**🔹 1. What is Java?**

**Answer:**  
Java is a high-level, object-oriented programming language developed by Sun Microsystems. It is platform-independent due to the Java Virtual Machine (JVM).

**🔹 2. What are the main features of Java?**

**Answer:**

* Platform Independent
* Object-Oriented
* Robust
* Secure
* Multithreaded
* High performance (with JIT)

**🔹 3. What is JVM, JRE, and JDK?**

**Answer:**

* **JVM:** Java Virtual Machine, runs Java bytecode
* **JRE:** Java Runtime Environment, contains JVM + libraries
* **JDK:** Java Development Kit, contains JRE + development tools (compiler, debugger)

**🔹 4. Why is Java platform independent?**

**Answer:**  
Because Java code is compiled into bytecode which runs on the JVM. The JVM is platform-specific, but bytecode is the same for all platforms.

**🔹 5. What is the difference between == and .equals() in Java?**

**Answer:**

* == compares references (memory addresses)
* .equals() compares values (contents)

**🔹 6. What are data types in Java?**

**Answer:**

* **Primitive:** int, char, boolean, float, double, byte, short, long
* **Non-Primitive:** Arrays, Strings, Objects

**🔹 7. What is a constructor in Java?**

**Answer:**  
A constructor is a special method used to initialize objects. It has the same name as the class and no return type.

**🔹 8. What is the difference between a constructor and a method?**

**Answer:**

* Constructor initializes an object
* Method performs actions
* Constructor has no return type; method does

**🔹 9. What is OOPs in Java?**

**Answer:**  
OOPs stands for Object-Oriented Programming, and its main concepts are:

* Encapsulation
* Inheritance
* Polymorphism
* Abstraction

**🔹 10. What is Inheritance?**

**Answer:**  
Inheritance allows one class (child) to inherit properties and methods from another class (parent). Achieved using extends keyword.

**🔹 11. What is Polymorphism?**

**Answer:**  
Polymorphism means one thing behaving differently in different contexts.

* **Compile-time (Method Overloading)**
* **Run-time (Method Overriding)**

**🔹 12. What is Method Overloading?**

**Answer:**  
When multiple methods in the same class have the same name but different parameters.

**🔹 13. What is Method Overriding?**

**Answer:**  
When a subclass provides its own implementation of a method defined in its superclass.

**🔹 14. What is Abstraction?**

**Answer:**  
Abstraction means hiding complex implementation and showing only necessary details. Achieved using abstract classes or interfaces.

**🔹 15. What is Encapsulation?**

**Answer:**  
Encapsulation means wrapping data and code together. Achieved using private variables and public getters/setters.

**🔹 16. What is the difference between abstract class and interface?**

**Answer:**

| **Abstract Class** | **Interface** |
| --- | --- |
| Can have method body | Only abstract methods (until Java 8) |
| Supports constructors | No constructors |
| Can have non-static variables | Only static and final variables |

**🔹 17. What is the static keyword?**

**Answer:**  
static means the member belongs to the class rather than an instance. It's shared among all objects.

**🔹 18. What is the final keyword?**

**Answer:**

* **final variable:** value can't be changed
* **final method:** can't be overridden
* **final class:** can't be extended

**🔹 19. What is a wrapper class in Java?**

**Answer:**  
Java provides object wrappers for primitive types (e.g., int → Integer, char → Character).

**🔹 20. What is Autoboxing and Unboxing?**

**Answer:**

* **Autoboxing:** converting primitive to wrapper
* **Unboxing:** converting wrapper to primitive

**🔹 21. What is the difference between Heap and Stack memory?**

**Answer:**

* **Heap:** Stores objects
* **Stack:** Stores method calls and local variables

**🔹 22. What is garbage collection in Java?**

**Answer:**  
Automatic memory management where the JVM removes unused objects from memory.

**🔹 23. What is the this keyword?**

**Answer:**  
Refers to the current object of the class.

**🔹 24. What is the super keyword?**

**Answer:**  
Refers to the parent class object and is used to access parent methods or constructors.

**🔹 25. What is an array in Java ?**

**Answer:**  
Array is a fixed-size data structure that holds elements of the same type.

**🔹 26. What is the difference between ArrayList and LinkedList?**

**Answer:**

* **ArrayList:** Fast in random access, slow in insertion/deletion
* **LinkedList:** Fast in insertion/deletion, slow in access

