());
        System.out.println("La comisión para el piso2 es: "+piso2.comision()+"\n");

        System.out.println("Datos de la casa1: " +casa1.toString());
        System.out.println("La comisión para la casa1 es: "+casa1.comision()+"\n");
    }

}
```



- Para la declaración del campo enumerado TipoVivienda se usarán los siguientes valores:
  PISO, DUPLEX, ATICO, CASA, CHALET, ADOSADO, PAREADO

```
public enum TipoVivienda {
    PISO, DUPLEX, ATICO, CASA, CHALET, ADOSADO, PAREADO;
}
```