Top 100 Java Interview Questions & Answers

Author - Praveen B

Core Java

1. **What is Java?**
   Java is an object-oriented, platform-independent programming language used for building applications.

2. **What are the features of Java?**
   Java is platform-independent, object-oriented, secure, robust, multithreaded, and provides automatic memory management.

3. **Difference between JDK, JRE, and JVM?**
   JDK is for development, JRE is for running Java apps, and JVM executes bytecode on different platforms.

4. **Why is Java platform-independent?**
   Java compiles code into bytecode, which JVM interprets on any OS.

5. **What is bytecode in Java?**
   Bytecode is an intermediate representation of Java code executed by JVM.

6. **What is the difference between path and classpath?**
   Path is for locating Java binaries, while classpath is for loading classes and libraries.

7. **What is an object in Java?**
   An object is an instance of a class that contains state (variables) and behavior (methods).

8. **What is a class in Java?**
   A class is a blueprint for objects, defining variables and methods.

9. **Difference between primitive and reference data types?**
   Primitive types (int, double) store values, while reference types (String, arrays) store memory addresses.

10. **What is encapsulation?**
    Encapsulation is the wrapping of data and methods into a single unit (class) with controlled access via getters and setters.

11. **What is inheritance?**
    Inheritance allows a class to acquire properties and behaviors from another class using extends.

12. **What is polymorphism?**
    Polymorphism enables a method to perform different behaviors in different contexts (method overloading and overriding).

13. **What is method overloading?**
    Method overloading allows multiple methods with the same name but different parameters in the same class.

14. **What is method overriding?**
    Method overriding occurs when a subclass provides a specific implementation of a method from the parent class.

15. **What is abstraction?**
    Abstraction hides implementation details and exposes only necessary functionality using abstract classes or interfaces.

16. **Difference between abstract class and interface?**
    Abstract classes can have constructors and defined methods, whereas interfaces only define method signatures (before Java 8).

17. **What is the difference between == and equals()?**
    == compares object references, while equals() checks content equality.

18. **What is the final keyword?**
    final prevents modification: final variables cannot be reassigned, final methods cannot be overridden, and final classes cannot be extended.

19. **What is a constructor?**
    A constructor is a special method used to initialize an object and runs automatically when an instance is created.

20. **What are static methods and variables?**
    Static methods and variables belong to the class rather than instances and can be accessed without creating an object.

21. **Difference between static and instance methods?**
    Static methods belong to the class and cannot access instance variables, whereas instance methods work on object-specific data.

22. **What is the this keyword?**
    this refers to the current object instance and is used to differentiate between instance variables and parameters.

23. **What is the super keyword?**
    super is used to call parent class constructors and methods.

24. **What is an interface in Java?**
    An interface is a contract that defines method signatures without implementations, allowing multiple inheritance.

25. **What is multiple inheritance, and does Java support it?**
    Java does not support multiple class inheritance but allows multiple interface implementations.

26. **What is an abstract class?**
    An abstract class cannot be instantiated and can have both abstract and non-abstract methods.

27. **Difference between an interface and an abstract class?**
    An abstract class can have constructors, fields, and methods, while an interface only had method signatures before Java 8.

28. **What is the difference between Array and ArrayList?**
    Arrays have fixed sizes, while ArrayLists are dynamic and provide built-in methods for manipulation.

29. **What is the difference between HashMap and HashTable?**
    HashMap is unsynchronized and allows null keys/values, whereas HashTable is synchronized and does not allow nulls.

30. **What is the difference between ArrayList and LinkedList?**
    ArrayList is better for fast access (random access), while LinkedList is better for insertions/deletions.

31. **What is Exception Handling in Java?**
    Exception handling manages runtime errors using try-catch blocks to prevent abnormal program termination.

32. **Difference between checked and unchecked exceptions?**
    Checked exceptions (IOException) are handled at compile-time, while unchecked exceptions (NullPointerException) occur at runtime.

33. **What is the purpose of the finally block?**
    The finally block executes after try-catch, ensuring cleanup operations run regardless of exceptions.

34. **What is a try-with-resources statement?**
    It ensures automatic resource management (like closing files) using AutoCloseable.