**🔹 27. What is the difference between ArrayList and Vector?**

**Answer:**

* ArrayList is not synchronized (faster)
* Vector is synchronized (thread-safe but slower)

**🔹 28. What is the difference between HashMap and Hashtable?**

**Answer:**

* HashMap: Not synchronized, allows null
* Hashtable: Synchronized, no null keys or values

**🔹 29. What is the difference between == and equals() for strings?**

**Answer:**  
== compares reference, .equals() compares actual content.

**🔹 30. What is a String in Java?**

**Answer:**  
A sequence of characters, immutable (cannot be changed once created).

**🔹 31. What is StringBuffer and StringBuilder?**

**Answer:**

* **StringBuffer:** Mutable, thread-safe.
* **StringBuilder:** Mutable, not thread-safe (faster)

**🔹 32. What is exception handling in Java?**

**Answer:**  
Managing runtime errors using try, catch, finally, and throw.

**🔹 33. What is the difference between throw and throws?**

**Answer:**

* throw: Used to throw an exception manually
* throws: Declares an exception in method signature

**🔹 34. What are checked and unchecked exceptions?**

**Answer:**

* **Checked:** Checked at compile-time (e.g., IOException)
* **Unchecked:** Checked at runtime (e.g., NullPointerException)

**🔹 35. What is a thread in Java?**

**Answer:**  
A thread is a lightweight process used for multitasking. Java supports multithreading using Thread class or Runnable interface.

**🔹 36. How to create a thread in Java?**

**Answer:**

* Extend Thread class and override run()
* Implement Runnable interface and override run()

**🔹 37. What is synchronization in Java?**

**Answer:**  
It controls thread access to resources to avoid data inconsistency.

**🔹 38. What is a deadlock in Java?**

**Answer:**  
When two or more threads are waiting for each other indefinitely to release resources.

**🔹 39. What are access modifiers in Java?**

**Answer:**

* **private** – within class
* **default** – within package
* **protected** – package + subclass
* **public** – everywhere

**🔹 40. Difference between compile-time and runtime polymorphism?**

**Answer:**

* **Compile-time (Overloading):** Method is resolved during compilation
* **Runtime (Overriding):** Method is resolved during execution.

**🔹 1. What is the Collection framework in Java?**

**Answer:**  
The Collection framework is a set of **classes and interfaces** in Java used to **store, retrieve, and manipulate groups of objects** (like lists, sets, maps). It provides **ready-to-use data structures** such as ArrayList, HashSet, HashMap, etc.

**🔹 2. What is the difference between Collection and Collections?**

**Answer:**

* Collection is an **interface** (e.g., List, Set extend it).
* Collections is a **utility class** with static methods like sort(), reverse(), shuffle(), etc.

**🔹 3. What is the difference between List and Set?**

**Answer:**

* **List** allows **duplicate elements** and maintains **insertion order** (e.g., ArrayList, LinkedList).
* **Set** does **not allow duplicates** and does **not guarantee order** (e.g., HashSet, TreeSet).

**🔹 4. What is the difference between ArrayList and LinkedList?**

**Answer:**

* ArrayList: Fast for **searching**, slow for **inserting/deleting** in the middle.
* LinkedList: Fast for **inserting/deleting**, slower for **searching**.

**🔹 5. What is the difference between HashMap and Hashtable?**

**Answer:**

* HashMap is **not synchronized**, allows **one null key** and **multiple null values**.
* Hashtable is **synchronized**, does **not allow null key or value**.

**🔹 6. How does HashSet prevent duplicate elements?**

**Answer:**  
HashSet uses a **HashMap internally**. It checks **hashCode()** and **equals()** of objects to ensure uniqueness.

**🔹 7. What is the difference between Iterator and ListIterator?**

**Answer:**

* Iterator: Works with **all collections**, **forward** direction only.
* ListIterator: Works only with **List**, supports **forward and backward** traversal.

**🔹 8. What is the difference between HashSet and TreeSet?**

**Answer:**

* HashSet: Does **not maintain order**, faster, uses **hashing**.
* TreeSet: Maintains elements in **sorted order**, slower, uses **Red-Black Tree**.

**🔹 9. What are the main interfaces of the Collection framework?**

**Answer:**

* Collection (base)
  + List
  + Set
  + Queue
* Map (not part of Collection but used in framework)

**🔹 10. How do you sort a list in Java?**

**Answer:**  
Use the Collections.sort() method:

java

CopyEdit

List<String> names = new ArrayList<>();

Collections.sort(names);