UnCodemy

Java Interview Questions PDF

Core Java interviews cover object-oriented principles, the difference between JDK, JRE and JVM, strings and immutability, collections such as ArrayList and HashMap, exceptions, multithreading, memory management and Java 8 features like lambdas and streams. This page gives short model answers with small code samples for practice.

Key takeaways

- Java source compiles to bytecode that the JVM runs on any platform.
- The four OOP pillars are encapsulation, inheritance, polymorphism and abstraction.
- Strings are immutable; use StringBuilder for many changes.
- Know how HashMap, ArrayList and HashSet work and when to use each.
- Understand checked and unchecked exceptions, threads and the memory areas of the JVM.

What is Java interview questions?

Java is a general-purpose, object-oriented programming language that compiles to bytecode and runs on the Java Virtual Machine, which lets the same program run on different operating systems.

What are JDK, JRE and JVM?

The **JVM** runs bytecode and provides memory management. The **JRE** is the JVM plus libraries needed to run programs. The **JDK** is the JRE plus tools such as the compiler (`javac`) to build programs. Java is described as platform independent because bytecode runs on any machine with a JVM.

What are the OOP pillars?

Encapsulation: keep data private and expose it through methods. **Inheritance:** a class reuses another with `extends` . **Polymorphism:** the same method name behaves differently, through overloading (compile time) and overriding (run time). **Abstraction:** hide details behind abstract classes and interfaces.

Abstract class or interface?

An abstract class can hold state and implemented methods and is for related classes sharing code. An interface defines a contract that unrelated classes can implement; since Java 8 it can have default and static methods. A class can extend one class but implement many interfaces.

Overloading versus overriding?

Overloading uses the same method name with different parameters in one class. Overriding redefines an inherited method in a subclass with the same signature, and is chosen at run time.

Why are strings immutable?

A `String` cannot change after creation, which makes it safe to share, cache in the string pool and use as a map key. Concatenating in a loop creates many objects, so use `StringBuilder` (not thread safe) or `StringBuffer` (thread safe).

```
String a = "java";
String b = "java";
System.out.println(a == b);        // true (same pool object)
System.out.println(a.equals(b));   // true (same content)
```

`==` compares references, and `equals` compares content.

How do collections compare?

Type	Key traits
ArrayList	Resizable array, fast index access, slow insert in the middle
LinkedList	Fast insert and remove at ends, slow index access
HashSet	Unique items, no order, fast lookup
TreeSet	Unique items, sorted
HashMap	Key-value pairs, one null key, no order
TreeMap	Sorted by key
ConcurrentHashMap	Thread-safe map

How does HashMap work?

It stores entries in buckets chosen by the key's hash code. Equal keys must have equal hash codes, so override `equals` and `hashCode` together. Collisions are handled in the same bucket with a list or a tree. Lookup is on average constant time.

What are checked and unchecked exceptions?

Checked exceptions (for example `IOException`) must be caught or declared. Unchecked exceptions (`RuntimeException` and subclasses such as `NullPointerException`) need not be. Use `try`, `catch`, `finally` and try-with-resources, which closes resources automatically.

```
try (BufferedReader r = new BufferedReader(new FileReader("a.txt"))) {
    System.out.println(r.readLine());
} catch (IOException e) {
    System.out.println("Could not read: " + e.getMessage());
}
```

How does multithreading work?

Create threads by extending `Thread` or implementing `Runnable`, and prefer executors for managing pools. Threads share memory, so use `synchronized`, locks or concurrent collections to avoid race conditions. Know deadlock (threads wait for each other forever) and how to avoid it by ordering locks.

How does JVM memory work?

The **heap** holds objects and is managed by garbage collection. The **stack** holds method calls and local variables for each thread. Class metadata lives in a separate area. Garbage collection frees objects that are no longer reachable; you cannot force it, only request it.

What are Java 8 features?

Lambdas, functional interfaces, streams, `Optional`, default methods and the new date and time API.

```
List<Integer> nums = List.of(1, 2, 3, 4, 5, 6);
int sumOfEvenSquares = nums.stream().filter(n -> n % 2 == 0).mapToInt(n -> n * n).sum();
```

What are static, final and this?

`static` belongs to the class, `final` prevents change (a variable, a method override or inheritance) and `this` refers to the current object.

What coding questions are common?

Typical exercises are reversing a string, checking a palindrome, finding duplicates in an array, counting character frequency with a map, removing duplicates from a list, finding the second largest number and writing a simple class hierarchy. Talk through your approach, choose the right collection, handle null and empty input and mention time complexity.

How do I explain design principles?

Mention SOLID in simple terms: a class should have one responsibility, be open for extension but closed for modification, subclasses should be usable in place of parents, interfaces should be small and code should depend on abstractions. Give one small example for any principle you describe.

What should I say about Java versions?

Know that Java has regular releases and that long-term support versions are common in companies. Mention features you have used, such as records, text blocks or switch expressions, only if you understand them. Check the official release notes for the current versions before an interview.

Frequently asked questions

What is the difference between JDK, JRE and JVM?

The JVM runs bytecode, the JRE adds the libraries to run programs and the JDK adds development tools such as the compiler.

Why is the String class immutable?

Immutability makes strings safe to share, cache in the string pool and use as map keys.

What is the difference between == and equals?

== compares references, while equals compares content when the class overrides it, as String does.

Abstract class or interface?

An abstract class shares state and code among related classes, while an interface defines a contract that unrelated classes can implement.

How does HashMap work?

It places entries in buckets chosen by the key hash code, and equal keys must produce equal hash codes.

Sources and further reading

- Java documentation
- OpenJDK

About this PDF

These PDFs are study material for learning and revision. They are not a certificate, a degree or a promise of a job, and technology changes, so check official documentation for the latest details. Spot an error? Write to info@uncodemy.com and the trainer team will review it.

More free resources and courses: uncodemy.com | info@uncodemy.com | +91 98183 66550