El futuro digital es de todos

MinTIC

# Ciclo: Programación Básica

# Programación Orientada a Objetos (POO)

## Misión TIC 2022

unab
VIGILADA MINEDUCACIÓN

Misión TIC 2022

# Programación Básica

- Herencia
- Polimorfismo
- Interfaces – Clases Abstractas

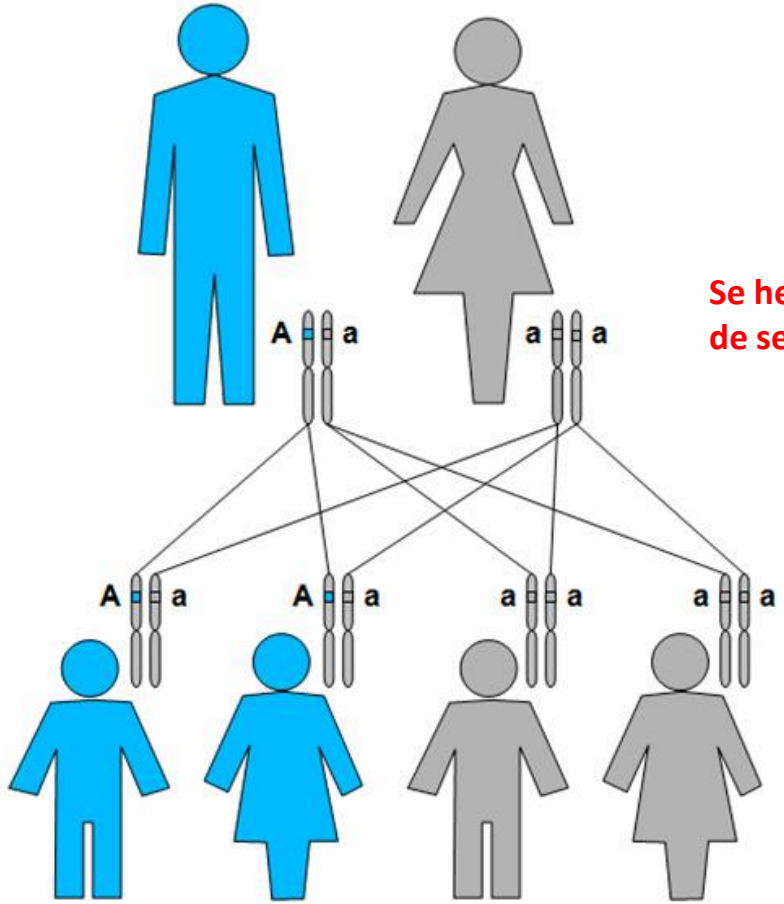
## Herencia

La **herencia** es un pilar importante de POO (Programación Orientada a Objetos). Es el mecanismo en Java por el cual una clase permite heredar las características (atributos y métodos) a otras clases.

En el lenguaje de Java, una clase que se hereda se denomina **superclase**. La clase que hereda se llama **subclase**. Se hereda atributos y métodos definidos por la superclase y la subclase puede agregar sus propios elementos únicos, atributos y métodos. También se habla del concepto de **ascendencia** y **descendencia.**

# Programación Orientada a Objetos

## Herencia - Humanos




Se heredan características físicas, formas de ser o actuar

# Programación Orientada a Objetos

## Herencia

**Terminología importante:**

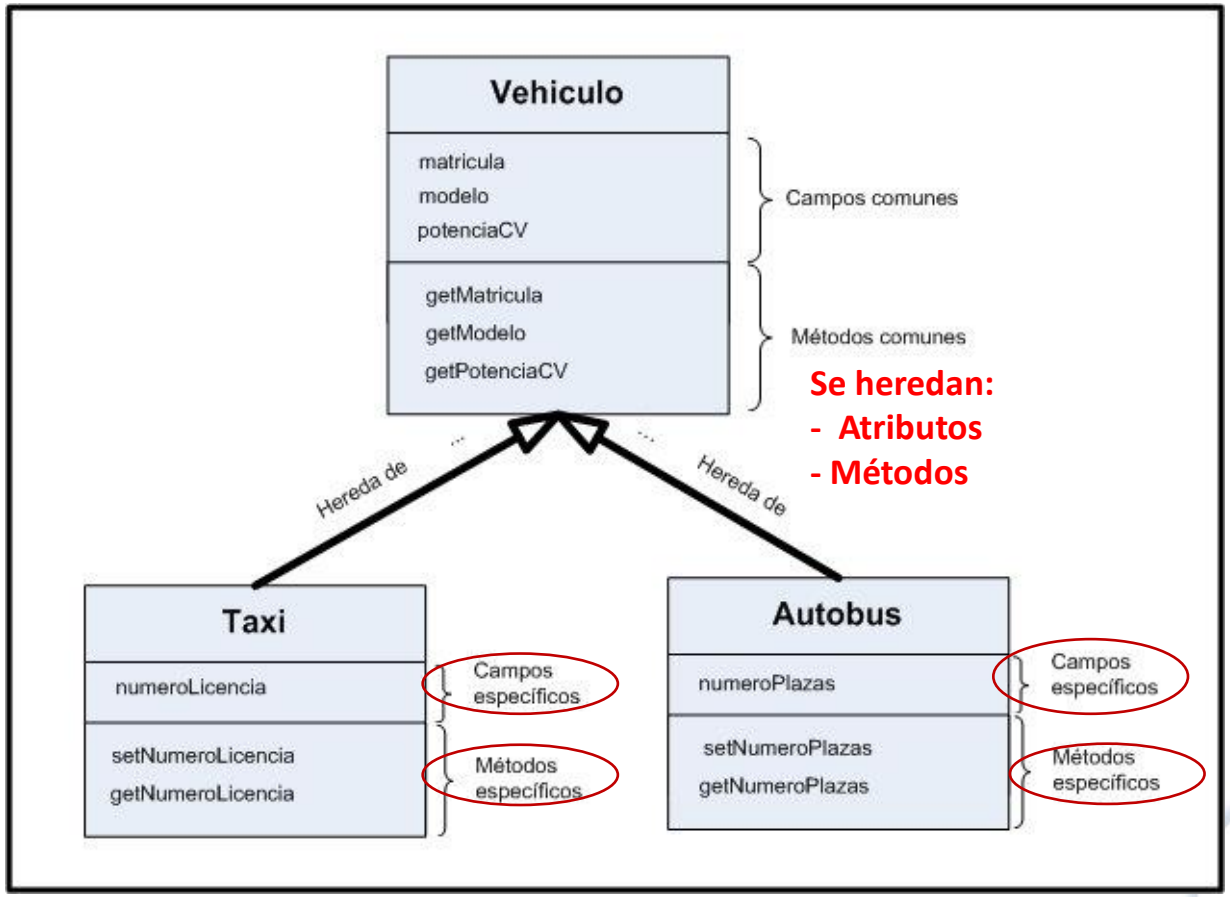
•**Superclase**: la clase cuyas atributos y métodos se heredan se conoce como superclase (o una clase base o una clase principal).

