50 Object Oriented Programming Interview QnA

Made by Coding Hubs Programmer

Want More Search Coding Hubs On Telegram & Join Channel

1. What is object-oriented programming (OOP)?

Object-oriented programming is a programming paradigm that organizes code into objects, which are instances of classes. It focuses on encapsulating data and behavior together and emphasizes concepts such as inheritance, polymorphism, and abstraction.

2. What is a class?

A class is a blueprint or template that defines the structure and behavior of objects. It defines the properties (attributes) and methods (functions) that objects of that class will have.

3. What is an object?

An object is an instance of a class. It represents a specific entity or concept and encapsulates its own unique state and behavior defined by the class.

4. What is inheritance?

Inheritance is a mechanism in which a class inherits properties and methods from another class, known as the superclass or base class. It allows for code reuse, promotes modularity, and establishes hierarchical relationships between classes.

5. What is encapsulation?

Encapsulation is the principle of bundling data and methods within a class, hiding internal implementation details and providing controlled access through methods or properties. It ensures data integrity, promotes code organization, and supports information hiding.

6. What is polymorphism?

Polymorphism is the ability of objects of different classes to respond to the same message or method invocation in different ways. It allows for flexibility, extensibility, and code reuse. Polymorphism can be achieved through method overriding and method overloading.

7. What is abstraction?

Abstraction is the process of representing essential features of an object while hiding unnecessary details. It focuses on creating simplified models that capture the core characteristics and behaviors relevant to the problem domain.

8. What is method overloading?

Method overloading is a feature that allows multiple methods with the same name but different parameters to coexist within a class. The compiler determines the appropriate method to call based on the number, types, and order of the arguments passed.

9. What is method overriding?

Method overriding is a feature that allows a subclass to provide its own implementation of a method defined in its superclass. The method in the subclass must have the same name, return type, and parameters as the method in the superclass.

10. What is a constructor?

A constructor is a special method within a class that is used to initialize objects of that class. It is called automatically when an object is created and helps set the initial state of the object.

11. What is the difference between a class method and an instance method?

A class method is a method that belongs to the class itself, rather than an instance of the class. It can be called directly on the class and does not have access to instance-specific data. An instance method, on the other hand, is associated with objects of the class and can access and modify instance-specific data.

12. What is method overloading?

Method overloading is a feature that allows a class to have multiple methods with the same name but different parameter lists. The compiler determines which method to invoke based on the number, types, and order of the arguments passed.

13. What is method overriding?

Method overriding is a feature that allows a subclass to provide its own implementation of a method that is already defined in its superclass. The method in the subclass must have the same name, return type, and parameters as the method in the superclass.

14. What is the difference between composition and inheritance?

Composition and inheritance are two ways to achieve code reuse and establish relationships between classes. Composition is a "has-a" relationship, where a class contains objects of other classes as its members. Inheritance is an "is-a" relationship, where a subclass inherits properties and behaviors from a superclass. Composition offers more flexibility and loose coupling, while inheritance provides a more rigid and hierarchical structure.

15. What is an abstract class?

An abstract class is a class that cannot be instantiated and serves as a base or parent class for other classes. It may contain abstract methods, which are methods without implementation, and concrete methods that provide default behavior. Subclasses derived from an abstract class must implement all the abstract methods or be declared abstract themselves.

16. What is an interface?

An interface is a collection of abstract methods that define a contract for classes to follow. It specifies a set of methods that implementing classes must provide. An interface can also include constants. Classes can implement multiple interfaces but can only inherit from a single class.

17. What is the difference between an abstract class and an interface?

An abstract class can contain both abstract and concrete methods, whereas an interface can only have abstract methods. A class can implement multiple interfaces, but it can only inherit from a single abstract class. Abstract classes can have constructor definitions, while interfaces cannot. Additionally, abstract classes can have instance variables, while interfaces cannot.

18. What is a static method?

A static method is a method that belongs to the class itself rather than an instance of the class. It can be called directly on the class without creating an object. Static methods are typically used for utility functions or operations that do not depend on the state of individual objects.

