# PRÁCTICA Nro10 SIS103

**Nombre:** Gonzales Suyo Franz Reinaldo **C.U.** 111-500

**Carrera:** Ing. Ciencias de la Computación

**Práctica de Paquetes:**

Se definieron clases para representar inmuebles, como Inmueble, InmuebleVivienda, y subclases como Casa, Apartamento, etc. Cada clase tiene atributos como identificador, área, y dirección, métodos para calcular el precio y mostrar información. Se creó un programa de prueba en la clase Prueba que instancia inmuebles y muestra sus datos, calculando el precio basado en el área.

## Clase Inmueble:

```
package Inmuebles;

/**
 * Esta clase denominada Inmueble modela un inmueble que posee
 * como atributos un identificador, un area, una dirección y un precio
 * de venta. Es la clase raíz de una jerarquía de herencia.
 *
 * @version 1.2/2020
 */
public class Inmueble {
    // Atributo para el identificador inmobiliario de un inmueble
    protected int identificadorInmobiliario;
    protected int area; // Atributo que identifica el area de un inmueble
    protected String dirección; /*
                                 * Atributo que identifica la dirección de
                                 * un inmueble
                                 */
    protected double precioVenta; /*
                                   * Atributo que identifica el precio de
                                   * venta de un inmueble
                                   */

    /**
     * Constructor de la clase Inmueble
     *
     * @param identificadorInmobiliario Parámetro que define el
     *                                   identificador de un inmueble
     * @param area                       Parámetro que define el area de un
inmueble
     * @param dirección                  Parámetro que define la dirección donde
se
     *                                   encuentra localizado un inmueble
     */

    Inmueble(int identificadorInmobiliario, int area, String dirección) {
```

```
        this.identificadorInmobiliario = identificadorInmobiliario;
        this.area = area;
        this.dirección = dirección;
    }

    /**
     * Método que a partir del valor del area de un inmueble, calcula su
     * precio de venta
     *
     * @param valorArea El valor unitario por area de un determinado
     *                  inmueble
     * @return Precio de venta del inmueble
     */
    double calcularPrecioVenta(double valorArea) {
        precioVenta = area * valorArea;
        return precioVenta;
    }

    /**
    * Método que muestra en pantalla los datos de un inmueble
    */
    void imprimir() {
        System.out.println("Identificador inmobiliario = " +
identificadorInmobiliario);
        System.out.println("Area = " + area);
        System.out.println("Dirección = " + dirección);
        System.out.println("Precio de venta = $" + precioVenta);
    }
}
```

## Clase InmuebleVivienda:

```
package Inmuebles;

/**
 * Esta clase denominada InmuebleVivienda modela un inmueble
 * destinado para la vivienda con atributos como el numero de
 * habitaciones y el numero de baños que posee
 *
 * @version 1.2/2020
 */
public class InmuebleVivienda extends Inmueble {

    /*
     * Atributo que identifica el numero de habitación de un inmueble
     * para vivienda
     */
    protected int numeroHabitaciones;
    /*
     * Atributo que identifica el numero de baños de un inmueble para
     * vivienda
     */
    protected int numeroBaños;
```

```
    /**
     * Constructor de la clase InmuebleVivienda
     *
     * @param identificadorInmobiliario Parametro que define el
     *                                  identificador inmobiliario de un
inmueble
     *                                  para la vivienda
     * @param area                      Parametro que define el area de un
inmueble
     *                                  para la
     *                                  vivienda
     * @param dirección                 Parametro que define la dirección donde
se
     *                                  encuentra localizado un inmueble para la
     *                                  vivienda
     * @param numeroHabitaciones        Parametro que define el numero de
     *                                  habitaciones que tiene un inmueble para
la
     *                                  vivienda
     * @param numeroBaños               Parametro que define el numero de baños
     *                                  que tiene un inmueble para la vivienda
     */
    public InmuebleVivienda(int identificadorInmobiliario, int area, String
dirección, int numeroHabitaciones, int numeroBaños) {
        super(identificadorInmobiliario, area, dirección); /*
                                                            * Invoca al
                                                            * constructor de la
clase padre
                                                            */
        this.numeroHabitaciones = numeroHabitaciones;
        this.numeroBaños = numeroBaños;
    }

    /**
    * Método que muestra en pantalla los datos de un inmueble para la
    * vivienda
    */
    void imprimir() {
        super.imprimir(); // Invoca al método imprimir de la clase padre
        System.out.println("Numero de habitaciones = " + numeroHabitaciones);
        System.out.println("Numero de baños = " + numeroBaños);
    }
}
```