•**Subclase**: la clase que hereda la otra clase se conoce como subclase (o una clase derivada, clase extendida o clase hija). La subclase puede agregar sus propios atributos y métodos además de los atributos y métodos de la superclase.

•**Reutilización**: la herencia respalda el concepto de “reutilización”, es decir, cuando queremos crear una clase nueva y ya hay una clase que incluye parte del código que queremos, podemos derivar nuestra nueva clase de la clase existente. Al hacer esto, estamos reutilizando los atributos y métodos de la clase existente.

# Programación Orientada a Objetos

## Herencia - POO



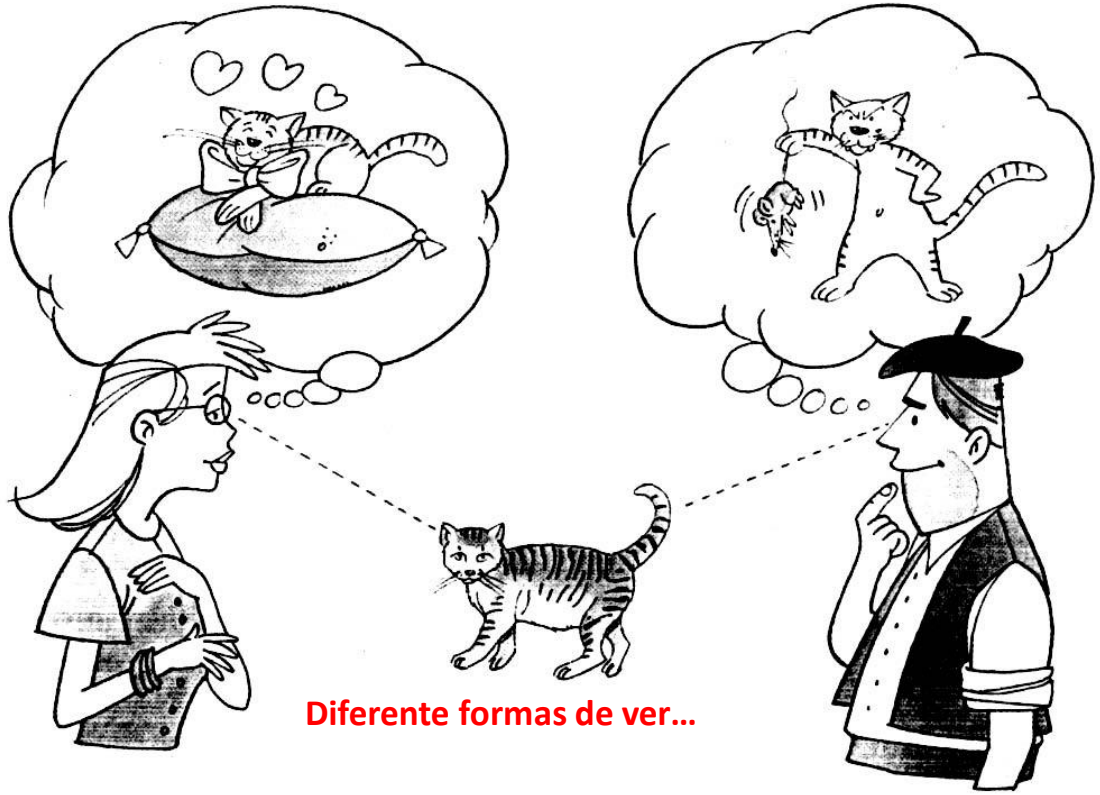
## Polimorfismo

En **programación** orientada a objetos se denomina **polimorfismo** a la capacidad que tienen los objetos de una clase de responder al mismo mensaje o evento en función de los parámetros utilizados durante su invocación.

En otras palabras, se puede apreciar el polimorfismo cuando método con el mismo nombre, pero diferente proceso se define en diferentes clases.

