Object-Oriented Programming (OOP) – Detailed Explanation

What is Object-Oriented Programming (OOP)?

Object-Oriented Programming (OOP) is a programming paradigm where **programs are designed using objects** that represent **real-world entities**.

An **object** is created using a **class**.
A **class** acts as a **blueprint or template**, and objects are the **real instances** created from that blueprint.

In OOP:

- **Every object has something → Properties (Variables)**
- **Every object does something → Behaviors (Methods)**

What is a Class?

A **class** is a **logical structure** that defines:

- **What an object has** → variables (data)
- **What an object can do** → methods (functions)

A class does not occupy memory until an object is created from it.

Example (conceptually):

- Class = Design of a house – like a blueprint
- Object = Actual house built from the design

What is an Object?

An **object** is a **real-world entity** created from a class.

An object occupies memory and can:

- Store data
- Call methods
- Interact with other objects

Example:

- Class: Dog
- Objects: dog1, dog2, dog3

Real-World Example – Dog

Dog is an animal.
It **has something** and **does something**.

Has (Variables / Properties):

- name
- color
- cost
- age

These are called:

- **Instance variables**
- Stored separately for each object

Does (Methods / Behaviors):

- barking()
- eating()
- running()

Methods represent **actions performed by the object**.

Real-World Example – Fan

Fan has properties and behaviors.

Properties (Variables):

- brand
- color
- cost
- speed

Behaviors (Methods):

- rotate()
- blowAir()
- stop()

This shows how **real-world objects are converted into programming objects**, which is the core strength of OOP.

Before OOP – Procedural Programming

Before OOP, programming was mainly **procedural**, like in **C language**.

Procedural Programming Characteristics:

- Program is divided into **functions**
- Data and functions are **separate**
- Focus is on **steps**, not objects

Disadvantages of Procedural Programming

- **Hard to manage large programs**
- **Code reuse is difficult**
- **No data security**
- **Changes in one part affect other parts**
- **Not suitable for real-world modeling**

Why OOP Was Introduced?

OOP was introduced to:

- Solve real-world problems easily
- Improve code structure
- Make programs scalable and reusable

Core Goals of Object-Oriented Programming

Goal	Meaning
Maintainability	Easy to modify and update code
Reusability	Write once, use multiple times
Extensibility	Add new features without breaking existing code
Security	Protect data using access control
Readability	Code is easy to understand and organized

How OOP Solves Real-World Problems

OOP allows:

- Mapping real-world entities to code
- Modeling relationships (like inheritance)
- Controlling access to data
- Managing large applications efficiently

Comparison: OOP vs Procedural Programming

Aspect	OOP	Procedural Programming
Approach	Object-based	Function-based
Data Security	High (encapsulation)	Low
Code Reusability	High	Low
Maintenance	Easy	Difficult
Real-World Modeling	Very easy	Hard
Examples	Java, C++, Python	C
Scalability	Suitable for large projects	Not suitable

Why Java Is Not 100% Object-Oriented

Java is **mostly object-oriented**, but **not 100% OOP**.

Reasons:

1. **Primitive data types exist**
    - int, double, char, boolean
    - These are not objects
2. **Static methods and variables**
    - Belong to class, not object
3. **Wrapper classes were added later**
    - To support OOP features

Hence, Java is called a “Hybrid OOP Language”.

Class vs Object in Java :

Aspect	Class	Object
Definition	Blueprint	Instance
Memory	No memory	Occupies memory
Nature	Logical	Physical
Creation	Using class keyword	Using new keyword
Example	Dog	dog1

The main Four pillars of Object Oriented Program are:

- Encapsulation
- Inheritance
- Polymorphism
- Abstraction

"We will address these topics in the future."

Conclusion

OOP is designed to model real-world systems, making programs:

- More organized
- Easier to maintain
- More secure
- Highly reusable

That’s why **Java is a pure object-oriented language** and widely used in enterprise-level applications.

At final:

What is a Class in Java

A **class** is a **blueprint or template** used to create objects.
It defines **what an object has** and **what an object can do**.

In a class, we define:

- **Variables** → what the object *has* (state / properties)
- **Methods** → what the object *does* (behavior)

What is an Object in Java

An **object** is a **real instance of a class**.
It represents a **real-world entity** created using the class.

An object:

- **Occupies memory**
- Has **actual values**
- Can **call methods**