Java OOPs Concepts

The main aim of object-oriented programming is to implement real-world entities, for example, object, classes, abstraction, inheritance, polymorphism, etc.

**Class**

A *class* is the blueprint from which individual objects are created. Class is a one which contains states (variables) and behaviour (methods)

**Object**

An object is an instance of a class. The term ‘object’, however, refers to an actual **instance** of a class. Every object must belong to a class.

Object of a calss can be created using the new keyword and name of the class.

**public class** Test

{

**public static void** main(String[] args)

{

Lover l1 = **new** Lover();

Lover l2 = **new** Lover();

}

}

**OOPS CONCEPTS**

**Inheritance**

*It’s a mechanism where the child class acquires the properties of the parent class or supper class.*

*java supports inheritance there are many types of in heritance*

o *single inheritance ( above example )*

o *multilevel inheritance*

o *hierarchical inheritance*

## **Single Inheritance Example**

When a class inherits another class, it is known as a single inheritance. In the example given below, Dog class inherits the Animal class, so there is the single inheritance.

## **Multilevel Inheritance Example**

When there is a chain of inheritance, it is known as multilevel inheritance. As you can see in the example given below, BabyDog class inherits the Dog class which again inherits the Animal class, so there is a multilevel inheritance.

## **Hierarchical Inheritance Example**

When two or more classes inherits a single class, it is known as hierarchical inheritance. In the example given below, Dog and Cat classes inherits the Animal class, so there is hierarchical inheritance.

*Java does not support the multiple inheritance where has it only supports multilevel inheritance.*

# Polymorphism

**Polymorphism in Java** is a concept by which we can perform a single action in different ways. Polymorphism is derived from 2 Greek words: poly and morphs. The word "poly" means many and "morphs" means forms. So polymorphism means many forms.

There are two types of polymorphism in Java: compile-time polymorphism and runtime polymorphism. We can perform polymorphism in java by method overloading and method overriding.

## **Runtime Polymorphism in Java**

**Runtime polymorphism** or **Dynamic Method Dispatch** is a process in which a call to an overridden method is resolved at runtime rather than compile-time.

### **Abstraction in Java**

**Abstraction** is a process of hiding the implementation details and showing only functionality to the user.

Another way, it shows only essential things to the user and hides the internal details, for example, sending SMS where you type the text and send the message. You don't know the internal processing about the message delivery.

### **Ways to achieve Abstraction**

There are two ways to achieve abstraction in java

1. Abstract class (0 to 100%)
2. Interface (100%)

# Encapsulation in Java

**Encapsulation in Java** is a *process of wrapping code and data together into a single unit*, for example, a capsule which is mixed of several medicines.



We can create a fully encapsulated class in Java by making all the data members of the class private. Now we can use setter and getter methods to set and get the data in it.

**Constructor**

*Constructor is special kind of method which gets called when object is created*

 *Constructor don’t have return type*

 *Constructor cannot be inherited*

 *Constructor name is same has the class name*

 *Every class should have a constructor, if we don’t write a constructor compiler will write a default constructor.*

 *Compiler will not write a default constructor, if we are going to write any of the constructor explicitly*

 *Constructor with no parameters or a zero parameter constructor is called has default constructor*

 *Constructor with parameter is called has parameterized constructor. These constructor is used to initialise the instance variables at the time of creating object*