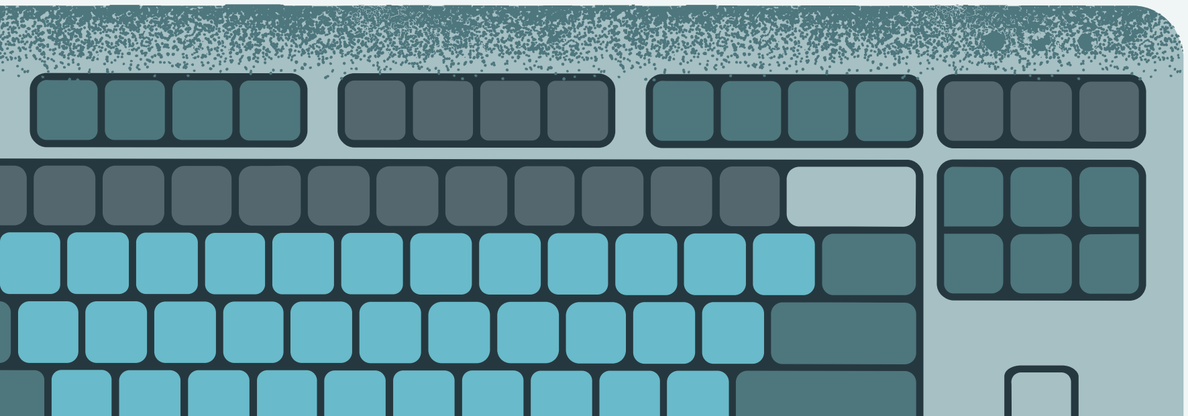
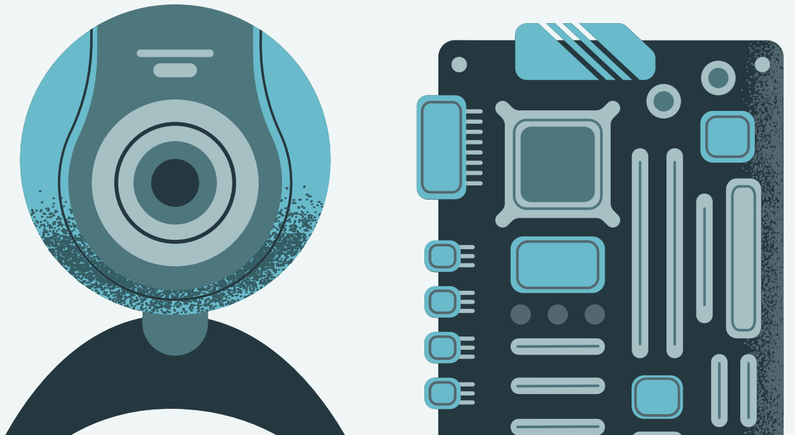
UNIVERSIDAD DE LAS FUERZAS ARMADAS "ESPE"

