INHERITANCE IN JAVA

1.What is inheritance in Java?

Inheritance allows a subclass to inherit data (fields) and behavior (methods) from a superclass, creating a hierarchical relationship where the subclass specializes or extends the functionality of the superclass.

2.What are the benefits of using inheritance?

Code reusability: Reuse existing code from the superclass in the subclass, reducing redundancy.
Polymorphism: Enables dynamic method dispatch based on the actual object type at runtime.
Code organization: Helps organize code logically based on class hierarchies.

3.What are the types of inheritance in Java?

Single inheritance: A subclass inherits from one superclass.
Multilevel inheritance: A subclass inherits from another subclass, forming a chain of inheritance.
Hierarchical inheritance: Multiple subclasses inherit from the same superclass.
Interface inheritance: Interfaces can inherit other interfaces to extend their functionality.

4.What are the key concepts in inheritance?

Subclasses: Classes that inherit from other classes.
Superclass: The class from which a subclass inherits.
Inheritance hierarchy: The tree-like structure showing inheritance relationships between classes.
Method overriding: When a subclass provides its own implementation of a method inherited from the superclass.
Method overloading: Having multiple methods with the same name but different parameter lists within the same class.

5.What is the difference between inheritance and

composition?

Inheritance creates an "is-a" relationship (e.g., Car is-a Vehicle), while composition uses objects of one class as members of another (e.g., Car has-a Engine).

6.What are the limitations of inheritance?

The "diamond problem" in multiple inheritance.
Tight coupling between classes due to inheritance relationships.
Potential fragility if superclass changes break subclasses.

7.When should you not use inheritance?

When classes don't share a logical "is-a" relationship.
When composition might be more flexible and maintainable.

8.Explain the concept of access modifiers and their role in inheritance.

Access modifiers (public, protected, private) control access to inherited members.
protected members are accessible by subclasses but not outside the inheritance hierarchy.

9.Describe constructor chaining in inheritance.

A subclass constructor can call other constructors within the same class or the superclass using this() and super().

10.How does inheritance relate to design patterns like the Strategy pattern and the Template Method pattern?

Inheritance can be used to implement different variations of an algorithm in subclasses for the Strategy pattern.
Inheritance can provide a common framework in the superclass with specific steps implemented in subclasses for the Template Method pattern.