CORE CS (THEORY)

By Abhishek Rathor

Instagram: code.abhii07 (SYNTAX ERROR)

❖Top 50 OOPS Interview Questions

---

1. What is OOPS?

OOPS (Object-Oriented Programming System) is a programming approach based on objects rather than functions. It focuses on real-world entities like objects and classes. OOPS improves code reusability, scalability, and maintainability. Languages like Java, C++, and Python support OOPS.

---

2. What is a Class?

A class is a blueprint or template used to create objects. It defines properties (variables) and behaviors (methods). A class does not occupy memory until an object is created. It helps in organizing code logically.

---

3. What is an Object?

An object is an instance of a class. It represents a real-world entity with state and behavior. Objects occupy memory and can interact with each other. Example: a car object created from a Car class.

---

4. What is Encapsulation?

Encapsulation is the process of wrapping data and methods into a single unit called a class. It protects data from unauthorized access using access modifiers. Encapsulation improves data security and control. It is also known as data hiding.

---

5. What is Abstraction?

Abstraction means hiding internal implementation details and showing only essential features. It helps reduce complexity and increases efficiency. Abstraction is achieved using abstract classes and interfaces. It focuses on what an object does, not how it does.

---

6. What is Inheritance?

Inheritance is a mechanism where one class acquires the properties of another class. It promotes code reusability and reduces redundancy. The existing class is called parent, and the new class is child. It represents an “is-a” relationship.

---

7. What is Polymorphism?

Polymorphism means one function behaving in multiple forms. It allows the same method name to perform different tasks. Polymorphism can be achieved using method overloading and method overriding. It improves flexibility and readability.

---

8. What is Method Overloading?

Method overloading occurs when multiple methods have the same name but different parameters. It is a compile-time polymorphism. Overloading improves code readability. It does not depend on return type alone.

---

9. What is Method Overriding?

Method overriding occurs when a child class provides a specific implementation of a parent class method. It is runtime polymorphism. The method signature must be the same. It supports dynamic method dispatch.

---

10. What is Constructor?

A constructor is a special method used to initialize objects. It is called automatically when an object is created. Constructors have the same name as the class. They do not have a return type.

---

11. What is Destructor?

A destructor is used to free memory occupied by objects. It is called automatically when an object is destroyed. Destructors help in resource management. Mainly used in languages like C++.

---

12. What is an Interface?

An interface is a blueprint that contains abstract methods only. It supports complete abstraction. A class can implement multiple interfaces. Interfaces help achieve multiple inheritance.

---

13. What is an Abstract Class?

An abstract class is a class that cannot be instantiated. It may contain both abstract and non-abstract methods. It provides partial abstraction. Child classes must implement abstract methods.

---

14. Difference between Class and Object

A class is a blueprint, while an object is a real instance. Class does not occupy memory, object does. Multiple objects can be created from one class. Class defines structure, object represents data.

---

15. Difference between Abstraction and Encapsulation

Abstraction hides implementation details, while encapsulation hides data. Abstraction focuses on design level, encapsulation on implementation level. Abstraction is achieved using interfaces and abstract classes. Encapsulation is achieved using access modifiers.

---

16. What are Access Modifiers?

Access modifiers define the scope of variables and methods. Common types are public, private, protected, and default. They help in data security. Proper use improves code control and safety.

---

17. What is Data Hiding?

Data hiding is restricting access to internal data. It is achieved using private access modifiers. It protects data from misuse. Data hiding is a part of encapsulation.

---

18. What is Static Keyword?

Static keyword is used to share variables or methods among all objects. Static members belong to the class, not the object. Memory is allocated only once. It improves memory efficiency.

---

19. What is Final Keyword?

Final keyword is used to restrict modification. A final variable cannot be changed. A final method cannot be overridden. A final class cannot be inherited.

---

20. What is This Keyword?

This keyword refers to the current object. It is used to differentiate between instance variables and parameters. It can call current class methods or constructors. It improves code clarity.

21. What is Super Keyword?

Super keyword refers to the parent class object. It is used to access parent class methods and variables. It can also call parent constructors. Useful in inheritance.

---

22. What is Multiple Inheritance?

Multiple inheritance allows a class to inherit from more than one class. It is not supported directly in Java due to ambiguity. It can be achieved using interfaces. C++ supports it directly.

---

23. What is Hierarchical Inheritance?

Hierarchical inheritance occurs when multiple child classes inherit from one parent class. It promotes code reusability. It represents a tree-like structure. Commonly used in real-world applications.

