1) What do you mean by platform independence? How is it achieved in Java?

Java is platform-independent because programs written in Java can run on any operating system without modification. This is achieved using the Java Virtual Machine (JVM), which converts Java bytecode into machine-specific code.

2) Why is Java not a pure object-oriented language?

Java is not purely object-oriented because it uses primitive data types (int, float, char, etc.) that are not objects.

3) Tell us something about JIT compiler.

The Just-In-Time (JIT) compiler improves Java performance by converting bytecode into native machine code at runtime.

4) What is Object Oriented Programming (OOPs)?

OOP is a programming paradigm that organizes code into objects (real-world entities). It focuses on reusability and modularity.

5) What is a Class?

A class is a blueprint for creating objects. Example: A Car class defines attributes (color, speed) and methods (drive, stop).

6) What is an Object?

An object is an instance of a class. Example: 'myCar' is an object of the Car class.

7) What is a getter method and a setter method?

Getters are used to read private variables, and setters are used to modify them. Example: getName(), setName().

8) What is abstraction?

Abstraction means hiding implementation details and showing only the functionality. Example: A driver uses a car without knowing how the engine works.

9) What is encapsulation?

Encapsulation is wrapping data (variables) and code (methods) together. It is achieved using private variables and public getters/setters. Example: A bank account class keeps balance private but provides deposit/withdraw methods.

10) Difference between Abstraction and Encapsulation?

Abstraction hides implementation; Encapsulation hides data. Abstraction is about 'what,' Encapsulation is about 'how'.

11) Difference between equals() and == in Java?

equals() checks content equality, while == checks reference (memory address) equality.

12) What is Constructor?

A constructor is a special method used to initialize objects. It has the same name as the class.

13) What is Constructor Overloading?

Having multiple constructors with different parameter lists in the same class.

14) Define Copy Constructor in Java.

A constructor that creates a new object as a copy of another object.

15) Difference between instance variable and class variable?

Instance variables belong to objects; class variables (declared with static) are shared across all objects.

16) Purpose of 'this' keyword in Java?

this is used to refer to the current object, often used to distinguish between instance variables and parameters.

17) Significance of 'static' keyword in Java?

Static variables are shared, static methods belong to the class, and static blocks are used for initialization.

18) Explain exception handling in Java.

It is a mechanism to handle runtime errors using try, catch, finally, and throw keywords.

19) Use of throws keyword in Java?

The throws keyword declares exceptions a method can throw, so the caller knows to handle them.

20) What is Inheritance?

Inheritance allows a class to acquire properties/methods of another class. Example: Dog class inherits from Animal class.

21) What is Polymorphism?

Polymorphism means 'many forms.' Types: Compile-time (method overloading) and Runtime (method overriding).

22) Explain access specifiers in Java.

public (accessible everywhere), private (within class), protected (package + subclasses), default (package only).

23) What is method overloading?

Defining multiple methods with the same name but different parameter lists.

24) Difference between overloading and overriding?

Overloading: same method name, different parameters (compile-time). Overriding: same method name/signature, redefined in subclass (runtime).

25) Types of Inheritance in Java?

Single, Multilevel, Hierarchical, Hybrid (via interfaces). Java doesn’t support multiple inheritance directly.

26) What is an interface?

An interface is a collection of abstract methods (from Java 8, it can also have default and static methods).

27) What is an abstract class?

A class with abstract (unimplemented) methods that cannot be instantiated directly.

28) Abstract class vs Interface?

Abstract class can have both abstract and concrete methods; interface has only abstract methods (until Java 7).

29) Is it always necessary to create objects from class?

No, if methods are static, we can call them without creating an object.

30) Difference between structure and class in C++?

Structure: members are public by default, supports no OOP features. Class: members are private by default, supports OOP features.

31) Use of final keyword in variable, method, class?

final variable: value cannot change; final method: cannot be overridden; final class: cannot be inherited.

32) Difference between final, finally, and finalize?

final: constant or restriction, finally: block for cleanup code, finalize(): method called by Garbage Collector.

33) Use of super keyword in Java?

super is used to refer to the parent class, access parent variables, methods, and constructors.

34) Difference between static methods, static variables, and static classes?

Static methods belong to class, static variables are shared, static classes (inner) can be used without an outer object.

35) What is garbage collection?

Process by which Java automatically removes unused objects from memory.

36) What is a singleton class?

A class that allows only one object to be created. Implemented using private constructor and a static instance method.

37) What is multithreading in Java?

Executing multiple threads simultaneously to improve performance. Implemented using Thread class or Runnable interface.

38) Java thread lifecycle?

States: New, Runnable, Running, Waiting, Terminated.

39) Java works as pass by value or reference?

Java is strictly pass-by-value. For objects, the reference is passed by value.

40) What are collections in Java?

Collections are frameworks that provide data structures like List, Set, Queue, Map. Example: ArrayList, HashSet, HashMap.