Trabajo 4 POO (Grupal)

Johan Sebastián Robles Rincón

Juan Sebastián Duran Roldan

José Manuel Castro Balvin

Estudiantes de Ingeniería de Sistemas E Informática

Profesor Walter Hugo Arboleda Mazo

Universidad Nacional De Colombia Sede Medellín

22/10/2022

Links de videos de los ejercicios:

- **4.1 ->** Ejercicio 4.1: Cuenta - YouTube
- **4.2 ->** Ejercicio 4.2: Inmueble - YouTube
- **4.7 ->** Ejercicio clases abstractas 4.7 - YouTube
- **4.8 ->** Ejercicio metodos abstractos 4.8 - YouTube
- **8.1 ->** Ejercicio 8.1 - YouTube

Ejercicio 4.1

Clase Cuenta:

```
public class Cuenta {
   protected float saldo;
   protected int númeroConsignaciones = 0;
   protected int númeroRetiros = 0;
   protected float tasaAnual;
   protected float comisiónMensual = 0;

   public Cuenta(float saldo, float tasaAnual) {
      this.saldo = saldo;
      this.tasaAnual = tasaAnual;
   }

   public void consignar(float cantidad) {
      saldo = saldo + cantidad;
      númeroConsignaciones = númeroConsignaciones + 1;
   }

   public void retirar(float cantidad) {
      float nuevoSaldo = saldo - cantidad;
      if (nuevoSaldo >= 0) {
      saldo -= cantidad;
      númeroRetiros = númeroRetiros + 1;
      } else {
      System.out.println("La cantida a retirar excede el saldo actual.");
```

```
    }
  }

  public void calcularInterés() {
    float tasaMensual = tasaAnual / 12;
    float interesMensual = saldo * tasaMensual;
    saldo += interesMensual;
  }

  public void extractoMensual() {
    saldo -= comisiónMensual;
    calcularInterés();
  }
}
```

Clase CuentaAhorros:

```
public class CuentaAhorros extends Cuenta {
  private boolean activa;

  public CuentaAhorros(float saldo, float tasa) {
    super(saldo, tasa);
    if (saldo < 10000)
    activa = false;
    else
    activa = true;
  }

  public void retirar(float cantidad) {
    if (activa)
    super.retirar(cantidad);
  }

  public void consignar(float cantidad) {
```

```
    if (activa)
    super.consignar(cantidad);
  }

  public void extractoMensual() {
    if (númeroRetiros > 4) {
    comisiónMensual += (númeroRetiros - 4) * 1000;
  }

  super.extractoMensual();
    if ( saldo < 10000 )
    activa = false;
  }

  public void imprimir() {
    System.out.println("Saldo = $ " + saldo);
    System.out.println("Comisión mensual = $ " +
    comisiónMensual);
    System.out.println("Número de transacciones = " +
    (númeroConsignaciones + númeroRetiros));
    System.out.println();
  }
}
```

Clase CuentaCorriente:

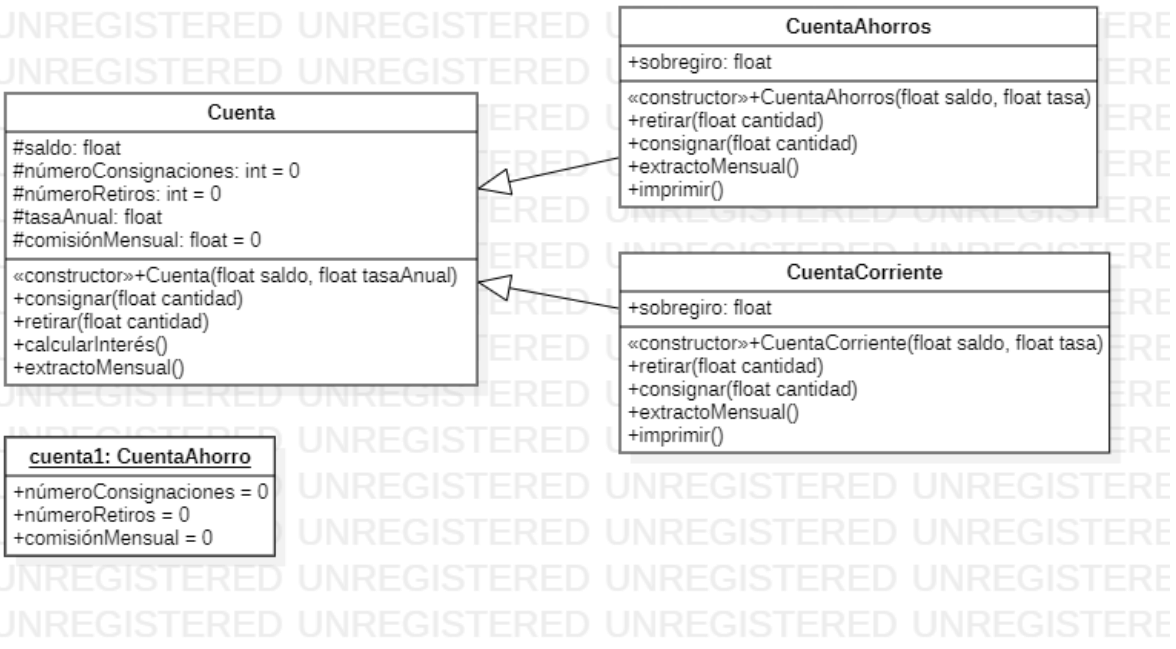
```
public class CuentaCorriente extends Cuenta {
  float sobregiro;

  public CuentaCorriente(float saldo, float tasa) {
    super(saldo, tasa);
    sobregiro = 0;
  }
```

```
public void retirar(float cantidad) {
   float resultado = saldo - cantidad;
   if (resultado < 0) {
   sobregiro = sobregiro - resultado;
   saldo = 0;
   } else {
   super.retirar(cantidad);
   }
}

public void consignar(float cantidad) {
   float residuo = sobregiro - cantidad;
   if (sobregiro > 0) {
   if ( residuo > 0) {
   sobregiro = 0;
   saldo = residuo;
   } else {
   sobregiro = -residuo;
   saldo = 0;
   }
   } else {
   super.consignar(cantidad);
   }
}

public void extractoMensual() {
   super.extractoMensual();
}

public void imprimir() {
   System.out.println("Saldo = $ " + saldo);
   System.out.println("Cargo mensual = $ " + comisiónMensual);
```

```
        System.out.println("Número de transacciones = " + (númeroConsignaciones + númeroRetiros));
        System.out.println("Valor de sogregiro = $" + (númeroConsignaciones + númeroRetiros));
        System.out.println();
    }
}
```