# Programación Orientada a Objetos

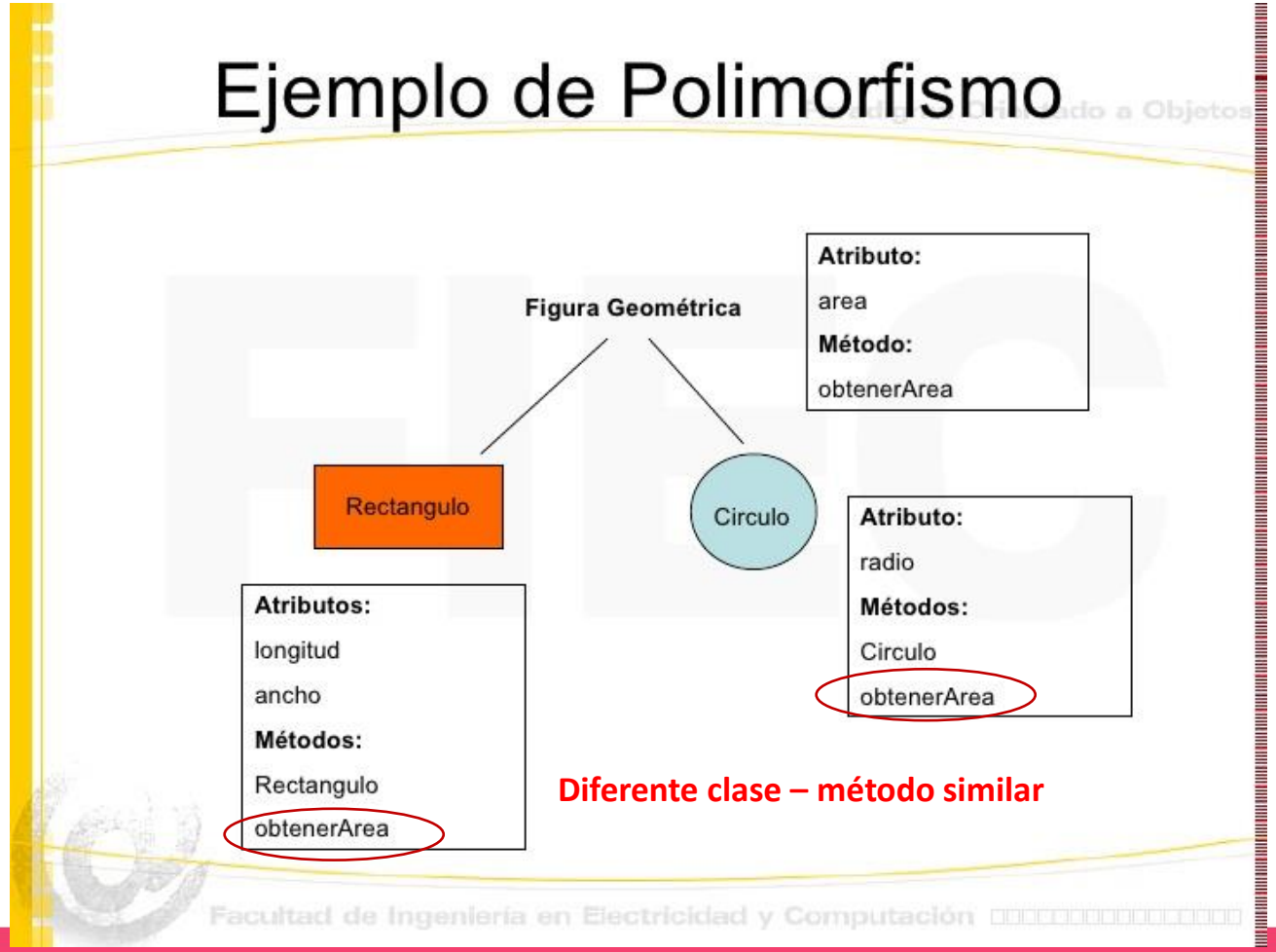
## Polimorfismo - Humanos



Diferente formas de ver...

# Programación Orientada a Objetos

## Polimorfismo - POO



# Programación Orientada a Objetos

# Ejercicio

## Ejemplo: Polimorfismo

**Dada las figuras geométricas cuadrado, rectángulo y círculo, de las cuales se conoce:**

**Cuadrado:**
- **Lado**

**Rectángulo:**
- **Base**
- **Altura**

**Círculo:**
- **radio**

**Se pide calcular para cada una de las figuras, el área y el perímetro.**

**Realizar el programa en Java que resuelva la situación problema presentada, utilizando el concepto de Clases y Objetos (POO) y aplicar el concepto de Polimorfismo.**

# Programación Orientada a Objetos

# Ejercicios

## Clases en StarUML

| Cuadrado |
| --- |
| +lado |
| +setLado(lado)<br>+getLado()<br>+area()<br>+perimetro() |

| Rectangulo |
| --- |
| +base<br>+altura |
| +setBase(base)<br>+getBase()<br>+setAltura(altura)<br>+getAltura()<br>+area()<br>+perimetro() |

| Circulo |
| --- |
| +radio |
| +setRadio(radio)<br>+getRadio()<br>+area()<br>+perimetro() |

## Programación Orientada a Objetos

# Ejercicios

### Método 1

Parámetros de entrada

**Cuadrado=>lado**

**Rectángulo=>base, altura**

**Círculo=>radio**

**Area()**

**Cuadrado=>lado*lado**

**Rectángulo=> base*altura**

**Círculo=> PI*radio^2**

Parámetros de salida

**área**

**FUNCIÓN retorna o regresa un solo valor**

## Programación Orientada a Objetos

# Ejercicios

### Método 2

Parámetros de entrada

**Cuadrado=>lado**

**Rectángulo=>base, altura**

**Círculo=>radio**

**Perimetro()**

**Cuadrado=> 4*lado**

**Rectángulo=> 2*base+2*altura**

**Círculo=> 2*PI*radio**

Parámetros de salida

**perimetro**

**FUNCIÓN retorna o regresa un solo valor**

# Programación Orientada a Objetos

# Ejercicios

## Ejercicio: (Clase 1)

```
public class Cuadrado {
    private double lado;

    public Cuadrado() {
    }
    public Cuadrado(double lado) {
        this.lado = lado;
    }
    public void setLado(double lado) {
        this.lado = lado;
    }
    public double getLado() {
        return lado;
    }
    public double area(){
        double a;
        a=Math.pow(this.lado, 2);
        return a;
    }
    public double perimetro(){
        double p;
        p=4*this.lado;
        return p;
    }
```

# Programación Orientada a Objetos

# Ejercicios

## Ejercicio: (Clase 2)

```
public class Rectangulo {
    private double base;
    private double altura;

    public Rectangulo() {
    }
    public Rectangulo(double base, double altura) {
        this.base = base;
        this.altura = altura;
    }
    public void setBase(double base) {
        this.base = base;
    }
    public void setAltura(double altura) {
        this.altura = altura;
    }
    public double getBase() {
        return base;
    }
    public double getAltura() {
        return altura;
    }
```

```
public double area(){
    double a;
    a=this.base*this.altura;
    return a;
}
public double perimetro(){
    double p;
    p=2*this.base+2*this.altura;
    return p;
}
```