## Clase Casa:

```
package Inmuebles;

/**
 * Esta clase denominada Casa modela un tipo específico de inmueble
 * destinado para la vivienda con atributos como el numero de pisos
```

```
 * que tiene una casa.
 *
 * @version 1.2/2020
 */
public class Casa extends InmuebleVivienda {
    protected int numeroPisos; /*
                                * Atributo que identica el numero de
                                * pisos que tiene una casa
                                */

    /**
     * Constructor de la clase Casa
     *
     * @param identificadorInmobiliario Parametro que define el
     *                                  identificador inmobiliario de una casa
     * @param area                      Parametro que define el area de una casa
     * @param dirección                 Parametro que define la dirección donde
se
     *                                  encuentra localizada una casa
     * @param numeroHabitaciones        Parametro que define el numero de
     *                                  habitaciones que tiene una casa
     * @param numeroBaños               Parametro que define el numero de baños
     *                                  que tiene una casa
     * @param numeroPisos               Parametro que define el numero de pisos
     *                                  que tiene una casa
     */
    public Casa(int identificadorInmobiliario, int area, String dirección,
            int numeroHabitaciones, int numeroBaños, int numeroPisos) {
        // Invoca al constructor de la clase padre
        super(identificadorInmobiliario, area, dirección, numeroHabitaciones,
numeroBaños);
        this.numeroPisos = numeroPisos;
    }

    /**
    * Método que muestra en pantalla los datos de una casa
    */
    void imprimir() {
        super.imprimir(); // Invoca al método imprimir de la clase padre
        System.out.println("Numero de pisos = " + numeroPisos);
    }
}
```

## Clase Apartamento:

```
package Inmuebles;

/**
 * Esta clase denominada Apartamento modela un tipo de inmueble
 * específico destinado para la vivienda.
 *
 * @version 1.2/2020
```

```
 */
public class Apartamento extends InmuebleVivienda {
    /**
     * Constructor de la clase Apartamento
     *
     * @param identificadorInmobiliario Parametro que define el
     *                                  identificador inmobiliario de un
apartamento
     * @param area                      Parametro que define el area de un
     *                                  apartamento
     * @param dirección                 Parametro que define la dirección donde
se
     *                                  encuentra localizado un apartamento
     * @param númeroHabitaciones        Parametro que define el número de
     *                                  habitaciones que tiene un apartamento
     * @param númeroBaños               Parámetro que define el número de baños
     *                                  que tiene un apartamento
     */
    public Apartamento(int identificadorInmobiliario, int area, String
dirección, int númeroHabitaciones, int númeroBaños) {
        // Invoca al constructor de la clase padre
        super(identificadorInmobiliario, area, dirección,
                númeroHabitaciones, númeroBaños);
    }

    /**
     * Método que muestra en pantalla los datos de un apartamento
     */
    void imprimir() {
        super.imprimir(); // Invoca al método imprimir de la clase padre
    }
}
```