Clase Prueba Cuenta:

```
import java.util.*;

public class PruebaCuenta{

    public static void main(String args[]) {
        Scanner input = new Scanner(System.in);
        System.out.println("Cuenta de ahorros");
        System.out.println("Ingrese saldo inicial= $");
        float saldoInicialAhorros = input.nextFloat();
        System.out.print("Ingrese tasa de interés= ");
        float tasaAhorros = input.nextFloat();
        CuentaAhorros cuenta1 = new
        CuentaAhorros(saldoInicialAhorros, tasaAhorros);
        System.out.print("Ingresar cantidad a consignar: $");
        float cantidadDepositar = input.nextFloat();
        cuenta1.consignar(cantidadDepositar);
        System.out.print("Ingresar cantidad a retirar: $");
        float cantidadRetirar = input.nextFloat();
        cuenta1.retirar(cantidadRetirar);
        cuenta1.extractoMensual();
        cuenta1.imprimir();
    }
}
```



Link del video: [(1) Ejercicio 4.1: Cuenta - YouTube]

Ejercicio 4.2

Clase Apartaestudio:

```
package Inmuebles;

public class Apartaestudio extends Apartamento {

   protected static double valorArea = 1500000;


   public Apartaestudio(int identificadorInmobiliario, int área, String dirección, int númeroHabitaciones, int
númeroBaños) {

      super(identificadorInmobiliario, área, dirección, 1, 1);

   }


   void imprimir() {

      super.imprimir();

      System.out.println();

   }

}
```

Clase Apartamento:

```
package Inmuebles;
```

```
public class Apartamento extends InmuebleVivienda {

   public Apartamento(int identificadorInmobiliario, int área, String dirección, int númeroHabitaciones, int
númeroBaños) {

   super(identificadorInmobiliario, área, dirección, númeroHabitaciones, númeroBaños);

   }


   void imprimir() {

      super.imprimir();

   }

}
```

Clase ApartamentoFamiliar:

```
package Inmuebles;


public class ApartamentoFamiliar extends Apartamento {

   protected static double valorArea = 2000000;

   protected int valorAdministración;


   public ApartamentoFamiliar(int identificadorInmobiliario, int área, String dirección, int
númeroHabitaciones, int númeroBaños, int valorAdministración) {

      super(identificadorInmobiliario, área, dirección, númeroHabitaciones, númeroBaños);

      this.valorAdministración = valorAdministración;

   }


   void imprimir() {

      super.imprimir();

      System.out.println("Valor de la administración = $" + valorAdministración);

      System.out.println();

   }

}
```

Clase Casa:

```
package Inmuebles;
```

```
public class Casa extends InmuebleVivienda {
   protected int númeroPisos;
   public Casa(int identificadorInmobiliario, int área, String dirección, int númeroHabitaciones, int
númeroBaños, int númeroPisos) {
      super(identificadorInmobiliario, área, dirección, númeroHabitaciones, númeroBaños);
      this.númeroPisos = númeroPisos;
   }

   void imprimir() {
      super.imprimir();
      System.out.println("Número de pisos = " + númeroPisos);
   }
}
```

Clase CasaConjuntoCerrado:

```
package Inmuebles;

public class CasaConjuntoCerrado extends CasaUrbana {
   protected static double valorArea = 2500000;
   protected int valorAdministración;
   protected boolean tienePiscina;
   protected boolean tieneCamposDeportivos;

   public CasaConjuntoCerrado(int identificadorInmobiliario, int área, String dirección, int
númeroHabitaciones, int númeroBaños, int númeroPisos, int valorAdministración, boolean tienePiscina,
boolean tieneCamposDeportivos) {
      super(identificadorInmobiliario, área, dirección, númeroHabitaciones, númeroBaños, númeroPisos);
      this.valorAdministración = valorAdministración;
      this.tienePiscina = tienePiscina;
      this.tieneCamposDeportivos = tieneCamposDeportivos;
   }

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

```
        System.out.println("Valor de la administración = " +
        valorAdministración);
        System.out.println("Tiene piscina? = " + tienePiscina);
        System.out.println("Tiene campos deportivos? = " + tieneCamposDeportivos);
        System.out.println();
    }
}
```

Clase CasaIndependiente:

```
package Inmuebles;

public class CasaIndependiente extends CasaUrbana {
    protected static double valorArea = 3000000;

    public CasaIndependiente(int identificadorInmobiliario, int área, String dirección, int númeroHabitaciones,
int númeroBaños, int númeroPisos) {
        super(identificadorInmobiliario, área, dirección,
        númeroHabitaciones, númeroBaños, númeroPisos);
    }

    void imprimir() {
        super.imprimir();
        System.out.println();
    }
}
```

Clase CasaRural:

```
package Inmuebles;

public class CasaRural extends Casa {
    protected static double valorArea = 1500000;
    protected int distanciaCabera;
    protected int altitud;
```

