1. ¿Qué es un TAD?
Un TAD es un Tipo Abstracto de Datos, que define un conjunto de valores y un conjunto de operaciones sobre esos valores. Es una forma de encapsular datos y funciones que operan sobre ellos en una estructura.

2. ¿Dónde se produce el encapsulamiento?
El encapsulamiento se produce dentro de las clases en la POO.

3. ¿Cuáles son las semejanzas y las diferencias entre funciones, procedimientos y métodos? 4. ¿Qué es UML? ¿Y cómo se representa una clase en ese lenguaje?
Funciones: Son bloques de código que realizan una tarea específica y pueden devolver un valor.

Procedimientos: Son similares a las funciones, pero no devuelven ningún valor.

Métodos: Son funciones o procedimientos asociados a un objeto o clase en la programación orientada a objetos.

UML es un lenguaje de modelado utilizado para visualizar, especificar, construir y documentar artefactos de sistemas de software. Para representar una clase en UML, se utiliza un rectángulo dividido en tres compartimentos:

5. Marcar con cruz.

5. Marcar con cruz.

	No se aplica a clases	Solo se aplica a atributos	Solo se aplica a clases	Se aplica a atributos, métodos y clases
Public				X
Private				X
Protected				X
Static				X
Final		X		
Primera letra en minúscula	X			
Primera letra en mayúscula	X			

6. Verdadero o falso
- Un constructor…
o Es el método principal para ejecutar un programa. F
o Crea instancias. V
o Devuelve el valor de un atributo privado. F
o Tiene sentencia return. V
o Siempre existe uno por defecto, sin parámetros ni inicializaciones de atributos. V
o Se puede sobreescribir. V
o Se puede sobrecargar. V
o Su nombre se escribe con mayúscula. V
o Su calificador de acceso es static. F

o Su tipo de devolución no se indica y corresponde a la clase. F

- Un método…
o Puede tener múltiples parámetros con el mismo nombre, siempre y cuando tengan tipos diferentes.V
o Puede sobrecargarse. V
o Puede sobreescribirse. V
o Puede ser static. V
o Puede ser tanto public como protected, pero no private. F
o Un método puede tener un modificador de acceso final. V

7. Calificadores de acceso. Completa.

a- Se necesita que cualquiera pueda acceder al color de un vehículo. Entonces, declaro color como: **public**.
b- Se necesita que color se pueda acceder a través no sólo de vehículo, sino ahora también de Buses. Entonces, declaro color como: **protected**.
a- Se necesita que color se pueda acceder solamente para vehículo. Entonces, declaro color como: **private**.

8. Se desea desarrollar un sistema de gestión de empleados para una empresa. El sistema debe permitir registrar empleados de dos tipos diferentes: gerentes y trabajadores. Cada empleado debe tener un nombre, una edad y un salario. Los gerentes tienen la capacidad de organizar actividades dentro de un departamento específico, mientras que los trabajadores están encargados de producir en un área determinada.

Implementa un sistema que modele esta situación utilizando herencia en Java. Define una clase base llamada Empleado que contenga los atributos y métodos comunes para todos los empleados, como el nombre, la edad, el salario y la capacidad de trabajar. Luego, crea dos subclases: Gerente y Trabajador, que hereden de Empleado. Los gerentes deben tener un atributo adicional para almacenar el departamento en el que trabajan, así como un método para organizar actividades dentro de ese departamento. Por otro lado, los trabajadores deben tener un atributo para indicar el área en la que trabajan y un método para producir en esa área.

Finalmente, en el programa principal, crea instancias de ambas subclases y muestra cómo se utilizan los métodos específicos de cada tipo de empleado, así como sus atributos.

```
class Employee {
    private String name;
    private int age;
    private double salary;
    public Employee(String name, int age, double salary) {
        this.name = name;
        this.age = age;
        this.salary = salary;
    }
    public String getName() {
        return name;
    }
    public int getAge() {
        return age;
    }
```

```
    public double getSalary() {
        return salary;
    }
    public void work() {
        System.out.println(name + " está trabajando.");
    }
}
class Manager extends Employee {
    private String department;
    public Manager(String name, int age, double salary, String department) {
        super(name, age, salary);
        this.department = department;
    }
    public void organizeActivities() {
        System.out.println(getName() + " está organizando actividades en el departamento de " +
department + ".");
    }
}
class Worker extends Employee {
    private String area;
    public Worker(String name, int age, double salary, String area) {
        super(name, age, salary);
        this.area = area;
    }
    public void produce() {
        System.out.println(getName() + " está produciendo en el área de " + area + ".");
    }
}


public class Main {
    public static void main(String[] args) {

        Manager manager = new Manager("Juan", 40, 5000, "Recursos Humanos");
        Worker worker = new Worker("María", 25, 3000, "Producción");
        manager.organizeActivities();
        worker.produce();
        System.out.println("Name of the manager: " + manager.getName());
        System.out.println("Age of the worker: " + worker.getAge());
        System.out.println("Salary of the manager: " + manager.getSalary());
        manager.work();
    }
}
```

9. Se desea implementar un programa en Java para modelar diferentes figuras geométricas, como círculos y rectángulos. Cada figura geométrica debe tener la capacidad de calcular su área y su perímetro.