## Clase CasaRural:

```
package Inmuebles;

/**
 * Esta clase denominada CasaRural modela un tipo específico de casa
 * ubicada en el sector rural
 *
 * @version 1.2/2020
 */
public class CasaRural extends Casa {
    // Atributo que identifica el valor por area para una casa rural
    protected static double valorArea = 1500000;
    /*
     * Atributo que identifica la distancia a la que se encuentra la casa
     * rural de la cabecera municipal
     */
    protected int distanciaCabera;
    // Atributo que identifica la altitud a la que se encuentra una casa rural
```

```java
    protected int altitud;

    /**
     * Constructor de la clase CasaRural
     *
     * @param identificadorInmobiliario Parametro que define el
     *                                  identificador inmobiliario de una casa
rural
     * @param area                      Parametro que define el area de una casa
     *                                  rural
     * @param direccion                 Parametro que define la direccion donde
se
     *                                  encuentra localizada una casa rural
     * @param numeroHabitaciones        Parametro que define el numero de
     *                                  habitaciones que tiene una casa rural
     * @param numeroBaños               Parametro que define el numero de baños
     *                                  que tiene una casa rural
     * @param numeroPisos               Parametro que define el numero de pisos
     *                                  que tiene una casa rural
     * @param distanciaCabera           Parametro que define la distancia de la
     *                                  casa rural a la cabecera municipal
     * @param altitud                   Parametro que define la altitud sobre el
     *                                  nivel del
     *                                  mar en que se encuentra una casa rural
     */

    public CasaRural(int identificadorInmobiliario, int area, String direccion,
int numeroHabitaciones, int númeroBaños,
            int numeroPisos, int distanciaCabera, int altitud) {
        // Invoca al constructor de la clase padre
        super(identificadorInmobiliario, area, direccion,
                numeroHabitaciones, numeroBaños, numeroPisos);
        this.distanciaCabera = distanciaCabera;
        this.altitud = altitud;
    }

    /**
    * Método que muestra en pantalla los datos de una casa rural
    */
    void imprimir() {
        super.imprimir(); // Invoca al método imprimir de la clase padre
        System.out.println("Distancia la cabecera municipal = " +
numeroHabitaciones + " km.");
        System.out.println("Altitud sobre el nivel del mar = " + altitud + "
metros.");
        System.out.println();
    }
}
```

**Clase CasaUrbana:**

```java
package Inmuebles;
```

```java
/**
 * Esta clase denominada CasaRural modela un tipo específico de casa
 * ubicada en el sector rural
 *
 * @version 1.2/2020
 */
public class CasaRural extends Casa {
    // Atributo que identifica el valor por area para una casa rural
    protected static double valorArea = 1500000;
    /*
     * Atributo que identifica la distancia a la que se encuentra la casa
     * rural de la cabecera municipal
     */
    protected int distanciaCabera;
    // Atributo que identifica la altitud a la que se encuentra una casa rural
    protected int altitud;

    /**
     * Constructor de la clase CasaRural
     *
     * @param identificadorInmobiliario Parametro que define el
     *                                  identificador inmobiliario de una casa
rural
     * @param area                      Parametro que define el area de una casa
     *                                  rural
     * @param direccion                 Parametro que define la direccion donde
se
     *                                  encuentra localizada una casa rural
     * @param numeroHabitaciones        Parametro que define el numero de
     *                                  habitaciones que tiene una casa rural
     * @param numeroBaños               Parametro que define el numero de baños
     *                                  que tiene una casa rural
     * @param numeroPisos               Parametro que define el numero de pisos
     *                                  que tiene una casa rural
     * @param distanciaCabera           Parametro que define la distancia de la
     *                                  casa rural a la cabecera municipal
     * @param altitud                   Parametro que define la altitud sobre el
     *                                  nivel del
     *                                  mar en que se encuentra una casa rural
     */

    public CasaRural(int identificadorInmobiliario, int area, String direccion,
int numeroHabitaciones, int númeroBaños,
            int numeroPisos, int distanciaCabera, int altitud) {
        // Invoca al constructor de la clase padre
        super(identificadorInmobiliario, area, direccion,
                numeroHabitaciones, numeroBaños, numeroPisos);
        this.distanciaCabera = distanciaCabera;
        this.altitud = altitud;
    }

    /**
    * Método que muestra en pantalla los datos de una casa rural
```

```
    */
    void imprimir() {
        super.imprimir(); // Invoca al método imprimir de la clase padre
        System.out.println("Distancia la cabecera municipal = " +
numeroHabitaciones + " km.");
        System.out.println("Altitud sobre el nivel del mar = " + altitud + "
metros.");
        System.out.println();
    }
}
```

