Apunte Actividad 1

this, Constructores y Sobrecarga de Métodos

1. La palabra clave this

En Java, **this** es una referencia al **objeto actual** dentro de una clase.
Se utiliza principalmente para:

1. **Diferenciar atributos de parámetros con el mismo nombre**:

```java
public class Persona {

    private String nombre;

    public Persona(String nombre) {
        this.nombre = nombre; // "this.nombre" → atributo, "nombre" → parámetro
    }
}
```

2. **Llamar a otros constructores dentro de la misma clase**:

```java
public class Persona {

    private String nombre;
    private int edad;

    public Persona(String nombre) {
        this(nombre, 0); // Llama al otro constructor
    }

    public Persona(String nombre, int edad) {
        this.nombre = nombre;
        this.edad = edad;
    }
}
```

3. **Pasar el objeto actual como argumento**:

```java
class Boton {

    public void click() {
        Accion.ejecutar(this); // pasa el objeto actual
    }
}
```

2. Constructores en Java

Un **constructor** es un método especial que se ejecuta al crear un objeto con new.
Características:

- Tienen el mismo nombre que la clase.
- No tienen tipo de retorno.
- Se pueden **sobrecargar**.

Ejemplo:

```java
public class Persona {

    private String nombre;
    private int edad;

    // Constructor por defecto
    public Persona() {
        this.nombre = "Desconocido";
        this.edad = 0;
    }

    // Constructor con parámetros
    public Persona(String nombre, int edad) {
        this.nombre = nombre;
        this.edad = edad;
    }
}
```

3. Sobrecarga de Constructores

La **sobrecarga** consiste en definir varios constructores con distintos parámetros para ofrecer diferentes formas de crear objetos.

Ejemplo:

```java
public class Rectangulo {

    private int base;
    private int altura;

    // Constructor 1
    public Rectangulo() {
        this(1, 1); // usa valores por defecto
    }

    // Constructor 2
    public Rectangulo(int base) {
        this(base, 1);
    }

    // Constructor 3
    public Rectangulo(int base, int altura) {
        this.base = base;
        this.altura = altura;
    }
}
```

Uso:

```java
Rectangulo r1 = new Rectangulo();        // base=1, altura=1
Rectangulo r2 = new Rectangulo(5);       // base=5, altura=1
Rectangulo r3 = new Rectangulo(4, 6);    // base=4, altura=6
```

4. Sobrecarga de Métodos

También podemos sobrecargar **métodos**, siempre que cambien los **parámetros** (cantidad o tipo).

El retorno puede ser diferente, pero no basta solo con eso para diferenciar métodos.

Ejemplo:

```java
public class Calculadora {

    public int sumar(int a, int b) {
        return a + b;
    }

    public double sumar(double a, double b) {
        return a + b;
    }

    public int sumar(int a, int b, int c) {
        return a + b + c;
    }
}
```

Uso:

```java
Calculadora calc = new Calculadora();
System.out.println(calc.sumar(2, 3));         // 5
System.out.println(calc.sumar(2.5, 3.7));     // 6.2
System.out.println(calc.sumar(1, 2, 3));      // 6
```