---

24. What is Hybrid Inheritance?

Hybrid inheritance is a combination of two or more inheritance types. It includes multiple and hierarchical inheritance. It increases complexity. Achieved using interfaces in Java.

---

25. What is Runtime Polymorphism?

Runtime polymorphism occurs when method calls are resolved at runtime. It is achieved using method overriding. It depends on object type, not reference type. It supports dynamic behavior.

---

26. What is Compile-time Polymorphism?

Compile-time polymorphism occurs during compilation. It is achieved using method overloading. Method selection happens at compile time. It is faster than runtime polymorphism.

---

27. What is Object-Oriented Design?

Object-oriented design focuses on planning software using objects. It improves modularity and maintainability. Design is based on classes and relationships. It simplifies complex systems.

---

28. What is Association?

Association represents a relationship between two classes. It shows how objects interact with each other. It can be one-to-one or one-to-many. It does not represent ownership.

29. What is Aggregation?

Aggregation is a weak association where child can exist independently. It represents a “has-a” relationship. Example: Department and Teacher. It improves code structure.

---

30. What is Composition?

Composition is a strong association where child cannot exist independently. It also represents a “has-a” relationship. If parent is destroyed, child is destroyed too. It provides better control.

---

31. What is Coupling?

Coupling refers to dependency between classes. Low coupling is preferred. It makes code flexible and easy to maintain. High coupling increases complexity.

---

32. What is Cohesion?

Cohesion refers to how closely related methods are in a class. High cohesion is desirable. It improves readability and maintainability. Each class should have a single responsibility.

---

33. What is Object Cloning?

Object cloning creates an exact copy of an object. It helps avoid creating objects repeatedly. Cloning can be shallow or deep. Used for performance optimization.

---

34. What is Garbage Collection?

Garbage collection automatically frees unused memory. It prevents memory leaks. Managed by JVM in Java. Improves program reliability.

---

35. What is Method Hiding?

Method hiding occurs when a static method in child hides parent static method. It is not true overriding. Method selection depends on reference type. Happens at compile time.

---

36. What is Virtual Function?

A virtual function is a member function overridden in derived class. It supports runtime polymorphism. Mainly used in C++. It ensures correct method call at runtime.

---

37. What is Friend Function?

Friend function can access private members of a class. It is declared using friend keyword. Used mainly in C++. It breaks encapsulation but useful in some cases.

---

38. What is Namespace?

Namespace is used to avoid name conflicts. It groups related identifiers together. Mainly used in C++. Improves code organization.

---

39. What is Object-Oriented Programming vs Procedural Programming?

OOPS focuses on objects, procedural focuses on functions. OOPS supports reusability and security. Procedural programming is simpler but less scalable. OOPS is better for large applications.

---

40. What is Message Passing?

Message passing is communication between objects. Objects interact by invoking methods. It improves modularity. Common in distributed systems.

---

41. What is Late Binding?

Late binding binds method calls at runtime. It is used in runtime polymorphism. Depends on object type. Also known as dynamic binding.

---

42. What is Early Binding?

Early binding binds method calls at compile time. Used in method overloading. Faster than late binding. Also called static binding.

---

43. What is Object Serialization?

Serialization converts object into byte stream. Used to store or transfer objects. Deserialization restores object state. Commonly used in Java.

---

44. What is Marker Interface?

Marker interface has no methods. It provides metadata to JVM. Example: Serializable, Cloneable. Used for special behavior.

---

45. What is Shallow Copy?

Shallow copy copies object references. Changes affect original object. Faster but unsafe. Used when deep copy not required.

---

46. What is Deep Copy?

Deep copy creates independent object copies. Changes do not affect original. Safer but slower. Used when data isolation is needed.

---

47. What is Object Lifetime?

Object lifetime is the duration object exists in memory. It starts at creation and ends at destruction. Managed automatically in Java. Important for memory management.

---

48. What is UML?

UML stands for Unified Modeling Language. It is used to design object-oriented systems. It provides visual diagrams. Helps in planning software structure.

---

49. What is SOLID Principle?

SOLID principles guide good object-oriented design. They improve code quality and flexibility. Include five design principles. Widely used in industry.

---

50. Why use OOPS?

OOPS improves code reusability and scalability. It models real-world problems effectively. Enhances security and maintenance. Preferred for large software systems.

Connect & Share:

Tag us on your success stories:

- **Instagram**: @code.abhii07
- **YouTube**: SYNTAX ERROR