# Programación Orientada a Objetos

# Ejercicios

## Ejercicio: (Clase 3)

```java
public class Circulo {
    private double radio;

    public Circulo() {
    }
    public Circulo(double radio) {
        this.radio = radio;
    }
    public double getRadio() {
        return radio;
    }
    public void setRadio(double radio) {
        this.radio = radio;
    }
    public double area(){
        double a;
        a=Math.PI*Math.pow(this.radio, 2);
        return a;
    }
    public double perimetro(){
        double p;
        p=2*Math.PI*this.radio;
        return p;
    }
}
```

# Programación Orientada a Objetos

# Ejercicios

## Ejercicio: (Main)

```
public static void main(String[] args) {
    // Definir consola
    Scanner consola=new Scanner(System.in);
    double lado,base,altura,radio,area,perimetro;
    int opcion=0;
    //Definición de variables objeto
    Cuadrado obj_cuad;
    Rectangulo obj_rect;
    Circulo obj_circ;
    do {
        System.out.println("    MENU DE OPCIONES");
        System.out.println();
        System.out.println("1. Figura Cuadrado");
        System.out.println("2. Figura Rectángulo");
        System.out.println("3. Figura Círculo");
        System.out.println("4. Salir");
        System.out.println("Ingrese Opción");
        opcion=consola.nextInt();
        switch(opcion){
            case 1:{
                System.out.println("Lado: ");
                lado=consola.nextDouble();
                obj_cuad=new Cuadrado(lado);
                area=obj_cuad.area();
                perimetro=obj_cuad.perimetro();
                System.out.println("Area Cuadrado: "+area);
                System.out.println("Perímetro Cuadrado: "+perimetro);
                break;
            }
```

# Programación Orientada a Objetos

# Ejercicios

## Ejercicio: (Main)

```
            case 2:{
                System.out.println("Base: ");
                base=consola.nextDouble();
                System.out.println("Altura: ");
                altura=consola.nextDouble();
                obj_rect=new Rectangulo(base,altura);
                area=obj_rect.area();
                perimetro=obj_rect.perimetro();
                System.out.println("Area Rectángulo: "+area);
                System.out.println("Perímetro Rectángulo: "+perimetro);
                break;
            }
            case 3:{
                System.out.println("Radio: ");
                radio=consola.nextDouble();
                obj_circ=new Circulo(radio);
                area=obj_circ.area();
                perimetro=obj_circ.perimetro();
                System.out.println("Area Círculo: "+area);
                System.out.println("Perímetro Círculo: "+perimetro);
                break;
            }
            case 4: break;
        }

    } while (opcion!=4);
}
```

Semana 3 Sesión 2

# Programación Orientada a Objetos

# Ejercicio

Ejemplo: Herencia - Polimorfismo

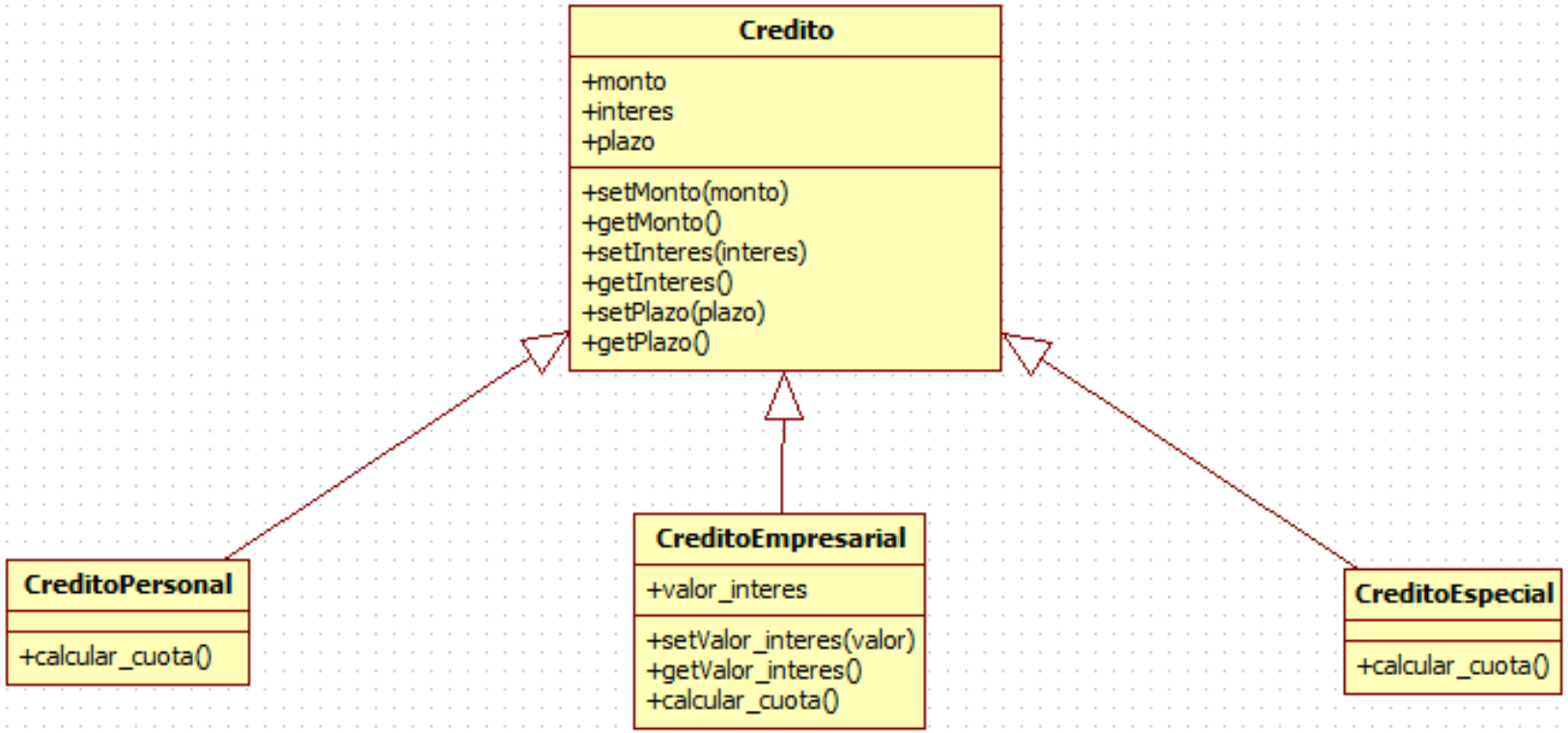
En un sistema bancario, los créditos manejan tres características básicas que son, el monto del crédito, porcentaje de interés y plazo. Sin embargo, se generan varias modalidades de crédito como son:

- Crédito Personal, en el cual el valor a pagar en cada cuota se genera como el monto del crédito + valor interés (porcentaje interés aplicado al monto) dividido sobre el plazo.
- Crédito Empresarial, en el cual se negocia un valor de interés total y la cuota es el monto del crédito + valor del interés negociado, dividido entre el plazo.
- Crédito Especial, en el cual el valor de la cuota es el monto del crédito dividido entre el plazo. (No se aplica interés)

Realizar el programa en Java que resuelva la situación problema presentada, utilizando el concepto de Clases y Objetos (POO) y aplicar el concepto de Herencia y Polimorfismo.

# Ejercicios

## Clase en StarUML



## Programación Orientada a Objetos

# Ejercicios

### Método Clase: CreditoPersonal

Parámetros de entrada

**Monto, interés, plazo**

**Calcular_cuota()**

**Cuota=(monto+monto*(interés/100))/plazo**

Parámetros de salida

**cuota**

**FUNCIÓN retorna o regresa un solo valor**

# Ejercicios

## Método Clase: CreditoEmpresarial

Parámetros de entrada

**Monto, valor_interés, plazo**

**calcular_cuota()**

**Cuota=(monto + valor_interés)/plazo**

Parámetros de salida

**cuota**

**FUNCIÓN retorna o regresa un solo valor**

# Ejercicios

## Método Clase: CreditoEspecial

Parámetros de entrada

**Monto, plazo**

**calcular_cuota()**

**Cuota=monto/plazo**

Parámetros de salida

**cuota**

**FUNCIÓN retorna o regresa un solo valor**

# Programación Orientada a Objetos

# Ejercicios

## Ejercicio: (Clase 1) Superclase

```java
public class Credito {
    private double monto;
    private double interes;
    private int plazo;

    public Credito() {
    }
    public Credito(double monto, double interes, int plazo) {
        this.monto = monto;
        this.interes = interes;
        this.plazo = plazo;
    }
    public void setMonto(double monto) {
        this.monto = monto;
    }
    public void setInteres(double interes) {
        this.interes = interes;
    }
    public void setPlazo(int plazo) {
        this.plazo = plazo;
    }
    public double getMonto() {
        return monto;
    }
    public double getInteres() {
        return interes;
    }
    public int getPlazo() {
        return plazo;
    }
```

# Programación Orientada a Objetos

# Ejercicios

## Ejercicio: (Clase 2)

```java
public class CreditoPersonal extends Credito {

    public CreditoPersonal() {
    }

    public CreditoPersonal(double monto, double interes, int plazo) {
        super(monto, interes, plazo);
    }

    public double calcular_cuota(){
        double cuota;
        cuota=(this.getMonto()+this.getMonto()*(this.getInteres()/100))/this.getPlazo();
        return cuota;
    }
}
```

# Programación Orientada a Objetos

# Ejercicios

## Ejercicio: (Clase 3)

```java
public class CreditoEmpresarial extends Credito {
    private double valor_interes;

    public CreditoEmpresarial() {
    }

    public CreditoEmpresarial(double monto, double interes, int plazo) {
        super(monto, interes, plazo);
    }

    public CreditoEmpresarial(double valor_interes, double monto, double interes, int plazo) {
        super(monto, interes, plazo);
        this.valor_interes = valor_interes;
    }

    public void setValor_interes(double valor_interes) {
        this.valor_interes = valor_interes;
    }
    public double getValor_interes() {
        return valor_interes;
    }
    public double calcular_cuota(){
        double cuota;
        cuota=(this.getMonto()+this.valor_interes)/this.getPlazo();
        return cuota;
    }
}
```

# Programación Orientada a Objetos

# Ejercicios

## Ejercicio: (Clase 4)

```java
public class CreditoPersonal extends Credito {

    public CreditoPersonal() {
    }

    public CreditoPersonal(double monto, double interes, int plazo) {
        super(monto, interes, plazo);
    }

    public double calcular_cuota(){
        double cuota;
        cuota=(this.getMonto()+this.getMonto()*(this.getInteres()/100))/this.getPlazo();
        return cuota;
    }
}
```

# Programación Orientada a Objetos

# Ejercicios

## Ejercicio: (Main)

```
import java.util.Scanner;


public class Herencia_creditos_G21 {


    public static void main(String[] args) {
        // Definición de la consola
        Scanner consola=new Scanner(System.in);
        // Definición de variables
        double monto,interes, valor_interes,cuota;
        int plazo, opcion=0;
        // Definición de las variables objeto
        CreditoPersonal obj_credper;
        CreditoEmpresarial obj_credemp;
        CreditoEspecial obj_credesp;

        do {
            System.out.println("     MENU CREDITOS");
            System.out.println("1. Crédito Personal");
            System.out.println("2. Crédito Empresarial");
            System.out.println("3. Crédito Especial");
            System.out.println("4. Salir");
            System.out.println("Ingrese Opción: ");
            opcion=consola.nextInt();
            switch(opcion){
```

```
case 1:{
            System.out.println("Monto: ");
            monto=consola.nextDouble();
            System.out.println("(%) Interés: ");
            interes=consola.nextDouble();
            System.out.println("Plazo: ");
            plazo=consola.nextInt();
            //Creación del objeto Crédito Personal
            obj_credper=new CreditoPersonal(monto,interes,plazo);
            cuota=obj_credper.calcular_cuota();
            System.out.println("Valor Cuota Crédito Personal: "+cuota);
            break;
        }
        case 2:{
            System.out.println("Monto: ");
            monto=consola.nextDouble();
            System.out.println("Valor interés: ");
            valor_interes=consola.nextDouble();
            System.out.println("Plazo: ");
            plazo=consola.nextInt();
            //Creación del objeto Crédito Empresarial
            obj_credemp=new CreditoEmpresarial(valor_interes,monto,0,plazo);
            cuota=obj_credemp.calcular_cuota();
            System.out.println("Valor Cuota Crédito Empresarial: "+cuota);
            break;
        }
```