## Clase AparatamentoFamiliar:

```
package Inmuebles;

/**
 * Esta clase denominada ApartamentoFamiliar modela un tipo
 * específico de apartamento con atributos como el valor por erea y el
 * valor de la administracion.
 *
 * @version 1.2/2020
 */
public class ApartamentoFamiliar extends Apartamento {
    // Atributo que identifica el valor por erea de un apartamento familiar
    protected static double valorArea = 2000000;

    protected int valorAdministracion;

    /**
    * Constructor de la clase ApartamentoFamiliar
    * @param identificadorInmobiliario Parametro que define el
    * identificador inmobiliario de un apartamento familiar
    * @param erea Parametro que define el erea de un apartamento familiar
    * @param direccion Parametro que define la direccion donde se
    * encuentra localizado un apartamento familiar
    * @param numeroHabitaciones Parametro que define el numero de
    * habitaciones que tiene un apartamento familiar
    * @param numeroBaños Parametro que define el numero de baños
    * que tiene un apartamento familiar
    * @param valorAdministracion Parametro que define el valor de la
    * administracion de un apartamento familiar
    */
    public ApartamentoFamiliar(int identificadorInmobiliario, int erea,
        String direccion, int numeroHabitaciones, int numeroBaños, int valor
Administracion) {
        // Invoca al constructor de la clase padre
        super(identificadorInmobiliario, erea, direccion,
        numeroHabitaciones, numeroBaños);
        this.valorAdministracion = valorAdministracion;
    }
```

```
    /**
    * Método que muestra en pantalla los datos de un apartamento familiar
    */
    void imprimir() {
        super.imprimir(); // Invoca al método imprimir de la clase padre
        System.out.println("Valor de la administracion = $" +
valorAdministracion);
        System.out.println();
    }
}
```

## Clase Apartaestudio:

```
package Inmuebles;

/**
 * Esta clase denominada Apartaestudio modela un tipo específico de
 * apartamento que tiene una sola habitacion.
 *
 * @version 1.2/2020
 */
public class Apartaestudio extends Apartamento {
    // Atributo que identifica el valor por area de un apartaestudio
    protected static double valorArea = 1500000;

    /**
     * Constructor de la clase Apartaestudio
     *
     * @param identificadorInmobiliario Parametro que define el
     *                                  identificador inmobiliario de un
     *                                  apartaestudio
     * @param area                      Parametro que define el area de un
     *                                  apartaestudio
     * @param direccion                 Parametro que define la direccion donde
se
     *                                  encuentra localizado un apartaestudio
     * @param numeroHabitaciones        Parametro que define el numero de
     *                                  habitaciones que tiene un apartaestudio
     * @param numeroBaños               Parametro que define el numero de baños
     *                                  que tiene un apartaestudio
     */
    public Apartaestudio(int identificadorInmobiliario, int area, String
direccion,
            int numeroHabitaciones, int numeroBaños) {
        // Invoca al constructor de la clase padre
        // Los apartaestudios tienen una sola habitacion y un solo baño
        super(identificadorInmobiliario, area, direccion, 1, 1);
    }

    /**
     * Método que muestra en pantalla los datos de un apartaestudio
     */
```

```
    void imprimir() {
        super.imprimir(); // Invoca al método imprimir de la clase padre
        System.out.println();
    }
}
```

