40 OOPS
Interview Q/A

1) What is Object-Oriented Programming (OOP)?

OOP is a programming paradigm based on the concept of objects, which can contain data in the form of fields (attributes or properties), and code in the form of procedures (methods or functions).

2) What are the four main principles of OOP?

Encapsulation, Inheritance, Polymorphism, and Abstraction.

3) Define Encapsulation.

Encapsulation is the bundling of data and methods that operate on the data into a single unit (class). It restricts direct access to some of the object's components.

4) What is Inheritance?

Inheritance is a mechanism in which one class acquires the properties and behaviors of another class. It promotes code reusability and establishes a relationship between classes.

5) Explain Polymorphism.

Polymorphism means the ability of objects to take on multiple forms. In programming, it allows objects of different classes to be treated as objects of a common superclass.

6) What is Abstraction?

Abstraction is the process of hiding the implementation details and showing only the essential features of the object. It helps to reduce programming complexity and effort.

7) What is a class?

A class is a blueprint or template for creating objects. It defines the properties and behaviors common to all objects of a certain kind.

8) What is an object?

An object is an instance of a class. It has state (attributes) and behavior (methods) defined by its class.

9) What is a constructor?

A constructor is a special method that is automatically called when an object of a class is instantiated. It is used to initialize the object's state.

10) Differentiate between a class and an object.

A class is a blueprint or template for creating objects, whereas an object is an instance of a class.

11) What is method overloading?

Method overloading is a feature that allows a class to have multiple methods with the same name but different parameters. The methods must differ in the number or types of their parameters.

12) What is method overriding?

Method overriding occurs when a subclass provides a specific implementation of a method that is already defined in its superclass. It allows a subclass to provide a specialized version of a method that is already provided by its superclass.

13) What is the 'super' keyword used for?

The 'super' keyword is used to refer to the immediate parent class object. It is used to invoke the superclass constructor, methods, or variables from the subclass.

14) What is a static method?

A static method belongs to the class rather than any specific instance. It can be called on the class itself without the need to create an instance of the class.

15) What is a static variable?

A static variable is a class-level variable shared among all instances of the class. There is only one copy of the static variable regardless of how many instances of the class are created.

16) What is a final class?

A final class is a class that cannot be subclassed. It prevents other classes from inheriting from it.

17) What is the final method?

A final method is a method that cannot be overridden by subclasses. Once defined in a class, its implementation cannot be changed in any subclass.

18) What is an abstract class?

An abstract class is a class that cannot be instantiated on its own and is meant to be subclassed. It may contain abstract methods, which are declared but not implemented in the abstract class.

19) What is an interface?

An interface is a reference type similar to a class that can contain only constants, method signatures, default methods, static methods, and nested types. It provides a contract for classes to adhere to.

20) Can an interface implement another interface?

Yes, an interface can extend another interface using the 'extends' keyword.

21) What is the difference between an abstract class and an interface?

Abstract classes can have method implementations and instance variables, while interfaces cannot contain method implementations or instance variables. A class can implement multiple interfaces but can only inherit from one abstract class.

22) What is multiple inheritance, and does Java support it?

Multiple inheritance is a feature where a class can inherit properties and behavior from more than one parent class. Java does not support multiple inheritance for classes but allows multiple inheritance for interfaces.

23) What is a package in Java?

A package in Java is a mechanism for organizing classes and interfaces into namespaces. It helps in preventing naming conflicts and provides access protection.

24) What is the difference between an ArrayList and a LinkedList?

An ArrayList internally uses a dynamic array to store elements and provides fast access to elements using indexes. A LinkedList uses a doubly linked list to store elements and provides fast insertion and deletion operations.

25) What is the difference between the '== 'operator and the 'equals()' method?

The '==' operator is used to compare the references of two objects, whereas the 'equals()' method is used to compare the contents or values of two objects.

26) What is a wrapper class?

A wrapper class is a class that wraps or encapsulates primitive data types into objects. It provides utility methods for performing operations on primitive data types.

27) What is the 'this' keyword used for?

The 'this' keyword is used to refer to the current object within an instance method or constructor. It is used to differentiate between instance variables and local variables with the same name.

28) What is garbage collection?

Garbage collection is the process of automatically reclaiming memory occupied by objects that are no longer in use by the program. It helps in preventing memory leaks and managing memory efficiently.

29) What is method chaining?

Method chaining is a technique where multiple methods are called on an object in a single statement by returning the object itself from each method.

30) What is a singleton class?

A singleton class is a class that allows only one instance of itself to be created and provides a global point of access to that instance.

31) What is the difference between composition and inheritance?

Composition is a "has-a" relationship where a class contains an instance of another class, whereas inheritance is an "is-a" relationship where a subclass inherits properties and behavior from a superclass.

32) What is the difference between aggregation and composition?

Aggregation is a type of association where a class has a relationship with another class but does not own the other class, whereas composition is a stronger form of aggregation where a class owns the other class and is responsible for its lifecycle.

33) What is a deep copy and a shallow copy?

A deep copy creates a new copy of an object and copies all of its fields recursively, whereas a shallow copy creates a new copy of an object but only copies its immediate fields, not the objects referenced by its fields.

34) What is method visibility in Java?

Method visibility refers to the accessibility of methods from other classes. There are four types of method visibility in Java: public (accessible from any class), protected (accessible from subclasses and classes in the same package), default (accessible only within the same package), and private (accessible only within the same class).

35) What is the method hiding in Java?

Method hiding occurs when a subclass defines a static method with the same signature as a static method in its superclass. The subclass method hides the superclass method, and the method is resolved at compile-time based on the reference type.

36) What is a constructor chaining?

Constructor chaining is the process of calling one constructor from another constructor within the same class or from a subclass constructor to a superclass constructor. It helps in reusing code and initializing objects.

37) What is the purpose of the 'final' keyword in Java?

The 'final' keyword is used to restrict the modification of classes, methods, and variables. It can be applied to classes to prevent subclassing, methods to prevent overriding, and variables to create constants.

38) What is a marker interface?

A marker interface is an interface with no methods or fields. It is used to indicate a special property or behavior to the compiler or runtime environment.

39) What is the difference between checked and unchecked exceptions?

Checked exceptions are checked at compile-time and must be handled by the programmer, whereas unchecked exceptions are not checked at compile-time and do not require handling.

40) What is the purpose of the 'extends' keyword in Java?

The 'extends' keyword is used to create a subclass that inherits properties and behavior from a superclass. It establishes an inheritance relationship between classes.