# Programación Orientada a Objetos

## Ejercicios

### Ejercicio: (Main)

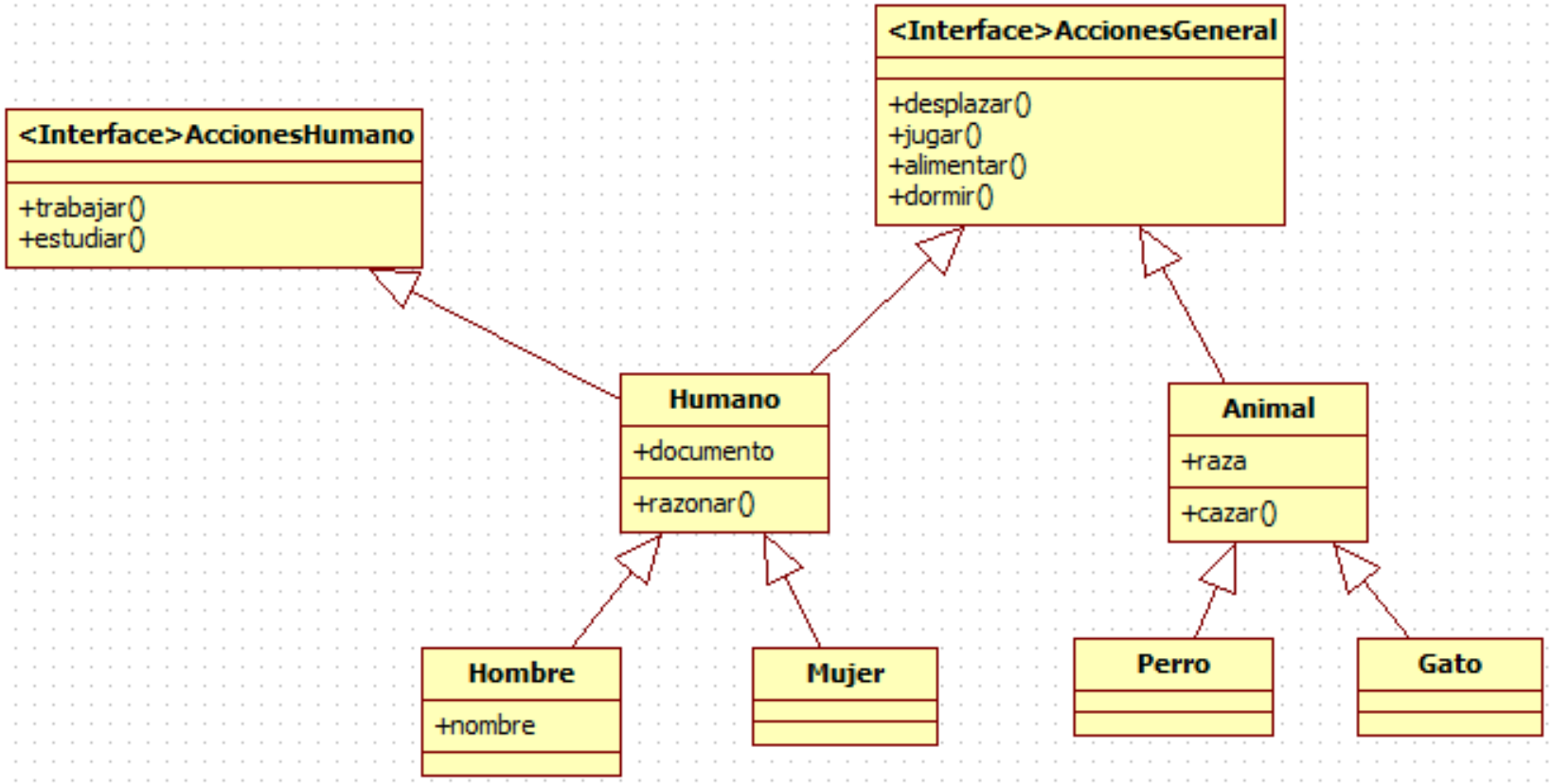
```
                case 3:{
                    System.out.println("Monto: ");
                    monto=consola.nextDouble();
                    System.out.println("Plazo: ");
                    plazo=consola.nextInt();
                    //Creación del objeto Crédito especial
                    obj_credesp=new CreditoEspecial(monto,0,plazo);
                    cuota=obj_credesp.calcular_cuota();
                    System.out.println("Valor Cuota Crédito Especial: "+cuota);
                    break;
                }
                case 4:break;
            }

        } while (opcion!=4);

}
```

## Interfaces (Métodos Abstractas)

Una **interfaz en Java** es una colección de métodos abstractos y propiedades constantes. En las **interfaces** se especifica qué se debe hacer pero no su implementación. Serán las clases que implementen estas **interfaces** las que describen la lógica del comportamiento de los métodos.