Define una clase base llamada FiguraGeometrica que contenga métodos abstractos para calcular el área y el perímetro de la figura. Luego, crea subclases para representar diferentes tipos de figuras geométricas, como Circulo y Rectángulo, que hereden de la clase base FiguraGeometrica. En la subclase Círculo, implementa métodos para calcular el área y el perímetro de un círculo, utilizando

el radio como atributo de la clase. En la subclase Rectángulo, implementa métodos para calcular el área y el perímetro de un rectángulo, utilizando la longitud y la anchura como atributos de la clase.

En el programa principal, crea instancias de diferentes figuras geométricas (al menos un círculo y un rectángulo) y muestra sus áreas y perímetros.

```
abstract class GeometricFigure {
    public abstract double calculateArea();
    public abstract double calculatePerimeter();
}
class Circle extends GeometricFigure {
    private double radius;

    public Circle(double radius) {
        this.radius = radius;
    }

    @Override
    public double calculateArea() {
        return Math.PI * radius * radius;
    }

    @Override
    public double calculatePerimeter() {
        return 2 * Math.PI * radius;
    }
}

class Rectangle extends GeometricFigure {
    private double length;
    private double width;

    public Rectangle(double length, double width) {
        this.length = length;
        this.width = width;
    }

    @Override
    public double calculateArea() {
        return length * width;
    }

    @Override
    public double calculatePerimeter() {
        return 2 * (length + width);
    }
}
public class Main {
    public static void main(String[] args) {
        Circle circle = new Circle(5);
        Rectangle rectangle = new Rectangle(4, 6);
        System.out.println("Círculo:");
```

```
        System.out.println("Área: " + circle.calculateArea());
        System.out.println("Perímetro: " + circle.calculatePerimeter());
        System.out.println("\nRectángulo:");
        System.out.println("Área: " + rectangle.calculateArea());
        System.out.println("Perímetro: " + rectangle.calculatePerimeter());
    }
}
```

10. Definir una clase Libro para manejar la información asociada a un libro. La información de interés para un libro es: el título, el autor y el precio.
Los métodos de interés son:
o Un constructor para crear un objeto libro, con título y autor como parámetros.
o Imprimir en pantalla el título, los autores y el precio del libro.
o Métodos get y set para cada atributo de un libro.

Se debe extender la clase Libro definiendo las siguientes clases:
o Libros de texto con un nuevo atributo que especifica el curso al cual está asociado el libro.
o Libros de texto de la Universidad Nacional de Colombia: subclase de la clase anterior. Esta subclase tiene un atributo que especifica cuál facultad lo publicó.
o Novelas: pueden ser de diferente tipo, histórica, romántica, policíaca, realista, ciencia ficción o aventuras.

Para cada una de las clases anteriores se debe definir su constructor y redefinir adecuadamente el método para visualizar del objeto.

```
class Book {
    private String title;
    private String author;
    private double price;

    public Book(String title, String author) {
        this.title = title;
        this.author = author;
    }

    public String getTitle() {
        return title;
    }

    public void setTitle(String title) {
        this.title = title;
    }

    public String getAuthor() {
        return author;
    }

    public void setAuthor(String author) {
        this.author = author;
    }

    public double getPrice() {
        return price;
```

```
    }

    public void setPrice(double price) {
        this.price = price;
    }

    public void mostrarDetalles() {
        System.out.println("Título: " + title);
        System.out.println("Autor: " + author);
        System.out.println("Precio: $" + price);
    }
}

class TextBook extends Book {
    private String course;

    public TextBook(String title, String author, String course) {
        super(title, author);
        this.course = course;
    }

    @Override
    public void mostrarDetalles() {
        super.mostrarDetalles();
        System.out.println("Curso: " + course);
    }
}

class UNALTextBook extends TextBook {
    private String faculty;

    public UNALTextBook(String title, String author, String course, String faculty) {
        super(title, author, course);
        this.faculty = faculty;
    }

    @Override
    public void mostrarDetalles() {
        super.mostrarDetalles();
        System.out.println("Facultad: " + faculty);
    }
}


class Novel extends Book {
    private String genre;

    public Novel(String title, String author, String genre) {
        super(title, author);
        this.genre = genre;
    }
```

```
    @Override
    public void mostrarDetalles() {
        super.mostrarDetalles();
        System.out.println("Género: " + genre);
    }
}


public class Main {
    public static void main(String[] args) {
        Book libro = new Book("El Arte de la Guerra", "Sun Tzu");
        libro.setPrice(15.99);
        libro.mostrarDetalles();
        System.out.println();

        TextBook libroTexto = new TextBook("Introducción a la Programación", "John Doe",
"Informática");
        libroTexto.setPrice(29.99);
        libroTexto.mostrarDetalles();
        System.out.println();

        UNALTextBook libroUNAL = new UNALTextBook("Álgebra Lineal", "Jane Smith",
"Matemáticas", "Facultad de Ciencias Naturales y Matemáticas");
        libroUNAL.setPrice(39.99);
        libroUNAL.mostrarDetalles();
        System.out.println();

        Novel novela = new Novel("La Sombra del Viento", "Carlos Ruiz Zafón", "Misterio");
        novela.setPrice(24.99);
        novela.mostrarDetalles();
    }
}
```