```
    public CasaRural(int identificadorInmobiliario, int área, String dirección, int númeroHabitaciones, int
númeroBaños, int númeroPisos, int distanciaCabera, int altitud) {
        super(identificadorInmobiliario, área, dirección, númeroHabitaciones, númeroBaños, númeroPisos);
        this.distanciaCabera = distanciaCabera;
        this.altitud = altitud;
    }

    void imprimir() {
        super.imprimir();
        System.out.println("Distancia la cabecera municipal = " + númeroHabitaciones + " km.");
        System.out.println("Altitud sobre el nivel del mar = " + altitud + " metros.");
        System.out.println();
    }
}
```

Clase CasaUrbana:

```
package Inmuebles;

public class CasaUrbana extends Casa {
    public CasaUrbana(int identificadorInmobiliario, int área, String dirección, int númeroHabitaciones, int
númeroBaños, int númeroPisos) {
        super(identificadorInmobiliario, área, dirección, númeroHabitaciones, númeroBaños, númeroPisos);
    }

    void imprimir() {
        super.imprimir();
    }
}
```

Clase Inmueble:

```
package Inmuebles;

public class Inmueble {
    protected int identificadorInmobiliario;
    protected int área;
```

```
    protected String dirección;
    protected double precioVenta;

    Inmueble(int identificadorInmobiliario, int área, String dirección) {
        this.identificadorInmobiliario = identificadorInmobiliario;
        this.área = área;
        this.dirección = dirección;
    }

    double calcularPrecioVenta(double valorArea) {
        precioVenta = área * valorArea;
        return precioVenta;
    }

    void imprimir() {
        System.out.println("Identificador inmobiliario = " + identificadorInmobiliario);
        System.out.println("Area = " + área);
        System.out.println("Dirección = " + dirección);
        System.out.println("Precio de venta = $" + precioVenta);
    }
}
```

Clase InmuebleVivienda:

```
package Inmuebles;

public class InmuebleVivienda extends Inmueble {
    protected int númeroHabitaciones;
    protected int númeroBaños;
    public InmuebleVivienda(int identificadorInmobiliario, int área, String dirección, int númeroHabitaciones,
int númeroBaños) {
        super(identificadorInmobiliario, área, dirección);
        this.númeroHabitaciones = númeroHabitaciones;
        this.númeroBaños = númeroBaños;
```

```
    }
    void imprimir() {
        super.imprimir();
        System.out.println("Número de habitaciones = " + númeroHabitaciones);
        System.out.println("Número de baños = " + númeroBaños);
    }
}
```

Clase Local:

```
package Inmuebles;

public class Local extends Inmueble {
    enum tipo {INTERNO, CALLE};
    protected tipo tipoLocal;

    public Local(int identificadorInmobiliario, int área, String dirección, tipo tipoLocal) {
        super(identificadorInmobiliario, área, dirección);
        this.tipoLocal = tipoLocal;
    }

    void imprimir() {
        super.imprimir();
        System.out.println("Tipo de local = " + tipoLocal);
    }
}
```

Clase LocalComercial:

```
package Inmuebles;

public class LocalComercial extends Local {
    protected static double valorArea = 3000000;
    protected String centroComercial;
```

```
   public LocalComercial(int identificadorInmobiliario, int área, String dirección, tipo tipoLocal, String
centroComercial) {
      super(identificadorInmobiliario, área, dirección, tipoLocal);
      this.centroComercial = centroComercial;
   }
   void imprimir() {
      super.imprimir();
      System.out.println("Centro comercial = " + centroComercial);
      System.out.println();
   }
}
```

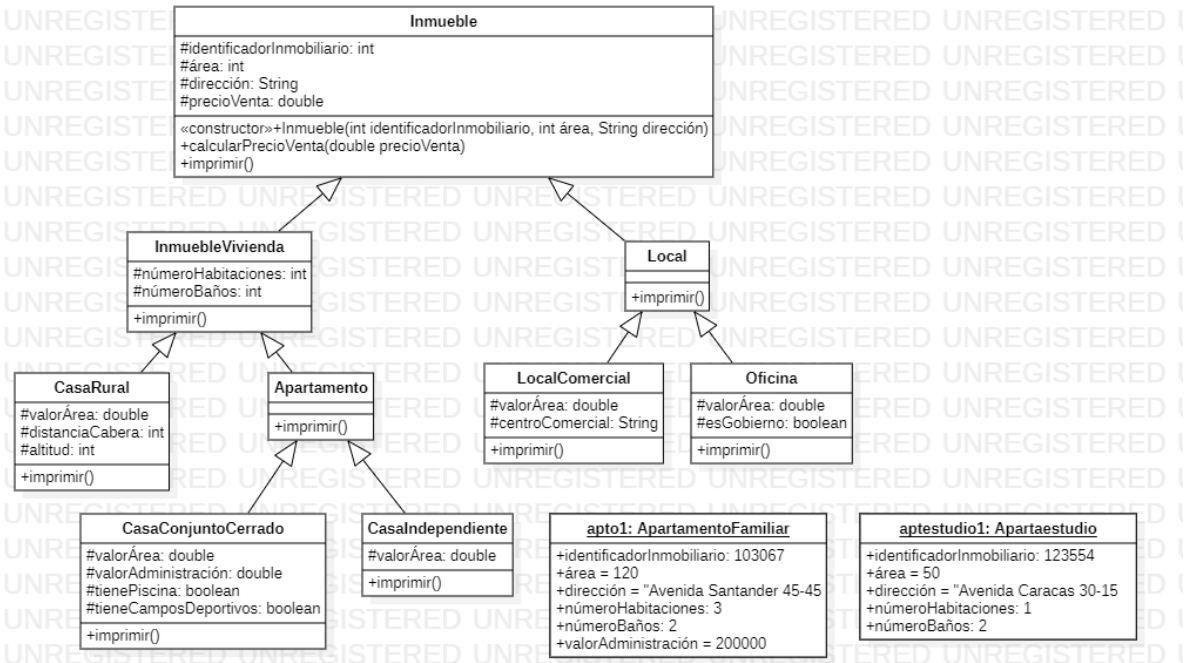
Clase Oficina:

```
package Inmuebles;

public class Oficina extends Local {
   protected static double valorArea = 3500000;
   protected boolean esGobierno;

   public Oficina(int identificadorInmobiliario, int área, String dirección, tipo tipoLocal, boolean esGobierno) {
      super(identificadorInmobiliario, área, dirección, tipoLocal);
      this.esGobierno = esGobierno;
   }
   void imprimir() {
      super.imprimir();
      System.out.println("Es oficina gubernamental = " + esGobierno);
      System.out.println();
   }
}
```

Clase Prueba:

```
package Inmuebles;
public class Prueba {
   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();
    }
}
```



Link del video: [Ejercicio 4.2: Inmueble - YouTube]

Ejercicio 4.7 Clases Abstractas

Clase Animal:

package Animales;

```
public abstract class Animal {
    protected String sonido;
    protected String alimentos;
    protected String hábitat;
    protected String nombreCientífico;

    public abstract String getNombreCientífico();

    public abstract String getSonido();

    public abstract String getAlimentos();

    public abstract String getHábitat();
}
```

Clase Cánido:

```
package Animales;
public abstract class Cánido extends Animal {
}
```

Clase Felino:

```
package Animales;
public abstract class Felino extends Animal {
}
```

Clase Gato:

```
package Animales;
public class Gato extends Felino {
   public String getSonido() {
      return "Maullido";
   }
   public String getAlimentos() {
      return "Ratones";
   }
   public String getHábitat() {
      return "Doméstico";
   }
   public String getNombreCientífico() {
      return "Felis silvestris catus";
   }
}
```

Clase león:

```
package Animales;
public class León extends Felino {
   public String getSonido() {
      return "Rugido";
   }
   public String getAlimentos() {
      return "Carnívoro";
   }
   public String getHábitat() {
      return "Praderas";
   }
   public String getNombreCientífico() {
      return "Panthera leo";
```

```
    }
}
```

Clase Lobo:

```
package Animales;
public class Lobo extends Cánido {
    public String getSonido() {
        return "Aullido";
    }
    public String getAlimentos() {
        return "Carnívoro";
    }
    public String getHábitat() {
        return "Bosque";
    }
    public String getNombreCientífico() {
        return "Canis lupus";
    }
}
```

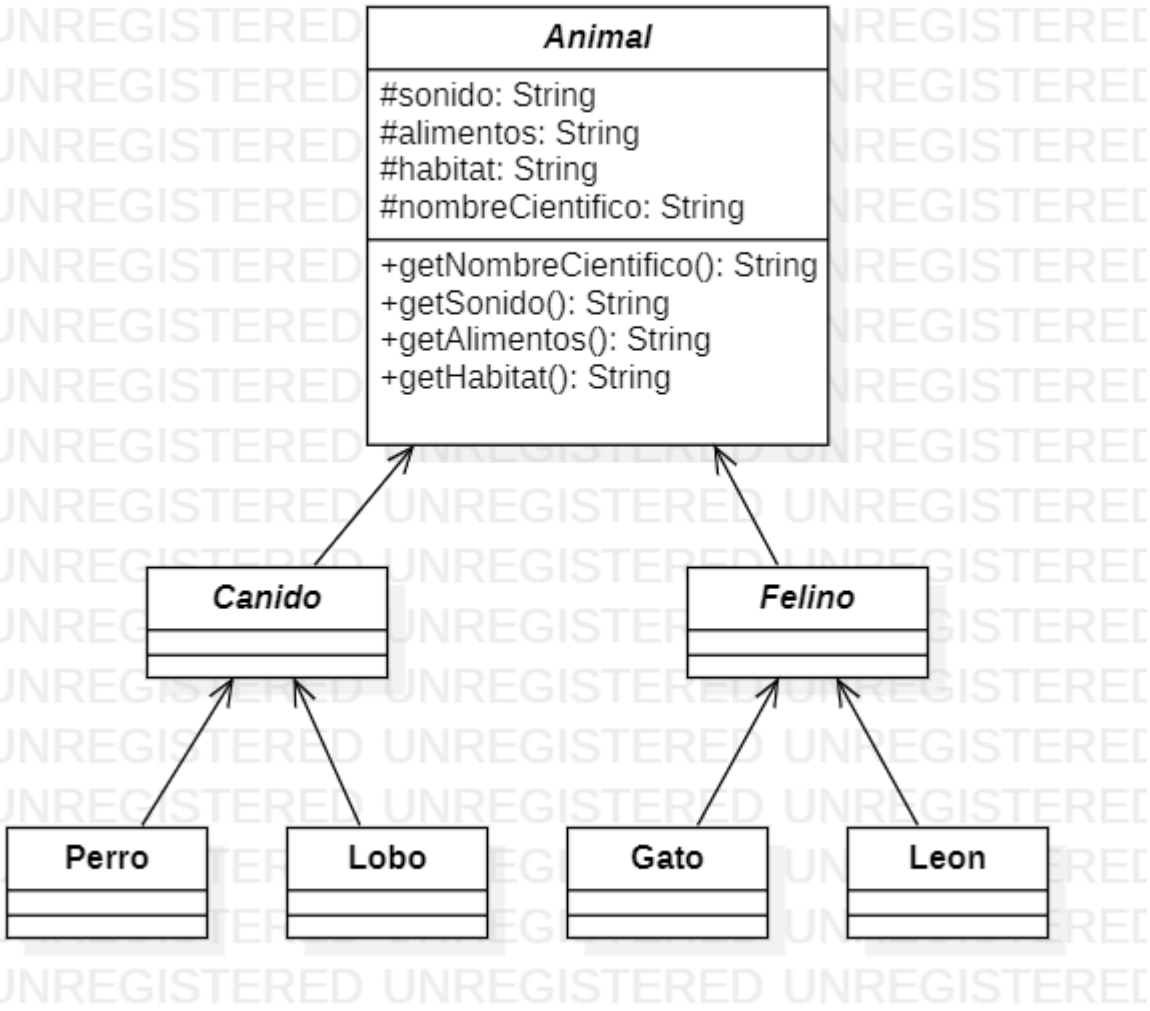
Clase Perro:

```
package Animales;
public class Perro extends Cánido {
    public String getSonido() {
        return "Ladrido";
    }
    public String getAlimentos() {
        return "Carnívoro";
    }
    public String getHábitat() {
        return "Doméstico";
```

```
    }
    public String getNombreCientífico() {
        return "Canis lupus familiaris";
    }
}
```

Clase Prueba:

```
package Animales;
public class Prueba {
public static void main(String[] args) {
        Animal[] animales = new Animal[4];
        animales[0] = new Gato();
        animales[1] = new Perro();
        animales[2] = new Lobo();
        animales[3] = new León();
        for (int i = 0; i < animales.length; i++) {
            System.out.println(animales[i].getNombreCientífico());
            System.out.println("Sonido: " + animales[i].getSonido());
            System.out.println("Alimentos: " + animales[i].
            getAlimentos());
            System.out.println("Hábitat: " + animales[i].getHábitat());
            System.out.println();
        }
    }
```



}

Link del video: [Ejercicio clases abstractas 4.7 - YouTube]

Ejercicio 4.8 Métodos abstractos

Clase ciclista:

```
package CarreraCiclistica;
public abstract class Ciclista {

    private int identificador;
    private String nombre;
    private int tiempoAcumulado = 0;

    public Ciclista(int identificador, String nombre) {
      this.identificador = identificador;
      this.nombre = nombre;
    }
  abstract String imprimirTipo();
  protected int getIdentificador() {
    return identificador;
  }
  protected void setIdentificador() {
    this.identificador = identificador;
  }
  protected String getNombre() {
    return nombre;
  }
  protected void setNombre(String nombre) {
    this.nombre = nombre;
  }
  protected int getPosiciónGeneral(int posiciónGeneral) {
```

```
        return posiciónGeneral;
    }
    protected void setPosiciónGeneral(int posiciónGeneral) {
        posiciónGeneral = posiciónGeneral;
    }

    protected int getTiempoAcumulado() {
        return tiempoAcumulado;
    }
    protected void setTiempoAcumulado(int tiempoAcumulado) {
        this.tiempoAcumulado = tiempoAcumulado;
    }
    protected void imprimir() {
        System.out.println("Identificador = " + identificador);
        System.out.println("Nombre = " + nombre);
        System.out.println("Tiempo Acumulado = " +
        tiempoAcumulado);
    }
}
```

Clase Contrarrelojista:

```
package CarreraCiclistica;
public class Contrarrelojista extends Ciclista {
    private double velocidadMáxima;
    public Contrarrelojista(int identificador, String nombre, double
    velocidadMáxima) {
        super(identificador, nombre);
        this.velocidadMáxima = velocidadMáxima;
    }
    protected double getVelocidadMáxima() {
```

```
        return velocidadMáxima;
    }

    protected void setVelocidadMáxima(double velocidadMáxima) {
        this.velocidadMáxima = velocidadMáxima;
    }
    protected void imprimir() {
        super.imprimir(); // Invoca el método imprimir de la clase padre
        System.out.println("Aceleración promedio = " +
        velocidadMáxima);
    }
    protected String imprimirTipo() {
        return "Es un constrarrelojista";
    }
}
```

Clase Equipo:

```
package CarreraCiclistica;
import java.util.*;
public class Equipo {
    private String nombre;
    private static double totalTiempo;
    private String país;
    Vector listaCiclistas;
    public Equipo(String nombre, String país) {
        this.nombre = nombre;
        this.país = país;
        totalTiempo = 0;
        listaCiclistas = new Vector();
    }
```

```
public String getNombre() {
   return nombre;
}
public void setNombre(String nombre) {
   this.nombre = nombre;
}
private String getPaís() {
   return país;
}
private void setPaís(String país) {
   this.país = país;
}
void añadirCiclista(Ciclista ciclista) {
   listaCiclistas.add(ciclista);
}
void listarEquipo() {
   for (int i = 0; i < listaCiclistas.size(); i++) {
      Ciclista c = (Ciclista) listaCiclistas.elementAt(i);
      System.out.println(c.getNombre());
   }
}
void buscarCiclista() {
   Scanner sc = new Scanner(System.in);
   String nombreCiclista = sc.next();
   for (int i = 0; i < listaCiclistas.size(); i++) {
      Ciclista c = (Ciclista) listaCiclistas.elementAt(i);
      if (c.getNombre().equals(nombreCiclista)) {
         System.out.println(c.getNombre());
      }
```

```
        }
    }
    void calcularTotalTiempo() {
        for (int i = 0; i < listaCiclistas.size(); i++) {
            Ciclista c = (Ciclista) listaCiclistas.elementAt(i);
            totalTiempo = totalTiempo + c.getTiempoAcumulado();
        }
    }
    void imprimir() {
        System.out.println("Nombre del equipo = " + nombre);
        System.out.println("País = " + país);
        System.out.println("Total tiempo del equipo = " + totalTiempo);
    }
}
```

Clase Escalador:

```
package CarreraCiclistica;
public class Escalador extends Ciclista {
    private double aceleraciónPromedio;
    private double gradoRampa;
    public Escalador(int identificador, String nombre, double
    aceleraciónPromedio, double gradoRampa) {
        super(identificador, nombre);
        this.aceleraciónPromedio = aceleraciónPromedio;
        this.gradoRampa = gradoRampa;
    }
    protected double getAceleraciónPromedio() {
        return aceleraciónPromedio;
    }
    protected void setAceleraciónPromedio(double
```