## Clase CasConjuntoCerrado:

```
package Inmuebles;

/**
 * Esta clase denominada CasaConjuntoCerrado modela un tipo
 * específico de casa urbana que se encuentra en un conjunto cerrado
 * con atributos como el valor por area, valor de la administracion y
 * valores booleanos para especificar si tiene piscina y campos deportivos.
 *
 * @version 1.2/2020
 */
public class CasaConjuntoCerrado extends CasaUrbana {
    // Atributo que define el valor por area de una casa en conjunto cerrado
    protected static double valorArea = 2500000;
    /*
     * Atributo que define el valor de administracion de una casa en
     * conjunto cerrado
     */
    protected int valorAdministracion;
    // Atributo que define si una casa en conjunto cerrado tiene piscina
    protected boolean tienePiscina;
    /*
     * Atributo que define si una casa en conjunto cerrado tiene campos
     * deportivos
     */
    protected boolean tieneCamposDeportivos;

    /**
     * Constructor de la clase CasaConjuntoCerrado
     *
     * @param identificadorInmobiliario Parametro que define el
     *                                  identificador inmobiliario de una casa
en
     *                                  conjunto cerrado
     * @param area                      Parametro que define el area de una casa
en
     *                                  conjunto
     *                                  cerrado
     * @param direccion                 Parametro que define la direccion donde
se
     *                                  encuentra localizada una casa en
conjunto
     *                                  cerrado
     * @param numeroHabitaciones        Parametro que define el numero de
```

```
     *                                  habitaciones que tiene una casa en
conjunto
     *                                  cerrado
     * @param numeroBaños               Parametro que define el numero de baños
     *                                  que tiene una casa en conjunto cerrado
     * @param numeroPisos               Parametro que define el numero de pisos
     *                                  que tiene una casa en conjunto cerrado
     * @param valorAdministracion       Parametro que define el valor de
     *                                  administracion para una casa en conjunto
     *                                  cerrado
     * @param tienePiscina              Parametro que define si una casa en
conjunto
     *                                  cerrado tiene o no piscina
     * @param tieneCamposDeportivos     Parametro que define si una casa
     *                                  en conjunto cerrado tiene o no campos
     *                                  deportivos
     */

    public CasaConjuntoCerrado(int identificadorInmobiliario, int area,
            String direccion, int numeroHabitaciones, int numeroBaños,
            int numeroPisos, int valorAdministracion, boolean tienePiscina,
            boolean tieneCamposDeportivos) {
        // Invoca al constructor de la clase padre
        super(identificadorInmobiliario, area, direccion,
                numeroHabitaciones, numeroBaños, numeroPisos);
        this.valorAdministracion = valorAdministracion;
        this.tienePiscina = tienePiscina;
        this.tieneCamposDeportivos = tieneCamposDeportivos;
    }

    /**
    * Método que muestra en pantalla los datos de una casa en conjunto
    * cerrado
    */
    void imprimir() {
        super.imprimir(); // Invoca al método imprimir de la clase padre
        System.out.println("Valor de la administracion = " +
valorAdministracion);
        System.out.println("Tiene piscina? = " + tienePiscina);
        System.out.println("Tiene campos deportivos? = " +
        tieneCamposDeportivos);
        System.out.println();
    }
}
```