35. **What is the difference between throw and throws?**
    throw is used to explicitly throw an exception, while throws declares exceptions in the method signature.

36. **What is multithreading in Java?**
    Multithreading allows concurrent execution of multiple threads for better performance.

37. **What is the difference between Thread and Runnable?**
    Thread is a class, whereas Runnable is an interface that allows a class to implement multithreading.

38. **What is synchronization in Java?**
    Synchronization controls access to shared resources in multithreading to prevent data inconsistency.

39. **What is a deadlock in Java?**
    A deadlock occurs when two or more threads wait indefinitely for resources held by each other.

40. **What is the difference between String, StringBuffer, and StringBuilder?**
    String is immutable, StringBuffer is mutable and synchronized, and StringBuilder is mutable but not synchronized.

41. **What is garbage collection in Java?**
    Garbage collection automatically reclaims unused memory to prevent memory leaks.

42. **What are weak references in Java?**
    Weak references allow objects to be garbage collected when memory is low, even if they have references.

43. **What is the difference between Stack and Queue?**
    Stack follows LIFO (Last In, First Out), while Queue follows FIFO (First In, First Out).

44. **What is the difference between HashSet and TreeSet?**
    HashSet does not maintain order, whereas TreeSet stores elements in sorted order.

45. **What is a lambda expression?**
    A lambda expression is a concise way to write anonymous functions in Java 8.

46. **What is a functional interface?**
    A functional interface has exactly one abstract method and is used for lambda expressions.

47. **What is the Optional class in Java?**
    Optional is a container object introduced in Java 8 to handle null values safely.

48. **What is the Stream API?**
    The Stream API processes collections using functional programming, supporting filtering, mapping, and reduction.

49. **What is dependency injection in Spring?**
    Dependency injection is a design pattern where dependencies are injected into a class rather than being created inside it.

50. **What is Spring Boot?**
    Spring Boot simplifies Spring applications by providing auto-configuration, embedded servers, and minimal setup.

Java Collections and Data Structures

51. **What is the difference between List and Set?**
    List allows duplicate elements and maintains order, whereas Set does not allow duplicates and may not maintain order.

52. **What are the different types of collections in Java?**
    Java Collections include List, Set, Queue, and Map interfaces, each with various implementations.

53. **What is the difference between HashSet and LinkedHashSet?**
    HashSet does not maintain insertion order, while LinkedHashSet maintains the order of elements.

54. **What is the difference between TreeMap and HashMap?**
    TreeMap maintains keys in sorted order, while HashMap does not.

55. **What is the difference between ConcurrentHashMap and HashMap?**
    ConcurrentHashMap is thread-safe and allows concurrent access, whereas HashMap is not thread-safe.

56. **What is a PriorityQueue in Java?**
    PriorityQueue is a special queue where elements are ordered based on natural ordering or a comparator.

57. **What is the difference between Comparable and Comparator?**
    Comparable is used for natural ordering (implements compareTo), while Comparator is used for custom sorting (compare method).

58. **What is a Deque in Java?**
    Deque (Double-Ended Queue) allows insertion and deletion at both ends.

59. **What is the difference between Stack and ArrayDeque?**
    Stack is synchronized and legacy, whereas ArrayDeque is faster and recommended for stack operations.

60. **What is the difference between a shallow copy and a deep copy?**
    A shallow copy copies object references, while a deep copy duplicates objects entirely.

Java Memory Management & Performance

61. **What is Heap Memory in Java?**
    Heap memory stores objects and is managed by the garbage collector.

62. **What is Stack Memory in Java?**
    Stack memory stores method calls and local variables.

63. **What is OutOfMemoryError in Java?**
    OutOfMemoryError occurs when JVM runs out of memory due to excessive object creation or memory leaks.

64. **What are memory leaks in Java?**
    Memory leaks happen when objects remain referenced unintentionally, preventing garbage collection.

65. **What is JVM tuning?**
    JVM tuning involves optimizing memory settings (-Xms, -Xmx), garbage collection, and performance parameters.

66. **What are strong, weak, soft, and phantom references?**
    - **Strong**: Prevent garbage collection.
    - **Weak**: Collected when memory is low.
    - **Soft**: Collected before OutOfMemoryError.
    - **Phantom**: Used for finalization before collection.