19. What is the difference between a static method and an instance method?

A static method belongs to the class itself and can be called directly on the class, while an instance method belongs to an object of the class and can only be called on instances of the class. Static methods do not have access to instance-specific data and can only access static variables, while instance methods can access both instance variables and static variables.

20. What is method hiding?

Method hiding occurs when a subclass defines a static method with the same name as a static method in its superclass. The subclass method "hides" the superclass method, and the choice of which method to call is determined at compile-time based on the reference type, not the actual object type.

21. What is a final class?

A final class is a class that cannot be inherited or subclassed. It is typically used when a class is complete and should not be extended further.

22. What is a final method?

A final method is a method that cannot be overridden by subclasses. Once a method is declared final in a superclass, it cannot be changed or overridden in any of its subclasses.

23.What is method chaining?

Method chaining, also known as fluent interface, is a coding style in which multiple method invocations are chained together in a single statement. Each method call returns an object, allowing subsequent method calls to be made on the returned object. This pattern improves code readability and conciseness.

24. What is a singleton class?

A singleton class is a class that allows only a single instance to be created and provides global access to that instance. It is often used when there is a need for centralized control or coordination.

25. What is a static variable?

A static variable is a variable that belongs to the class itself, rather than individual instances of the class. It is shared among all instances of the class and can be accessed directly through the class name.

26. What is method synchronization?

Method synchronization is a technique used to ensure that only one thread can execute a synchronized method of an object at a time. It prevents multiple threads from concurrently accessing or modifying shared data, thereby avoiding data inconsistencies or race conditions.

27. What is the difference between shallow copy and deep copy?

Shallow copy creates a new object that shares the same memory as the original object, including references to the referenced objects. Deep copy creates a completely independent copy of the object and all its referenced objects, recursively.

28. What is the purpose of the "this" keyword in Java?

The "this" keyword in Java is a reference to the current object within an instance method or constructor. It is used to differentiate between instance variables and parameters or local variables with the same name. It can also be used to invoke other constructors of the same class.

29. What is method overloading in Java?

Method overloading in Java is the ability to have multiple methods with the same name but different parameter lists within a class. The compiler determines which method to call based on the number, types, and order of the arguments passed.

30. What is method overriding in Java?

Method overriding in Java is the ability of a subclass to provide its own implementation of a method that is already defined in its superclass. The method in the subclass must have the same name, return type, and parameters as the method in the superclass.

31. What is the difference between checked and unchecked exceptions?

Checked exceptions are exceptions that need to be declared in the method signature or handled using try-catch blocks. They are typically used for recoverable conditions that the calling code should be aware of and handle. Unchecked exceptions, also known as runtime exceptions, do not need to be explicitly declared or caught. They are used for unexpected or unrecoverable conditions such as programming errors or system failures.

32. What is the Java Collections Framework?

The Java Collections Framework is a set of interfaces, classes, and algorithms that provide reusable data structures and methods for manipulating collections of objects. It includes classes like ArrayList, LinkedList, HashMap, and interfaces like List, Set, and Map.

33. What is the difference between ArrayList and LinkedList?

ArrayList and LinkedList are both implementations of the List interface in the Java Collections Framework. The main difference between them is in their internal data structure. ArrayList uses a dynamically resizable array, providing fast random access but slower insertions and deletions. LinkedList uses a doubly linked list, providing efficient insertions and deletions but slower random access.

34. What is the difference between HashSet and TreeSet?

HashSet and TreeSet are both implementations of the Set interface in the Java Collections Framework. HashSet stores elements in an unordered manner, using hashing to provide constant-time performance for add, remove, and contains operations. TreeSet stores elements in a sorted manner, providing log-time performance for these operations.

35. What is the difference between an ArrayList and an array?

An ArrayList is a dynamic data structure provided by the Java Collections Framework, while an array is a fixed-size container for holding elements of the same type. Unlike arrays, ArrayLists can grow or shrink dynamically and provide additional methods for manipulating the collection.