### Clase CasaIndependiente:

```
/**
 * Esta clase denominada CasaIndependiente modela un tipo específico
 * de casa urbana que no esta en conjunto cerrado y es completamente
 * independiente de otras casas. Tiene un atributo estatico para
 * especificar un valor del area del inmueble.
```

```
 *
 * @version 1.2/2020
 */
public class CasaIndependiente extends CasaUrbana {
    // Atributo que identifica el valor por area de una casa independiente
    protected static double valorArea = 3000000;

    /**
     * Constructor de la clase CasaIndependiente
     *
     * @param identificadorInmobiliario Parametro que define el
     *                                  identificador inmobiliario de una casa
     *                                  independiente
     * @param area                      Parametro que define el area de una casa
     *                                  independiente
     * @param direccion                 Parametro que define la direccion donde
se
     *                                  encuentra localizada una casa
independiente
     * @param numeroHabitaciones        Parametro que define el numero de
     *                                  habitaciones que tiene una casa
     *                                  independiente
     * @param numeroBanos               Parametro que define el numero de banos
     *                                  que tiene una casa independiente
     * @param numeroPisos               Parametro que define el numero de pisos
     *                                  que tiene una casa independiente
     */
    public CasaIndependiente(int identificadorInmobiliario, int area,
            String direccion, int numeroHabitaciones, int numeroBanos, int
numeroPisos) {
        // Invoca al constructor de la clase padre
        super(identificadorInmobiliario, area, direccion,
                numeroHabitaciones, numeroBanos, numeroPisos);
    }

    /**
     * Método que muestra en pantalla los datos de una casa independiente
     */
    void imprimir() {
        super.imprimir(); // Invoca al método imprimir de la clase padre
        System.out.println();
    }
}
```

## Clase Local:

```
package Inmuebles;

/**
 * Esta clase denominada Local modela un tipo específico de inmueble
 * que no esta destinado para la vivienda que tiene como atributos un
 * tipo que especifica si es un local interno o que da a la calle.
 *
```

```
 * @version 1.2/2020
 */
public class Local extends Inmueble {

    enum tipo {
        INTERNO, CALLE
    }; /*
        * Tipo de inmueble especificado
        * como un valor enumerado
        */

    protected tipo tipoLocal; /*
                               * Atributo que identifica el tipo de
                               * inmueble
                               */

    /**
     * Constructor de la clase Local
     *
     * @param identificadorInmobiliario Parametro que define el
     *                                  identificador inmobiliario de un local
     * @param area                      Parametro que define el area de un local
     * @param direccion                 Parametro que define la direccion donde
se
     *                                  encuentra localizado un local
     * @param tipoLocal                 Parametro que define el tipo de local
     *                                  (interno o
     *                                  que da a la calle)
     */
    public Local(int identificadorInmobiliario, int area, String direccion,
            tipo tipoLocal) {
        // Invoca al constructor de la clase padre
        super(identificadorInmobiliario, area, direccion);
        this.tipoLocal = tipoLocal;
    }

    /**
    * Método que muestra en pantalla los datos de un local
    */
    void imprimir() {
        super.imprimir(); // Invoca al método imprimir de la clase padre
        System.out.println("Tipo de local = " + tipoLocal);
    }
}
```

## Clase Oficina:

```
package Inmuebles;

/**
 * Esta clase denominada Oficina modela un tipo específico de local
 * con atributos como el valor por area y un valor booleano para
 * determinar si pertenece o no al gobierno.
```

```java
 *
 * @version 1.2/2020
 */
public class Oficina extends Local {
    // Atributo que identifica el valor por area de una oficina
    protected static double valorArea = 3500000;
    // Atributo que identifica si una oficina pertenece o no al gobierno
    protected boolean esGobierno;

    /**
     * Constructor de la clase Oficina
     *
     * @param identificadorInmobiliario Parametro que define el
     *                                  identificador inmobiliario de una
oficina
     * @param area                      Parametro que define el area de una
oficina
     * @param direccion                 Parametro que define la direccion donde
se
     *                                  encuentra localizada una oficina
     * @param tipoLocal                 Parametro que define el tipo de una
oficina
     *                                  (interna o que da a la calle)
     * @param esGobierno                Parametro que define un valor booleano
para
     *                                  determinar si la oficina es del gobierno
o
     *                                  no
     */
    public Oficina(int identificadorInmobiliario, int area, String direccion,
tipo tipoLocal, boolean esGobierno) {
        // Invoca al constructor de la clase padre
        super(identificadorInmobiliario, area, direccion, tipoLocal);
        this.esGobierno = esGobierno;
    }

    /**
    * Método que muestra en pantalla los datos de una oficina
    */

    void imprimir() {
        super.imprimir(); // Invoca al método imprimir de la clase padre
        System.out.println("Es oficina gubernamental = " + esGobierno);
        System.out.println();
    }
}
```

