Nombre del estudiante: Arley Ivan Salgado Mañay
Asignatura: Programación estructurada y funcional
Fecha: 15/06/2025
Título: Aprendizaje Autónomo 2

1 ¿Qué es UML?

Las siglas UML significan Lenguaje Unificado de Modelado, el cuál fue creado para que se tenga una imagen visual de la arquitectura de un diseño y su implementación en los sistemas de Software. Este lenguaje se lo puede comparar con planes utilizados en otros campos y este se basa en el uso de diagramas. Por lo general, dentro de este diagrama se tienen los límites, una estructura y cual va a ser el comportamiento del sistema. (*Qué es el lenguaje unificado de modelado (UML)*, s. f.)

1.1 Uso en programación orientada a objetos.

Se tiene el diagrama de clases, que es el utilizado en un sistema que esta orientado a objetos. Dentro de este diagrama se los divide como clases, atributos y funciones, haciendo que de esta manera se muestre la clase y la función que empeñas cada uno. (*Diagrama UML*, s. f.)

2 ¿Qué es un diagrama de clases?

Este diagrama muestra las clases que tienen los diferentes sistemas y la relación que mantienen entre ellas. Las clases tienen como función el crear y manejar los objetos, los cuales son las instancias de las clases. El diagrama de clases tiene como objetivo visualizar clases de los diferentes sistemas y sus características y también tiene como objeto analizar las relaciones que existen entre las clases. (*Diagrama de clases*, s. f.)

3 Elaboración de diagrama UML

3.1 Clases

- ***Departamento:*** Área interna de la empresa.
- ***Empleado:*** Persona que trabaja en la empresa.
- ***Proyecto:*** Trabajo a realizarse en la empresa.
- ***Dependiente:*** Persona que requiere un empreado.

3.2 Atributos por clase

3.2.1 Clase: Departamento

Atributos:

- nombre: String
- numero: Int
- ubicaciones: List<String>
- fechaInicioGerencia: Date

Relaciones:

- gerente: Empleado
- empleados: List<Empleado>

- proyectos: List<Proyecto>

3.2.2 Clase: Empleado

Atributos:

- nombre: String
- ssn: String
- direccion: String
- salario: Float
- genero: String
- fechaNacimiento: Date

Relaciones:

- departamento: Departamento
- supervisor: Empleado
- asignaciones: List<AsignacionProyecto>
- dependientes: List<Dependiente>

3.2.3 Clase: Proyecto

Atributos:

- nombre: String
- numero: Int
- ubicacion: String

Relaciones:

- departamento: Departamento

3.2.4 Clase: Dependiente

Atributos:

- nombre: String
- genero: String
- fechaNacimiento: Date
- relacion: String

Relaciones:

- empleado: Empleado

3.2.5 Clase: AsignaciónProyecto (clase de asociación)

Atributos:

- horasPorSemana: Float

Relaciones:

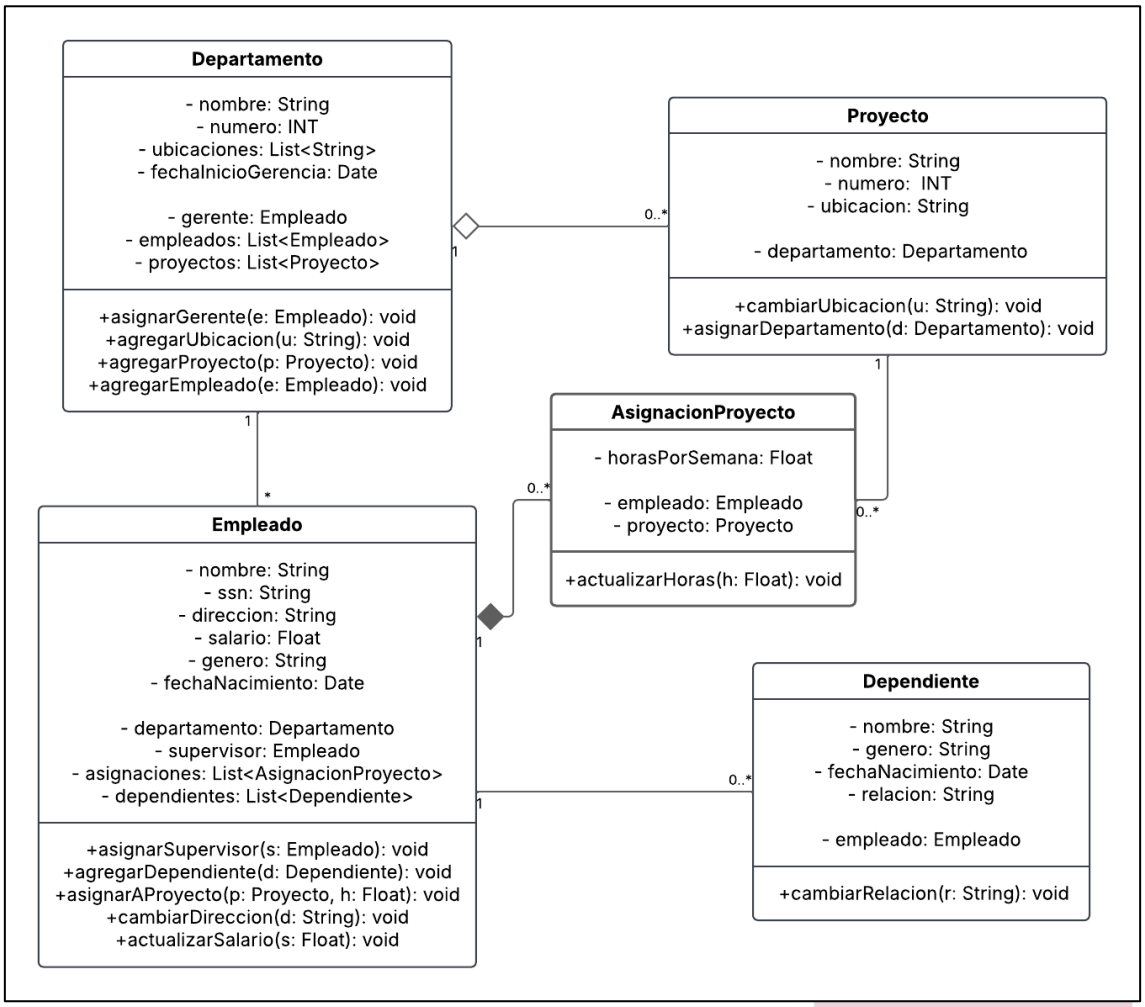
- empleado: Empleado
- proyecto: Proyecto

3.3 Relaciones entre clases

- ***Departamento →Empleado (gerente):*** Uno a uno.
- ***Departamento → Empleado (empleado):*** Uno a muchos.

- ***Empleado → Proyecto:*** Muchos a muchos.
- ***Empleado → Dependiente:*** Uno a muchos.
- ***Departamento → Proyecto:*** Uno a muchos.

3.4 Diagrama UML



Link: Diagrama UML

4 Ejecución

4.1 Clases

4.1.1 Circulo

Se crea una clase publica con el nombre de Circulo, a continuación, se crea un atributo privado de tipo INT que va a representar al radio del círculo. Dentro del constructor se recibe el valor del radio para después asignarlo dentro de la clase, por medio del comando *this.radio*se diferencia el parámetro del atributo. Finalmente se crea una clase publica en la que se devuelve el valor del área del círculo mediante la fórmula A= π*r*r y otra clase pública en la que se va a calcular el perímetro del radio por medio de la fórmula P= 2*π*r

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

    // Constructor
    public Circulo(int radio) {
        this.radio = radio;
    }

    // Método para calcular el área
    public double calcularArea() {
        return Math.PI * radio * radio;
    }

    // Método para calcular el perímetro
    public double calcularPerimetro() {
        return 2 * Math.PI * radio;
    }
}
```

4.1.2 Cuadrado

Se realiza casi el mismo proceso que en el círculo con la diferencia que se debe crear un atributo para representar al lado del cuadrado. Igualmente se recibe el valor del lado para colocarlo a la interna de la clase. Y al final se crean 2 clases públicas, una en la que devuelva el valor del área por medio de la fórmula A=lado*lado y la otra que devuelva el valor del perímetro con la fórmula P=4*lado.