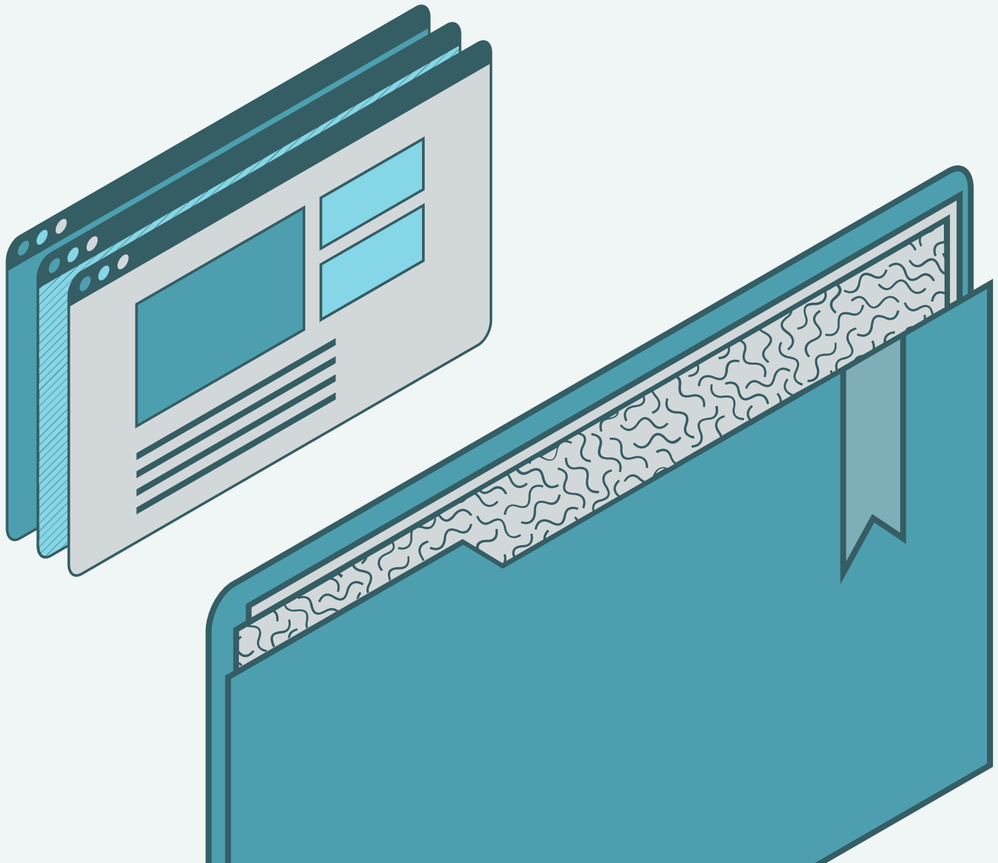
OBJECT-ORIENTED PROGRAMMING

NRC: 1940
DATE: 05/11/2024

1922
ECUADOR
ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

GENERAL PRINCIPLES OF OBJECT-ORIENTED PROGRAMMING



Group Members:

-CHASIPANTA IAN

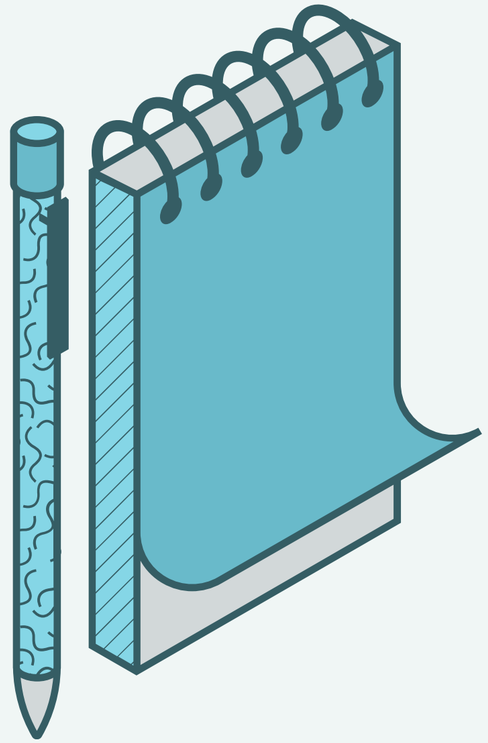
-CALVOPIÑA DAVID

-CHASIPANTA SAUL

-ROJAS JONATHAN

GENERAL PRINCIPLES OF OBJECT-ORIENTED PROGRAMMING

OBJECT-ORIENTED PROGRAMMING (OOP) IS A PARADIGM THAT ORGANIZES SOFTWARE DESIGN AROUND "OBJECTS" RATHER THAN ACTIONS OR FUNCTIONS. THE BASIC PRINCIPLES OF OOP MAKE IT EASIER TO CREATE MORE MODULAR, REUSABLE, AND MAINTAINABLE APPLICATIONS. THE ESSENTIAL CONCEPTS ARE DESCRIBED BELOW:



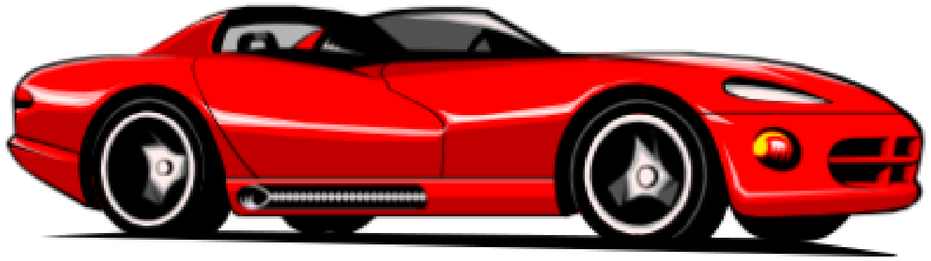
1. CLASSES
-A CLASS IS A TEMPLATE OR MODEL THAT DEFINES A SET OF ATTRIBUTES AND METHODS THAT CHARACTERIZE A PARTICULAR TYPE OF OBJECT.

-CLASSES HELP STRUCTURE THE CODE, SPECIFYING HOW OBJECTS BASED ON THEM SHOULD BEHAVE.

-FOR EXAMPLE, THE VEHICLE CLASS SERVES AS A GENERAL MODEL FOR ANY TYPE OF VEHICLE.