## Clase Prueba:

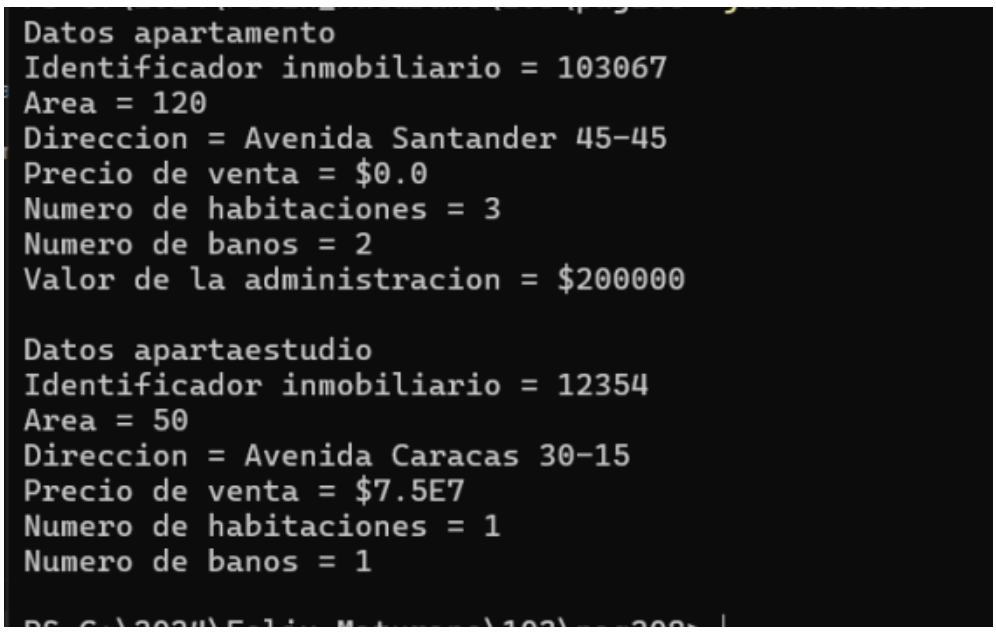
```java
package Inmuebles;

/**
 * Esta clase prueba diferentes inmuebles, se calcula su precio de
 * acuerdo al área y se muestran sus datos en pantalla
 *
 * @version 1.2/2020
 */
public class Prueba {
/**
* Método main que crea dos inmuebles, calcula su precio de
* acuerdo al área y se muestran sus datos en pantalla
*/
    public static void main(String args[]) {
        ApartamentoFamiliar apto1 = new ApartamentoFamiliar(103067,120, "Avenida
Santander 45-45",3,2,200000);

        System.out.println("Datos apartamento");

        apto1.calcularPrecioVenta(apto1.valorArea);
        apto1.imprimir();
        System.out.println("Datos apartamento");
        Apartaestudio aptestudio1 = new Apartaestudio(12354,50,"Avenida Caracas
30-15",1,1);
        aptestudio1.calcularPrecioVenta(aptestudio1.valorArea);
        aptestudio1.imprimir();
    }
}
```