```java
public class Cuadrado {
    private int lado;

    // Constructor
    public Cuadrado(int lado) {
        this.lado = lado;
    }

    // Método para calcular el área
    public double calcularArea() {
        return lado * lado;
    }

    // Método para calcular el perímetro
    public double calcularPerimetro() {
        return 4 * lado;
    }
}
```

4.1.3 Rectángulo

El proceso para construir la clase, se cran los atributos que conforman el rectángulo que son la base y la altura de este. En esta clase a diferencia de las anteriores va a recibir 2 valores las cuales van a pertenecer a la base y ala altura. Finalmente las clases publicas van a devolver el valor del área con la fórmula A=base*altura y el perímetro con la fórmula P=2*(base*altura).

```java
public class Rectangulo {
    private int base;
    private int altura;

    // Constructor
    public Rectangulo(int base, int altura) {
        this.base = base;
        this.altura = altura;
    }

    // Método para calcular el área
    public double calcularArea() {
        return base * altura;
    }

    // Método para calcular el perímetro
    public double calcularPerimetro() {
        return 2 * (base + altura);
    }
}
```

4.1.4 Triangulo Rectángulo

Esta clase comienza de la misma manera, se crean 2 atributos que tengan la base y la altura del triángulo. Por medio del comando *this.xxx* se recibe el parámetro de la clase principal. Se crean 2 clases públicas una para devolver el valor del área A=(base*altura)/2 y otra para devolver el valor del perímetro con la fórmula P=base+altura+hipotenusa.

A diferencia de las anteriores clases se debe calcular el valor de la hipotenusa y para esto se hace uso del teorema de Pitágoras haciendo que el mismo código devuelva ese valor por medio de la fórmula hipotenusa= √(base^2 + altura^2).

En este caso se debe determinar si el triángulo que se está evaluando es equilátero, isósceles o escaleno. Esto se logra por medio de un bucle *if* en el que se da un condicionamiento que si todos los lados son iguales es un triángulo *equilátero*, en el caso de que ningún lado sea igual sería un triángulo *isósceles* y en el caso de que no cumpla ninguno de estos condicionamientos va a ser un triángulo *isósceles*.

```java
public class TrianguloRectangulo {
    private int base;
    private int altura;

    // Constructor
    public TrianguloRectangulo(int base, int altura) {
        this.base = base;
        this.altura = altura;
    }

    // Método para calcular el área
    public double calcularArea() {
        return (base * altura) / 2.0;
    }

    // Método para calcular el perímetro (base + altura + hipotenusa)
    public double calcularPerimetro() {
        return base + altura + calcularHipotenusa();
    }

    // Método para calcular la hipotenusa
    public double calcularHipotenusa() {
        return Math.sqrt(base * base + altura * altura);
    }

    // Método para determinar el tipo de triángulo según sus lados
    public String determinarTipoTriangulo() {
        double hipotenusa = calcularHipotenusa();

        if (base == altura && altura == hipotenusa) {
            return "Equilátero";
        } else if (base == altura || altura == hipotenusa || base == hipotenusa) {
            return "Isósceles";
        } else {
            return "Escaleno";
        }
    }
}
```

4.1.5 Trapecio

Para poder realizar el mismo proceso con el trapecio se sigue el mismo lineamiento, pero en este caso se crean 5 atributos para que se tengan todos los datos del trapecio tales como: base mayor, base menor, altura, lado1 y lado 2. Al igual que en los anteriores códigos se debe recibir los parámetros dentro de la clase y finalmente construir las 2 clases públicas para devolver el valor de área con la fórmula A=((baseMayor + baseMenor)*altura)/2 y para devolver el perímetro por medio de la fórmula P=baseMayor+baseMenor+lado1+lado2.

Trapecio.java > Java > Trapecio > Trapecio(int baseMayor, int baseMenor, int altura, int lado1, int lado2)

```java
public class Trapecio {
    private int baseMayor;
    private int baseMenor;
    private int altura;
    private int lado1;
    private int lado2;

    //Constructor
    public Trapecio (int baseMayor, int baseMenor, int altura, int lado1, int lado2) {
        this.baseMayor = baseMayor;
        this.baseMenor = baseMenor;
        this.altura = altura;
        this.lado1 = lado1;
        this.lado2 = lado2;
    }

    // Método para calcular el área
    public double calcularArea() {
        return ((baseMayor + baseMenor) * altura) / 2.0;
    }