67. **What are the different garbage collectors in Java?**
    Java provides Serial GC, Parallel GC, G1 GC, and ZGC for memory management.

68. **How to prevent memory leaks in Java?**
    Use weak references, close resources, avoid static references, and profile memory usage.

69. **What is a ThreadLocal variable?**
    ThreadLocal provides variables that are isolated per thread, preventing shared state issues.

70. **What are immutable objects, and why are they useful?**
    Immutable objects (e.g., String) cannot be modified after creation, ensuring thread safety and better caching.

Multithreading & Concurrency

71. **What are the different thread states in Java?**
    NEW, RUNNABLE, BLOCKED, WAITING, TIMED_WAITING, and TERMINATED.

72. **What is the difference between sleep() and wait()?**
    sleep() pauses a thread for a specific time, while wait() releases a lock and waits for a signal.

73. **What is the difference between notify() and notifyAll()?**
    notify() wakes one waiting thread, while notifyAll() wakes all waiting threads.

74. **What is a synchronized block?**
    A synchronized block ensures that only one thread executes the critical section at a time.

75. **What is the volatile keyword?**
    volatile ensures a variable is read from main memory, preventing CPU caching issues in multithreading.

76. **What is the difference between Callable and Runnable?**
    Callable returns a result (Future<T>), while Runnable does not return any value.

77. **What is a ThreadPool in Java?**
    ThreadPool manages a pool of worker threads for efficient task execution.

78. **What is a ReentrantLock in Java?**
    ReentrantLock allows more control over synchronization compared to synchronized methods/blocks.

79. **What is an Executor Framework?**
    It manages and controls thread execution using ExecutorService.

80. **What is the Fork/Join framework?**
    It is a parallel computing framework for dividing tasks into smaller sub-tasks and merging results.

Java 8+ Features

81. **What is the purpose of the default method in interfaces?**
    Default methods allow adding new methods to interfaces without breaking existing implementations.

82. **What is the purpose of static methods in interfaces?**
    Static methods allow utility methods inside interfaces.

83. **What is a stream pipeline?**
    A stream pipeline consists of a source, intermediate operations, and a terminal operation.

84. **What is the difference between findFirst() and findAny()?**
    findFirst() returns the first element, while findAny() returns any matching element in parallel streams.

85. **What are Collectors in Java 8?**
    Collectors provide methods to accumulate elements from streams (e.g., Collectors.toList()).

86. **What is the difference between map() and flatMap()?**
    map() transforms elements individually, whereas flatMap() flattens nested structures.

87. **What is method reference in Java 8?**
    A method reference is a shorthand for lambda expressions (ClassName::methodName).

88. **What is a Predicate in Java?**
    A Predicate is a functional interface that tests conditions (boolean test(T t)).

89. **What is the difference between Optional.of() and Optional.ofNullable()?**
    Optional.of() throws NullPointerException for null values, while Optional.ofNullable() allows null values.

90. **What is the difference between Spliterator and Iterator?**
    Spliterator supports parallel processing, whereas Iterator is sequential.

91. **What is the purpose of the forEachRemaining() method in Iterator?**

    It processes remaining elements sequentially without needing explicit iteration.

92. **What is the difference between count() and size() in Streams?**

    count() returns elements in a stream, while size() works on collections.

93. **What is the advantage of using Optional in Java 8?**

    It prevents NullPointerException by handling missing values safely.

94. **What is the difference between Stream.of() and Arrays.stream()?**

    Stream.of() creates a stream from elements, while Arrays.stream() works on arrays.

95. **What is the purpose of the takeWhile() method in Streams?**

    It returns elements from the stream until a condition becomes false.

96. **What is the difference between sorted() and Comparator.reverseOrder()?**

    sorted() sorts naturally, while Comparator.reverseOrder() sorts in reverse.

97. **What are default methods used for in functional interfaces?**

    They allow adding methods without breaking existing implementations.

98. **What is the purpose of peek() in Streams?**

    peek() is used for debugging by inspecting elements without modifying them.

99. **What is a BiFunction in Java?**

    BiFunction takes two arguments and returns a result (apply(T, U) → R)

100. **What is the difference between limit() and skip() in Streams?**

    limit() restricts elements, while skip() discards the first N elements.