Java World

A blog about java tutorials , java interview questions.

Object Oriented Programming (OOP) concepts interview questions

What is an Object ?

Well It is the most basic concept and a key to understand the Object-Oriented programming . Object is an entity that has two characteristics , State and Behavior . Some examples of real world object can be : Bike , Chair , Dog etc. Lets take an example of a Bike . Bike has some state (current gear , current speed) and behavior (change gear , apply brake) . One way to begin thinking in an object-oriented way is to identify the state and behavior of real world objects . Software objects are also similar to the real world objects. They too contains State and Behavior . An Object stores its state in fields and exposes the behavior through methods.

What is Class ?

Class is a blueprint from which objects of same type can be created . Lets take an example of a Bike again . There are thousands of bikes with the same characteristics i.e having same make and model . They are created from the same prototype / blueprint called class.

What are the principles concepts of OOPS?

The principle concepts of OOPs are :

- Inheritance
- Polymorphism
- Abstraction
- Encapsulation

What is Inheritance?

Inheritance is the capability of one class to acquire the common state and behavior of another class while adding its own functionality . In Object Oriented programming , the concept of inheritance provides the idea of re-usability . This means we can add additional features to a class without modifying it.

How to achieve Inheritance in Java ?
extends keyword is used to inherit features of a class .

What is Polymorphism?

Polymorphism means ability of one type to take more than one form . This can also be applied in Java where in sub-classes of a class can define their own unique behaviors and yet share some of the common functionalites of the parent class.
Example : Lets take an example of Bike . A company can produce different kind of bikes say , a mountain bike , and a roadbike . Now the two different types of bikes inherit the common features of a bike but also define their own separate features.

How to achieve Polymorphism in Java ?
Overloading and **Overriding** are the two ways to achieve Polymorphism.

What is Abstraction?

Abstraction is an essential element of Object Oriented programming . It refers to the act of presenting essential features without including the background details or explanations. We can say that abstraction is purposeful suppression , hiding of some details of a process or artifact , in order to bring out more clearly other aspects , details or structures. Abstraction can be of control (abstraction of actions) or data .

How to achieve Abstraction in Java ?
Abstract classes and **Interfaces** are used to achieve Abstraction in Java.

What is Encapsulation?

Encapsulation is the binding of data and its associated functions in a single unit called Class . Access to the code and data inside the wrapper is tightly controlled through a well defined interface. Encapsulation provides a protective barrier that prevents the access of code and data from outside the class.

The main benefit of encapsulation is the ability to re-factor the implemented code without breaking the code of others and thus provides flexibility and maintainability.

What is the difference between abstraction and encapsulation?

The two concepts are distinct , but closely related and are often found together.

- Abstraction focuses on the outside view of an object and hides any unnecessary detail while Encapsulation provides a protective barrier that prevents the access of code and data from outside the class.
- Abstraction solves the problem in the design side while Encapsulation is the Implementation and infact I would say that Encapsulation in effect provides Abstraction.

Explain the different types and forms of Polymorphism.

Polymorphism is of two types i.e **Compile time** and **run time polymorphism**. Compile time polymorphism is method overloading. Runtime time polymorphism is done using inheritance and interface.

Different forms of polymorphism are - **Method overloading** and **Method overriding** (through inheritance and interfaces).

What is runtime polymorphism or dynamic method dispatch?

In Java, runtime polymorphism is a process in which call to an overridden method is resolved at runtime rather than at compile-time. The overridden method is called through the reference variable of the superclass but the determination of actual method to be invoked is based on the type of object created at run time.

What is Dynamic Binding?

Resolving the actual method to be invoked on basis of a method call is called Binding.Methods can be resolved at compile time or run time.

Types of binding are :

- Static binding (early binding) - Method call is resolved at compile time.
- Dynamic binding (late binding) - Method call is resolved at run time.

What is method overloading?

Defining multiple methods with same name in the same class but different arguments is overloading . The benefit of method overloading is that it allows you to implement methods that support the same semantic operation but differ by argument number or type.

Important points about Overloading :

- Overloaded methods MUST change either the number of arguments or the type of arguments.
- Overloaded methods CAN change the access modifier.
- Overloaded methods CAN declare new or broader checked exceptions.
- A method can be overloaded in the same class or in a subclass.

What is method overriding?

Method overriding occurs when sub class declares a method that has the same type arguments as a method declared by one of its superclass. The key benefit of overriding is the ability to define behavior that's specific to a particular subclass type. The overriding method cannot have a more restrictive access modifier than the method being overridden. The overriding method cannot have more restrictive access modifier. The subclass can not override a method marked final or static.

What are the differences between method overloading and method overriding?

Below table shows the differences :

Parameters	Overloading	Overriding
Arguments	Must change	Must not change
Return type	Can change	Can't change except for covariant returns
Exceptions	Can change	Can reduce or eliminate. Must not throw new or broader checked exceptions
Access	Can change	Must not make more restrictive (can be less restrictive)
Invocation	Reference type determines which overloaded version is selected. Happens at compile time.	Object type determines which method is selected. Happens at runtime.

Can we override already overloaded method ?

YES , sub classes can override the overloaded methods.

Is it possible to override the main method?

Static method can't be overridden in Java.

How to invoke a superclass version of an Overridden method?

Using **super** keyword

How do you prevent a method from being overridden?

Make it **final** .