    //Metodo para calcular el perímetro
    public double calcularPerimetro() {
        return baseMayor + baseMenor + lado1 + lado2;
    }

}

```

4.1.6 Prueba Figuras

Se crea una clase pública con el método *main()* para que sea la parte principal del programa. A continuación, se crean 5 objetos en los que se va a tener los valores de las dimensiones de cada una de las figuras creadas en las anteriores clases, dentro del presente código se crear variables con el nombre de figura#. Se hace uso del comando *System.out.println* para que la terminal pueda imprimir el mensaje que el desarrollador desea junto con el calculo realizado dentro de cada clase en el caso de del área se usa *figura1.calcularArea()* y para el perímetro *figura1.calcularPerimetro()*. Este mismo proceso se lleva a cabo para cada una de las clases creadas con cada figura geométrica.

Para que dentro del terminal se muestre también el tipo de triangulo que se calculo se realiza el mismo proceso que el perímetro y área pero se utiliza *figura4.determinarTipoTriangulo()*.

```java
public class PruebaFiguras {
    Run main | Debug main | Run | Debug
    public static void main(String[] args) {

        Circulo figura1 =  new Circulo(radio:2);
        Rectangulo figura2 = new Rectangulo(base:1,altura:2);
        Cuadrado figura3 = new Cuadrado(lado:3);
        TrianguloRectangulo figura4 = new TrianguloRectangulo(base:3,altura:5);
        Trapecio figura5 = new Trapecio(baseMayor:4, baseMenor:3, altura:5, lado1:6, lado2:0);

        System.out.println("El área del círculo: " + figura1.calcularArea());
        System.out.println("El perímetro del círculo: " + figura1.calcularPerimetro());
        System.out.println();

        System.out.println("El área del rectángulo: " + figura2.calcularArea());
        System.out.println("El perímetro del rectángulo: " + figura2.calcularPerimetro());
        System.out.println();

        System.out.println("El área del cuadrado: " + figura3.calcularArea());
        System.out.println("El perímetro del cuadrado: " + figura3.calcularPerimetro());
        System.out.println();

        System.out.println("El área del triángulo: " + figura4.calcularArea());
        System.out.println("El perímetro del triángulo: " + figura4.calcularPerimetro());
        System.out.println("El triángulo es " + figura4.determinarTipoTriangulo());
        System.out.println();

        System.out.println("El área del tracepcio: "+ figura5.calcularArea());
        System.out.println("El perímetro del trapecio: " + figura5.calcularPerimetro());

    }
}
```

4.2 Resultado

Se muestra la ejecución de la clase PruebaFiguras:

```
El perímetro del círculo: 12.566370614359172

El área del rectángulo: 2.0
El perímetro del rectángulo: 6.0

El área del cuadrado: 9.0
El perímetro del cuadrado: 12.0

El área del triángulo: 7.5
El perímetro del triángulo: 13.8309518948453
El triángulo es Escaleno

El área del tracepcio: 17.5
El perímetro del trapecio: 13.0
```

5 GitHub

A continuación, se tiene el enlace de GitHub en el que se tienen los códigos creados y el trabajo desarrollado

https://github.com/arsalgadoma/Programaci-nEstructuradaYFuncional.git

6 Conclusiones

- El diagrama de clases UML permite tener una correcta planeación en cuanto al diseño, comunicación y mantenimiento del Software, permitiendo tener una representación estática de un sistema. Se la considera como un elemento importante para que la coherencia, escalabilidad y alineamiento del sistema sea garantizado.

7 Referencias

Diagrama de clases: Qué es, cómo hacerlo y ejemplos | Miro. (s. f.). https://miro.com/. Recuperado 14 de junio de 2025, de https://miro.com/es/diagrama/que-es-diagrama-clases-uml/

Diagrama UML: Qué es, cómo hacerlo y ejemplos | Miro. (s. f.). https://miro.com/. Recuperado 14 de junio de 2025, de https://miro.com/es/diagrama/que-es-diagrama-uml/

Qué es el lenguaje unificado de modelado (UML). (s. f.). Lucidchart. Recuperado 14 de junio de 2025, de https://www.lucidchart.com/pages/es/que-es-el-lenguaje-unificado-de-modelado-uml