```
    aceleraciónPromedio) {
        this.aceleraciónPromedio = aceleraciónPromedio;
    }
    protected double getGradoRampa() {
        return gradoRampa;
    }
    protected void setGradoRampa(double gradoRampa) {
        this.gradoRampa = gradoRampa;
    }
    protected void imprimir() {
        super.imprimir(); // Invoca el método imprimir de la clase padre
        System.out.println("Aceleración promedio = " +
        aceleraciónPromedio);
        System.out.println("Grado de rampa = " + gradoRampa);
    }
    protected String imprimirTipo() {
        return "Es un escalador";
    }
}
```

Clase Prueba:

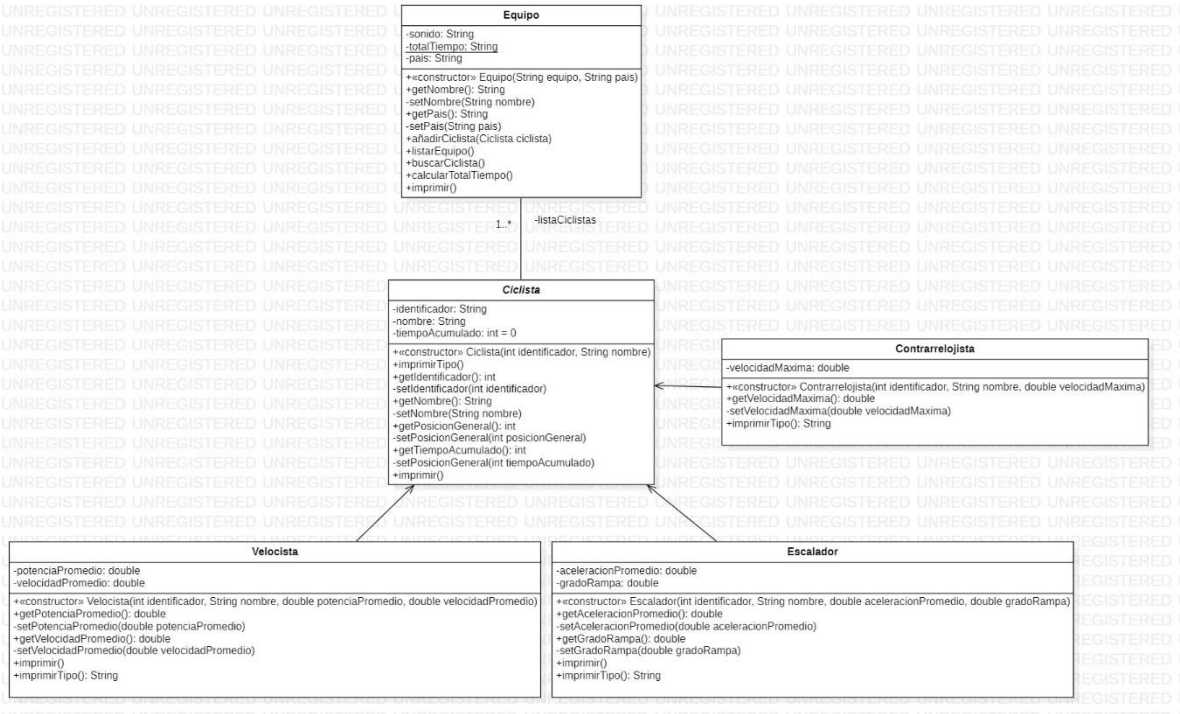
```
package CarreraCiclistica;
public class Prueba {
    public static void main(String args[]) {
        Equipo equipo1 = new Equipo("Sky","Estados Unidos");
        Velocista velocista1 = new Velocista(123979, "Geraint Thomas",
        320, 25);
        Escalador escalador1 = new Escalador(123980, "Egan Bernal",
        25, 10);
        Contrarrelojista contrarrelojista1 = new Contrarrelojista(123981,
```

```
        "Jonathan Castroviejo", 120);
        equipo1.añadirCiclista(velocista1);
        equipo1.añadirCiclista(escalador1);
        equipo1.añadirCiclista(contrarrelojista1);
        velocista1.setTiempoAcumulado(365);
        escalador1.setTiempoAcumulado(385);
        contrarrelojista1.setTiempoAcumulado(370);
        equipo1.calcularTotalTiempo();
        equipo1.imprimir();
        equipo1.listarEquipo();
    }
}
```

Clase Velocista:

```
package CarreraCiclistica;
public class Velocista extends Ciclista {
    private double potenciaPromedio;
    private double velocidadPromedio;
    public Velocista(int identificador, String nombre, double
    potenciaPromedio, double velocidadPromedio) {
        super(identificador, nombre);
        potenciaPromedio = potenciaPromedio;
        this.velocidadPromedio = velocidadPromedio;
    }
    protected double getPotenciaPromedio() {
        return potenciaPromedio;
    }
    protected void setPotenciaPromedio(double potenciaPromedio) {
```