# Programación Orientada a Objetos

## Ejemplo – Interfaces



# Programación Orientada a Objetos

# Ejercicios

## Interface_1

```
public interface AccionesGeneral {
    public void desplazar();
    public void jugar();
    public void alimentar();
    public void dormir();

}
```

# Programación Orientada a Objetos

## Interface_2

# Ejercicios

```
public interface AccionesHumano {
    public void trabajar();
    public void estudiar();
}
```

# Programación Orientada a Objetos

## Clases (SuperClases)

# Ejercicios

```java
public class Humano implements AccionesGeneral,AccionesHumano {
    private String documento;

    public void setDocumento(String documento) {
        this.documento = documento;
    }

    public String getDocumento() {
        return documento;
    }
    public void razonar(){
        System.out.println("El Humano con documento "+this.documento+" Está razonando");
    }

    @Override
    public void desplazar() {
        System.out.println("El Humano con documento "+this.documento+" se está desplazando");
    }

    @Override
    public void jugar() {
        System.out.println("El Humano con documento "+this.documento+" Está jugando");
    }

    @Override
    public void alimentar() {
        System.out.println("El Humano con documento "+this.documento+" se está alimentando");
    }

    @Override
    public void dormir() {

    }

    @Override
    public void trabajar() {
        System.out.println("El Humano con documento "+this.documento+" está trabajando");
    }

    @Override
    public void estudiar() {

    }
}
```

# Programación Orientada a Objetos

## Clases (SuperClases)

# Ejercicios

```
public class Animal implements AccionesGeneral {
    private String raza;

    public void setRaza(String raza) {
        this.raza = raza;
    }

    public String getRaza() {
        return raza;
    }
    public void cazar(){
        System.out.println("El Animal de raza "+this.raza+" está cazando");
    }

    @Override
    public void desplazar() {
        System.out.println("El Animal de raza "+this.raza+" se está desplazando");
    }

    @Override
    public void jugar() {
        System.out.println("El Animal de raza "+this.raza+" está jugando");
    }

    @Override
    public void alimentar() {

    }

    @Override
    public void dormir() {

    }
}
```

# Programación Orientada a Objetos

## Clases (Subclases)

# Ejercicios

```java
public class Hombre extends Humano {
    private String nombre;

    public void setNombre(String nombre) {
        this.nombre = nombre;
    }

    public String getNombre() {
        return nombre;
    }

}
```

# Programación Orientada a Objetos

# Ejercicios

## Main

```
package interfaces_g21;

import java.util.Scanner;


public class Interfaces_G21 {


    public static void main(String[] args) {
        // Definición de consola
        Scanner consola=new Scanner(System.in);
        //Definición de variables objetos
        Humano obj_humano;
        Animal obj_animal;
        Hombre obj_hombre;
        //Creación del objeto Humano
        obj_humano=new Humano();
        obj_humano.setDocumento("91254478");
        obj_humano.razonar();
        obj_humano.desplazar();
        obj_humano.alimentar();
        obj_humano.jugar();
        obj_humano.trabajar();
        System.out.println();
```

# Programación Orientada a Objetos

# Ejercicios

## Main

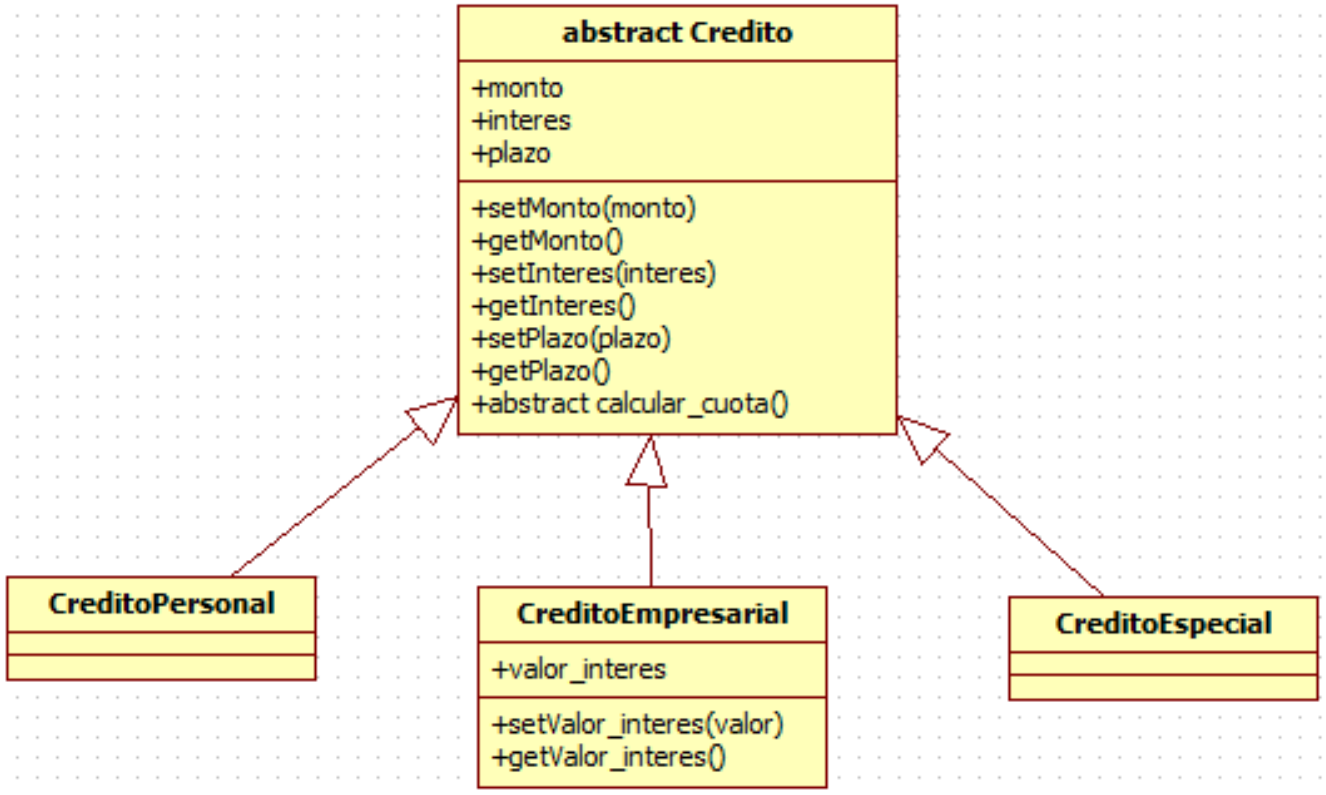
```
        //Creación del objeto Animal
        obj_animal=new Animal();
        obj_animal.setRaza("Orangutan");
        obj_animal.cazar();
        obj_animal.desplazar();
        obj_animal.jugar();
        System.out.println();
        //Creación del objeto Hombre
        obj_hombre=new Hombre();
        obj_hombre.setDocumento("91254478");
        obj_hombre.setNombre("Sergio Medina");
        obj_hombre.razonar();
        obj_hombre.desplazar();
        obj_hombre.trabajar();
    }

}
```

