

Introduction to Object Oriented Programming

Terminology - Some commonly used words in Object-Oriented Programming

- ▶ class
- ▶ object
- ▶ instance
- ▶ method
- ▶ overloading
- ▶ overriding
- ▶ static / class variable
- ▶ instance variable
- ▶ constructor
- ▶ modifiers
- ▶ access modifiers
- ▶ non-access modifiers
- ▶ abstract class
- ▶ interface
- ▶ this keyword
- ▶ super keyword
- ▶ inheritance
- ▶ polymorphism
- ▶ data abstraction
- ▶ encapsulation
- ▶ package
- ▶ scope of variables
- ▶ method signature
- ▶ annotation
- ▶ exception
- ▶ comparator
- ▶ parametric constructor

Pillars of Object-Oriented Programming

- ▶ **Abstraction**
- ▶ **Encapsulation**
- ▶ **Inheritance**
- ▶ **Polymorphism**

Data Abstraction

- ▶ Showing only the essential details to the user and not displaying trivial or non-essential units to the user.
- ▶ An example would be **driving a car**
- ▶ To drive a car, one should only know how to and when to use
 - ▶ Accelerator
 - ▶ Brakes
 - ▶ Gears
- ▶ The person driving a car doesn't need to know
 - ▶ How an accelerator increases the car speed
 - ▶ How applying brakes stops the car
 - ▶ Or how airbags (if any) are popping out when you hit or get hit by some other vehicle
- ▶ This is what Data Abstraction is, even though those inner details are required for a car to run (safely), the driver doesn't need to know them.
- ▶ In Java, abstraction can be achieved through concepts like
 - ▶ Abstract classes
 - ▶ Interfaces

Encapsulation

- ▶ Wrapping up data under a single unit
- ▶ It's a mechanism that binds together the code and the data it manipulates
- ▶ It's a protective shield that prevents the data from being accessed by the code outside this shield
- ▶ Technically, the data in a class is hidden from any other class and can be accessed only through any member function of the class in which they are declared
- ▶ In encapsulation, the data is hidden from other classes, which is like data-hiding. So, the names “encapsulation” and “data-hiding” are used synonymous.
- ▶ Encapsulation can be used private and protected access modifiers and using nested classes.

Inheritance

- ▶ Obtaining the features of another class
- ▶ The class which obtains the features from another class is referred as child or derived or sub class
- ▶ The class from which the features are obtained is called as a parent or super class
- ▶ It's like we humans passing on the essential genetic information (features) to the next generations
- ▶ Re-usability is the main use of inheritance
- ▶ As a class is a collection of member functions (methods) and member variables (variables / fields), the child class get the access to the members of parent class. So, we don't have to write them for the children again.
- ▶ We can achieve Inheritance in Java using **extends** keyword.
- ▶ Inheritance is also known as **“is-a”** relationship.

Polymorphism

- ▶ Poly means **many** and morphism means **ability to take forms**
- ▶ So, the **ability to appear in many forms** is Polymorphism.
- ▶ It refers to the ability of object-oriented programming languages to differentiate between entities with the same name efficiently.
- ▶ This is done by Java with the help of the signature and declaration of these entities.
- ▶ Polymorphism in Java is mainly of 2 types
 - ▶ Compile-Time Polymorphism. It can be achieved through a concept called **Method Overloading**
 - ▶ Run-Time Polymorphism. It can be achieved through a concept called **Method Overriding**

Class

- ▶ A class is a collection of member functions and member variables
- ▶ Member functions are actions / behaviors (Verbs)
- ▶ Member variables are attributes (Nouns)

1.

▶ Student

2.



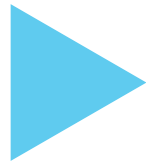
Bank Customer

3.



Person

4.



Circle

5.



Hospital

6.



Library

7.



Superhero

8.



Bank ATM

9.



Vehicle

10.

▶ **Memer / Troller /
Instagram model**