```
        this.potenciaPromedio = potenciaPromedio;
    }
    protected double getvelocidadPromedio() {
        return velocidadPromedio;
    }
    protected void setVelocidadPromedio(double velocidadPromedio) {
        this.velocidadPromedio = velocidadPromedio;
    }
    protected void imprimir() {
        super.imprimir();
        System.out.println("Potencia promedio = " + potenciaPromedio);
        System.out.println("Velocidad promedio = " +
        velocidadPromedio);
    }
    protected String imprimirTipo() {
        return "Es un velocista";
    }
}
```

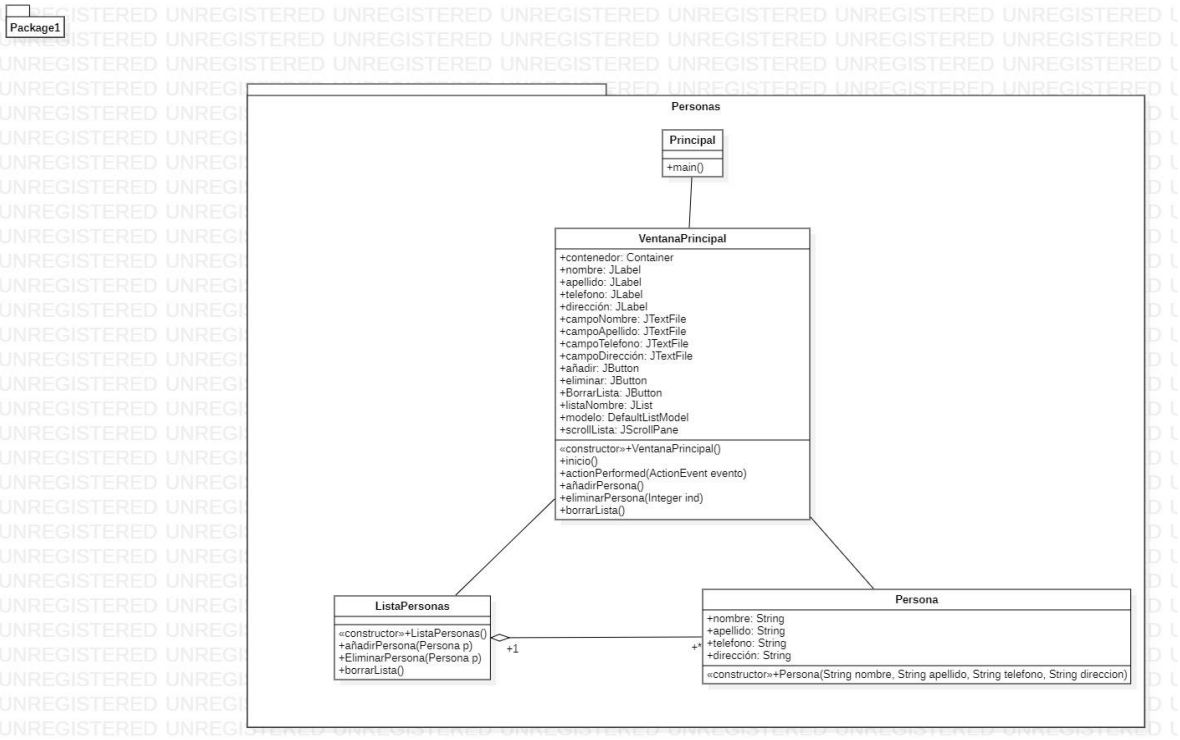


Link del video:

[Ejercicio metodos abstractos 4.8 - YouTube]

Ejercicio 8.1.

Este ejercicio consta de 4 archivos; PrincipalPersonas.java, Personas.java, ListaPersonas.java y VentanaPrincipal



PrincipalPersonas:

```
package personas;

public class PrincipalPersonas {
    public static void main(String[] args) {
        VentanaPrincipal ventana = new VentanaPrincipal();
        ventana.setVisible(true);
        ventana.setTitle("Lista De Personas");

    }

}
```

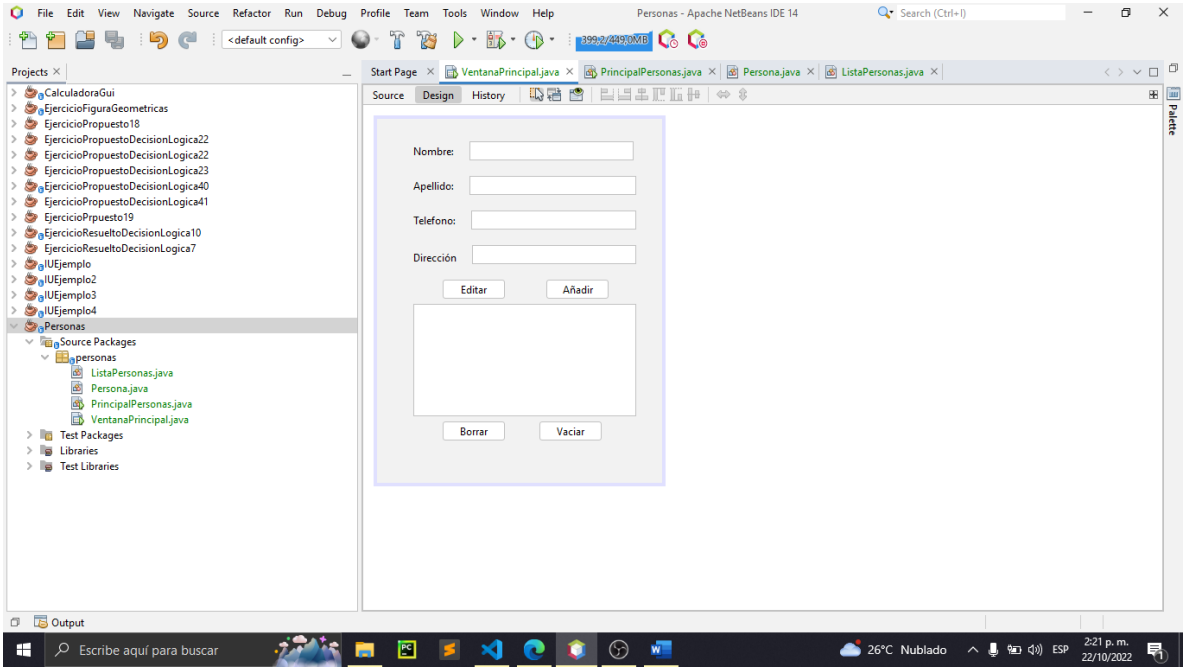
Personas:

```java
package personas;

public class Persona {
    String nombre;
    String apellidos;
    String teléfono;
    String dirección;

    public Persona(String nombre, String apellidos, String teléfono, String
dirección) {
        this.nombre = nombre;
        this.apellidos = apellidos;
        this.teléfono = teléfono;
        this.dirección = dirección;
    }

}
```

ListaPersonas

```java
package personas;
import java.util.*;

public class ListaPersonas {

    ArrayList<Persona> listaPersonas; // Atributo que identifica un vector
de personas
    public ListaPersonas() {
        listaPersonas = new ArrayList<Persona>();
    }
    public void añadirPersona(Persona p) {
        listaPersonas.add(p);
    }

    public void eliminarPersona(int i) {
        listaPersonas.remove(i);
    }

    public void borrarLista() {
        listaPersonas.removeAll(listaPersonas);
    }

}
```

VentanaPrincipal




```
public class VentanaPrincipal extends javax.swing.JFrame {

    ListaPersonas lista = new ListaPersonas();
    DefaultListModel modelo = new DefaultListModel();

    public VentanaPrincipal() {
        initComponents();
        listPersonas.setModel(modelo);
    }
private void btnAñadirActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_btnAñadirActionPerformed
        Persona persona;
        String nombre = txtNombre.getText();
        String apellido = txtApellido.getText();
        String telefono = txtTelefono.getText();
        String direccion = txtDireccion.getText();

        if (nombre.equals("")|| apellido.equals("") || telefono.equals("")
|| direccion.equals("")){
                JOptionPane.showMessageDialog(null,"Faltan datos por
llenar.");
        }else{
```

```
            if (verificarNombres(nombre, apellido)){
                JOptionPane.showMessageDialog(null,"La persona ya existe en
la lista.");
            }else{
                try{
                persona = new Persona(nombre, apellido, telefono,
direccion);
                lista.añadirPersona(persona);
                JOptionPane.showMessageDialog(null,"La persona ha sido
añadido correctamente.");
                modelo.removeAllElements();
                lista.listaPersonas.forEach(p ->
modelo.addElement(String.format("%s-%s-%s-%s.", p.nombre, p.apellidos,
p.teléfono, p.dirección)));

                txtNombre.setText("");
                txtApellido.setText("");
                txtTelefono.setText("");
                txtDireccion.setText("");
                }catch (NumberFormatException ex){
                    JOptionPane.showMessageDialog(null,"Los valores
ingresados estan en formato Incorrecto.");
                    txtNombre.setText("");
                    txtApellido.setText("");
                    txtTelefono.setText("");
                    txtDireccion.setText("");
                }

            }
            }

    }//GEN-LAST:event_btnAñadirActionPerformed

    private void btnVaciarActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_btnVaciarActionPerformed
        lista.borrarLista();
        modelo.clear();
        txtNombre.setText("");
        txtApellido.setText("");
        txtTelefono.setText("");
        txtDireccion.setText("");
    }//GEN-LAST:event_btnVaciarActionPerformed

    private void btnBorrarActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_btnBorrarActionPerformed
```

```
        //System.out.println(listPersonas.getSelectedIndex());
        try{
            lista.eliminarPersona(listPersonas.getSelectedIndex());
            modelo.remove(listPersonas.getSelectedIndex());
            txtNombre.setText("");
            txtApellido.setText("");
            txtTelefono.setText("");
            txtDireccion.setText("");
        }catch (IndexOutOfBoundsException ex){
            JOptionPane.showMessageDialog(null,"No se ha seleccionado
ninguna persona.");
        }

    }//GEN-LAST:event_btnBorrarActionPerformed

    private void btnEditarActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_btnEditarActionPerformed
        lista.listaPersonas.get(listPersonas.getSelectedIndex()).nombre =
txtNombre.getText();
        lista.listaPersonas.get(listPersonas.getSelectedIndex()).apellidos =
txtApellido.getText();
        lista.listaPersonas.get(listPersonas.getSelectedIndex()).teléfono =
txtTelefono.getText();
        lista.listaPersonas.get(listPersonas.getSelectedIndex()).dirección =
txtDireccion.getText();
        JOptionPane.showMessageDialog(null,"La persona ha sido modificada
correctamente.");
        txtNombre.setText("");
        txtApellido.setText("");
        txtTelefono.setText("");
        txtDireccion.setText("");
        modelo.removeAllElements();
        lista.listaPersonas.forEach(p ->
modelo.addElement(String.format("%s-%s-%s-%s.", p.nombre, p.apellidos,
p.teléfono, p.dirección)));
    }//GEN-LAST:event_btnEditarActionPerformed
 private void listPersonasMouseClicked(java.awt.event.MouseEvent evt)
{//GEN-FIRST:event_listPersonasMouseClicked
        if (!listPersonas.isSelectionEmpty()){
            txtNombre.setText(lista.listaPersonas.get(listPersonas.getSelect
edIndex()).nombre);
            txtApellido.setText(lista.listaPersonas.get(listPersonas.getSele
ctedIndex()).apellidos);
            txtTelefono.setText(lista.listaPersonas.get(listPersonas.getSele
ctedIndex()).teléfono);
```

```
            txtDireccion.setText(lista.listaPersonas.get(listPersonas.getSel
ectedIndex()).dirección);
        }

    }//GEN-LAST:event_listPersonasMouseClicked

    public boolean verificarNombres(String nombre, String apellido){
        for (int i = 0; i<lista.listaPersonas.size(); i++){
            if (lista.listaPersonas.get(i).nombre.equals(nombre) &&
lista.listaPersonas.get(i).apellidos.equals(apellido)){
                return true;
            }
        }
        return false;
    }
```

Link del video:

[Ejercicio 8.1 - YouTube]