## Clases Abstractas

Una **clase abstracta** no es más que una **clase** común la cual posee atributos, métodos, constructores y por lo menos un método abstracto. Una **clase abstracta** no puede ser instanciada, solo heredada

# Programación Orientada a Objetos

## Ejemplo – Clase Abstracta



# Programación Orientada a Objetos

# Ejercicios

## Clase (Abstracta)

```
public abstract class Credito {
    private double monto;
    private double interes;
    private int plazo;

    public Credito() {
    }
    public Credito(double monto, double interes, int plazo) {
        this.monto = monto;
        this.interes = interes;
        this.plazo = plazo;
    }
    public double getMonto() {
        return monto;
    }
    public void setMonto(double monto) {
        this.monto = monto;
    }
    public double getInteres() {
        return interes;
    }
    public void setInteres(double interes) {
        this.interes = interes;
    }
    public int getPlazo() {
        return plazo;
    }
    public void setPlazo(int plazo) {
        this.plazo = plazo;
    }
    //Declaración del método abstracto
    public abstract double calcular_cuota();
}
```

# Programación Orientada a Objetos

## Clase

# Ejercicios

```java
public class CreditoPersonal extends Credito {

    public CreditoPersonal() {
    }

    public CreditoPersonal(double monto, double interes, int plazo) {
        super(monto, interes, plazo);
    }
    @Override
    public double calcular_cuota() {
        double cuota;
        cuota=(this.getMonto()+(this.getMonto()*this.getInteres()/100))/this.getPlazo();
        return cuota;
    }

}
```

# Programación Orientada a Objetos

## Clase

# Ejercicios

```
public class CreditoEmpresarial extends Credito {
    private double valor_interes_total;

    public CreditoEmpresarial() {
    }

    public CreditoEmpresarial(double valor_interes_total, double monto, double interes, int plazo) {
        super(monto, interes, plazo);
        this.valor_interes_total = valor_interes_total;
    }
    public double getValor_interes_total() {
        return valor_interes_total;
    }
    public void setValor_interes_total(double valor_interes_total) {
        this.valor_interes_total = valor_interes_total;
    }
    @Override
    public double calcular_cuota() {
        double cuota;
        cuota=(this.getMonto()+this.valor_interes_total)/this.getPlazo();
        return cuota;
    }

}
```

# Programación Orientada a Objetos

## Clase

# Ejercicios

```java
public class CreditoEspecial extends Credito {

    public CreditoEspecial() {
    }

    public CreditoEspecial(double monto, double interes, int plazo) {
        super(monto, interes, plazo);
    }
    @Override
    public double calcular_cuota() {
        double cuota;
       cuota=this.getMonto()/this.getPlazo();
       return cuota;
    }

}
```

# Programación Orientada a Objetos

# Ejercicios

## Ejercicio: (Main)

```
package herencia_creditos_g21;

import java.util.Scanner;

/**
 *
 * @author SERGIO
 */
public class Herencia_creditos_G21 {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // Definición de la consola
        Scanner consola=new Scanner(System.in);
        // Definición de variables
        double monto,interes, valor_interes,cuota;
        int plazo, opcion=0;
        // Definición de las variables objeto
        CreditoPersonal obj_credper;
        CreditoEmpresarial obj_credemp;
        CreditoEspecial obj_credesp;

        do {
            System.out.println("     MENU CREDITOS");
            System.out.println("1. Crédito Personal");
            System.out.println("2. Crédito Empresarial");
            System.out.println("3. Crédito Especial");
            System.out.println("4. Salir");
            System.out.println("Ingrese Opción: ");
            opcion=consola.nextInt();
            switch(opcion){
                case 1:{
                    System.out.println("Monto: ");
                    monto=consola.nextDouble();
                    System.out.println("(%) Interés: ");
                    interes=consola.nextDouble();
                    System.out.println("Plazo: ");
                    plazo=consola.nextInt();
                    //Creación del objeto Crédito Personal
                    obj_credper=new CreditoPersonal();
                    obj_credper.setMonto(monto);
                    obj_credper.setInteres(interes);
                    obj_credper.setPlazo(plazo);
                    cuota=obj_credper.calcular_cuota();
                    System.out.println("Valor Cuota Crédito Personal: "+cuota);
                    break;
                }
```

# Programación Orientada a Objetos

## Ejercicios

### Ejercicio: (Main)

```
case 2:{
    System.out.println("Monto: ");
    monto=consola.nextDouble();
    System.out.println("Valor interés: ");
    valor_interes=consola.nextDouble();
    System.out.println("Plazo: ");
    plazo=consola.nextInt();
    //Creación del objeto Crédito Empresarial
    obj_credemp=new CreditoEmpresarial();
    obj_credemp.setMonto(monto);
    obj_credemp.setValor_interes(valor_interes);
    obj_credemp.setPlazo(plazo);
    cuota=obj_credemp.calcular_cuota();
    System.out.println("Valor Cuota Crédito Empresarial: "+cuota);
    break;
}
case 3:{
    System.out.println("Monto: ");
    monto=consola.nextDouble();
    System.out.println("Plazo: ");
    plazo=consola.nextInt();
    //Creación del objeto Crédito especial
    obj_credesp=new CreditoEspecial();
    obj_credesp.setMonto(monto);
    obj_credesp.setPlazo(plazo);
    cuota=obj_credesp.calcular_cuota();
    System.out.println("Valor Cuota Crédito Especial: "+cuota);
    break;
```

```
            }
            case 4:break;
        }

    } while (opcion!=4);

    }

}
```

El futuro digital es de todos

MinTIC

# Ciclo: Programación Básica

# Programación Orientada a Objetos (POO)

## Misión TIC 2022

unab
VIGILADA MINEDUCACIÓN

Misión TIC 2022