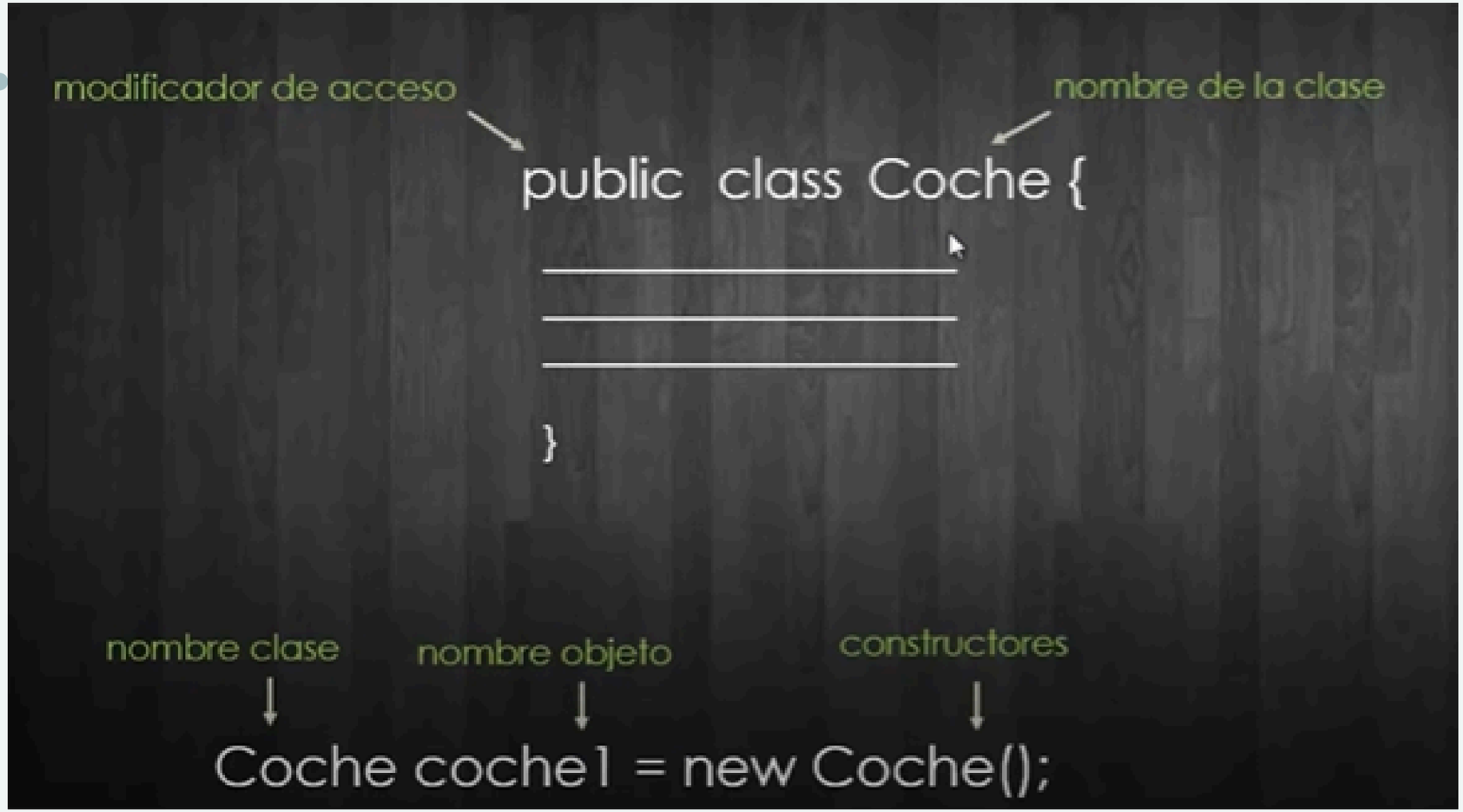
OBJECT

An object, also known as an instance of a class, is the concrete and specific representation of that class that resides in the computer's memory.



- Atributos:
  - color
  - velocidad
  - ruedas
  - motor
- Métodos:
  - arranca()
  - frena()
  - dobla()