## Prueba de Ejecución del Código



## 2. Ejecicios Propuestos

```
class Mascota {
    protected String nombre;
    protected int edad;
    protected String color;

    // Constructor de la clase Mascota
    public Mascota(String nombre, int edad, String color) {
        this.nombre = nombre;
        this.edad = edad;
        this.color = color;
    }
}

class Perro extends Mascota {
    protected double peso;
    protected boolean muerde;

    // Constructor de la clase Perro
    public Perro(String nombre, int edad, String color, double peso,
            boolean muerde) {
        super(nombre, edad, color);
        this.peso = peso;
        this.muerde = muerde;
    }

    // Método estático para sonido de perros
    public static void sonido() {
        System.out.println("Los perros ladran");
    }
}

// Clase para gatos
class Gato extends Mascota {
    protected double alturaSalto;
    protected double longitudSalto;

    // Constructor de la clase Gato
    public Gato(String nombre, int edad, String color, double alturaSalto,
double longitudSalto) {
        super(nombre, edad, color);
        this.alturaSalto = alturaSalto;
        this.longitudSalto = longitudSalto;
    }

    // Método estático para sonido de gatos
    public static void sonido() {
        System.out.println("Los gatos maúllan y ronronean");
    }
}

// Clases para razas específicas de perros
class PerroPequeno extends Perro {
```

```java
    // Constructor de la clase PerroPequeno
    public PerroPequeno(String nombre, int edad, String color, double peso,
boolean muerde) {
        super(nombre, edad, color, peso, muerde);
    }
}

class PerroMediano extends Perro {
    // Constructor de la clase PerroMediano
    public PerroMediano(String nombre, int edad, String color, double peso,
boolean muerde) {
        super(nombre, edad, color, peso, muerde);
    }
}

class PerroGrande extends Perro {
    // Constructor de la clase PerroGrande
    public PerroGrande(String nombre, int edad, String color, double peso,
boolean muerde) {
        super(nombre, edad, color, peso, muerde);
    }
}

// Clases para razas específicas de gatos
class GatoSinPelo extends Gato {
    // Constructor de la clase GatoSinPelo
    public GatoSinPelo(String nombre, int edad, String color, double
alturaSalto, double longitudSalto) {
        super(nombre, edad, color, alturaSalto, longitudSalto);
    }
}

class GatoPeloLargo extends Gato {
    // Constructor de la clase GatoPeloLargo
    public GatoPeloLargo(String nombre, int edad, String color, double
alturaSalto, double longitudSalto) {
        super(nombre, edad, color, alturaSalto, longitudSalto);
    }
}

class GatoPeloCorto extends Gato {
    // Constructor de la clase GatoPeloCorto
    public GatoPeloCorto(String nombre, int edad, String color, double
alturaSalto, double longitudSalto) {
        super(nombre, edad, color, alturaSalto, longitudSalto);
    }
}

public class MainMascota {

    public static void main(String[] args) {
        // Crear un perro
        Perro perro = new Perro("Fido", 3, "Marrón", 8.5, false);
```

```
        System.out.println("Datos del perro:");
        System.out.println("Nombre: " + perro.nombre);
        System.out.println("Edad: " + perro.edad);
        System.out.println("Color: " + perro.color);
        System.out.println("Peso: " + perro.peso);
        System.out.println("¿Muerde?: " + perro.muerde);
        Perro.sonido(); // Llamar al método estático de la clase Perro para
hacer sonido
        System.out.println(); // Espacio en blanco para separar perro y gato
        // Crear un gato
        Gato gato = new Gato("Whiskers", 2, "Blanco", 1.5, 2.0);
        System.out.println("Datos del gato:");
        System.out.println("Nombre: " + gato.nombre);
        System.out.println("Edad: " + gato.edad);
        System.out.println("Color: " + gato.color);
        System.out.println("Altura de salto: " + gato.alturaSalto);
        System.out.println("Longitud de salto: " + gato.longitudSalto);
        Gato.sonido(); // Llamar al método estático de la clase Gato para hacer
sonido
    }
}
```

## Prueba de Ejecución:

```
PS C:\GONZALES\SIS103\Paquetes> javac MainMascota.java
PS C:\GONZALES\SIS103\Paquetes> java MainMascota
Datos del perro:
Nombre: Fido
Edad: 3
Color: Marrón
Peso: 8.5
¿Muerde?: false
Los perros ladran

Datos del gato:
Nombre: Whiskers
Edad: 2
Color: Blanco
Altura de salto: 1.5
Longitud de salto: 2.0
Los gatos maúllan y ronronean
PS C:\GONZALES\SIS103\Paquetes>
```