1. what are the features of java ?

platform independence (Write Once, Run Anywhere), object-oriented design, simplicity, robustness, and high performance.

1. **What are the four pillars of OOP?**

Inheritance, Encapsulation, Polymorphism, Abstraction.

1. **Difference between method overloading and overriding?**

Overloading: same method name, different parameters (compile-time).

Overriding: same method signature in subclass (runtime).

1. **What is encapsulation and how does Java achieve it?**

Wrapping data and methods; using private fields and public getters/setters.

1. What is the difference between == and .equals() in Java?

== compares values for primitives and references for objects.

.equals() compares content for objects (when overridden) and references by default.

1. What are wrapper classes? Why are they used?

Wrapper classes in Java allow primitive data types (like int, char, boolean) to be treated as objects. They are used to enable functionality like using primitives with collections (which require objects), and methods requiring objects as parameters. Autoboxing and unboxing simplify the conversion between primitives and their wrapper class equivalents.

1. What is the role of final, finally, and finalize()?

The **final** keyword is used to declare constants, prevent method overriding, and prevent inheritance.

final int MAX = 100;

MAX = 200; // Error

The **finally** block is used in **exception handling** to execute important code such as closing resources. It always runs, whether an exception is thrown or not.

The **finalize()** method was used to perform cleanup actions before an object is garbage collected. – deprecated.

1. What is the difference between String, StringBuilder, and StringBuffer?

In Java, String, StringBuilder, and StringBuffer are classes used to handle text (sequences of characters), but they differ significantly in **mutability**, **performance**, and **thread safety**.

|  |  |  |  |
| --- | --- | --- | --- |
| **Feature** | **String** | **StringBuilder** | **StringBuffer** |
| Mutability | Immutable | Mutable | Mutable |
| Thread-safe | Yes (immutable) | ❌ No | ✅ Yes (synchronized) |
| Performance | Slowest | Fastest | Slower than StringBuilder |
| Use case | Constant text | Fast local changes | Safe concurrent changes |