CREATING CLASSES AND OBJECTS

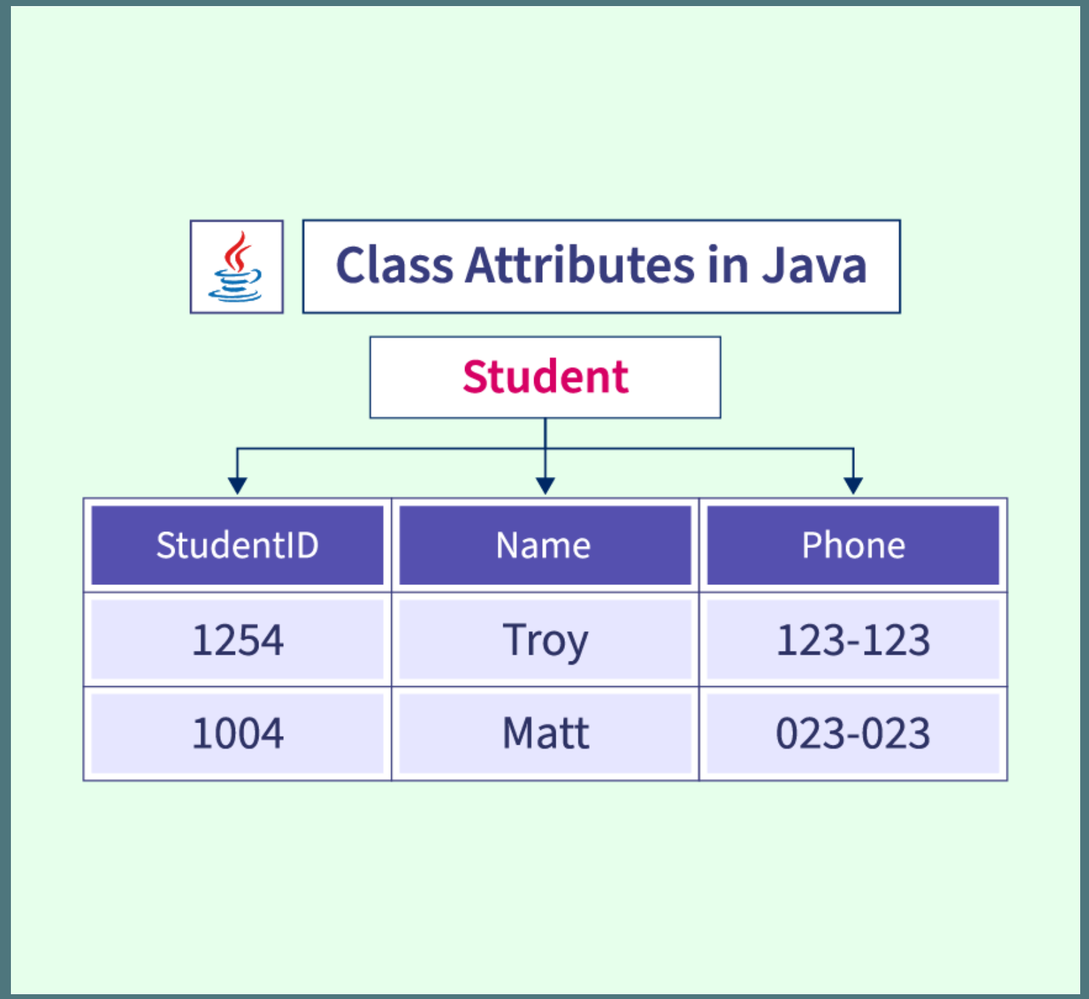


ATTRIBUTES

- Attributes are data members inside a class or object.
- They represent the features or characteristics of the class.
- Example: A car class might include attributes like:
- Tire, door, seats, license plate, headlight, wheel, handle, make, year.
- Defining attributes keeps the code simple and maintainable.

Attributes

>>> Door
>>> Seats
>>> Licence plate
>>> Wheel
>>> Side mirror
>>> Handle
>>> Make



Instance Attribute: Data member of an object, defined inside the constructor. Scope of access is within the object’s creation.

Class Attribute: Defined outside the constructor. Shared and accessible by all objects of the class.

```java
public class Coche {
    //Atributos
    String color;
    String marca;
    int km;

    public static void main(String[] args){
        Coche coche1= new Coche ();
        coche1.color="Azul";
        coche1.marca="Mazda";
        coche1.km=0;

        System.out.println("Color coche 1: "+coche1.color);
        System.out.println("Marca coche 1: "+coche1.marca);
        System.out.println("Kilometraje coche 1: "+coche1.km);

        // Se pueden crear más objetos
    }
}
```

Output - POO (run)

```
run:
Color coche 1: Azul
Marca coche 1: Mazda
Kilometraje coche 1: 0
BUILD SUCCESSFUL (total time: 0 seconds)
```

METHODS (ACTIONS)

In classes, methods define actions that objects can perform, like starting or stopping. They either manage an attribute or carry out a task, ensuring code is clear, consistent, and easy to reuse. Methods are like functions but are specifically tied to the class, so the same actions apply to all objects created from that class.

Methods

```
>>>headLightOn
>>>Start
>>>Brake
>>>Horn
>>>turnOnAC
>>>startWiper
>>>muffler
```

KEY CONCEPTS OF METHODS

```java
package ClasesObjetos;

import java.util.Scanner;

public class Operacion {
    //Atributos
    int n1,n2,suma;

     // Atributo para el objeto Scanner
    Scanner scanner = new Scanner(System.in);

    //Métodos

    //Método para pedir al usuario q nos digite dos números

    public void leer(){
        System.out.print("Digite el primer numero: ");
        n1 = scanner.nextInt();
        System.out.print("Digite el segundo numero: ");
        n2 = scanner.nextInt();
    }

    //Método para sumar ambos números
    public void sumar(){
        suma = n1+n2;

    }

    public void mostrar (){
        System.out.println("La suma es: "+suma);
    }
}
```

```java
    public static void main (String[] args){
        Operacion op = new  Operacion();

        op.leer();
        op.sumar();
        op.mostrar();

    }
}
```

Output - POO2 (run)

```
run:
Digite el primer numero: 4
Digite el segundo numero: 9
La suma es: 13
BUILD SUCCESSFUL (total time: 10 seconds)
```