36. What is the difference between a deep copy and a shallow copy?

A deep copy creates a completely independent copy of an object and all its referenced objects, recursively, while a shallow copy creates a new object that shares the same memory as the original object, including references to the referenced objects.

37. What is the difference between composition and aggregation?

Composition and aggregation are both forms of association between classes, but with different lifetimes and ownership semantics. In composition, the composed object cannot exist without the owning object, and if the owning object is destroyed, the composed object is also destroyed. In aggregation, the aggregated object can exist independently and has a weaker ownership relationship with the owning object.

38. What is the difference between abstract classes and interfaces in Java?

Abstract classes can have both abstract and concrete methods, while interfaces can only have abstract methods. A class can implement multiple interfaces, but it can only inherit from a single abstract class. Abstract classes can have constructor definitions, while interfaces cannot. Additionally, abstract classes can have instance variables, while interfaces cannot.

39. What is the difference between method overloading and method overriding?

Method overloading occurs when a class has multiple methods with the same name but different parameter lists. The methods may have different return types, but the method name must be the same. Method overriding occurs when a subclass provides its own implementation of a method that is already defined in its superclass. The method in the subclass must have the same name, return type, and parameters as the method in the superclass.

40. What is the purpose of the "super" keyword in Java?

The "super" keyword in Java is used to refer to the superclass of a class. It can be used to access superclass methods, invoke superclass constructors, and differentiate between superclass and subclass members with the same name.

41. What are access modifiers in OOP?

Access modifiers define the level of access to class members (variables and methods) from other parts of the program. The three main access modifiers are:

Public: Allows access from anywhere.

Private: Restricts access to within the class itself.

Protected: Allows access within the class and its subclasses.

42. What is method hiding?

Method hiding occurs when a subclass defines a static method with the same name and signature as a static method in its superclass. The subclass method hides the superclass method, and the method chosen for execution depends on the type of the reference used to invoke it, not the actual type of the object.

43. What is the purpose of the 'this' keyword?

The 'this' keyword is a reference to the current object within a class. It is used to differentiate between class members (variables and methods) and local variables or

parameters that have the same name. It is also used to invoke one constructor from another constructor in the same class.

44. What is the purpose of the 'super' keyword?

The 'super' keyword is used to refer to the superclass (parent class) of a subclass. It can be used to call the superclass constructor, access superclass methods and variables, and invoke the superclass implementation of overridden methods.

45. What is the difference between method overloading and method overriding?

Method overloading: Method overloading occurs within the same class and involves creating multiple methods with the same name but different parameters. The methods have different signatures and are differentiated based on the number, type, or order of parameters. Method overloading is determined at compile-time (static binding).

Method overriding: Method overriding occurs between a superclass and its subclass. It involves creating a method in the subclass with the same name, return type, and parameters as the method in the superclass. The subclass method provides a specific implementation that overrides the superclass method. Method overriding is determined at runtime (dynamic binding).

46. What is method overriding?

Method overriding is a feature in Java that allows a subclass to provide a specific implementation of a method that is already defined in its superclass. The subclass must use the same method signature (name, return type, and parameters) as the superclass method. Method overriding is used to achieve polymorphism, where objects of different classes can be treated interchangeably based on their common superclass.

47. What is method overloading?

Method overloading is a feature in Java that allows a class to have multiple methods with the same name but different parameters. The methods can have different numbers of parameters or different types of parameters. During compilation, the appropriate method is selected based on the arguments passed to it. Method overloading allows for code reuse and provides a way to create methods with similar functionality but different input.

48. What is a singleton class?

A singleton class is a class that allows only a single instance of itself to be created. It provides a global point of access to that instance. Singleton classes are often used for resource sharing, logging, caching, or when there should be only one instance of a class throughout the application.

49. What is the purpose of the 'this' keyword?

The 'this' keyword is a reference to the current object within a class. It is used to differentiate between class members (variables and methods) and local variables or

parameters that have the same name. It is also used to invoke one constructor from another constructor in the same class.

50. What is method chaining?

Method chaining is a technique in which multiple methods are invoked on an object in a single line, using the return value of one method as the object for the next method call. It improves code readability and conciseness.