INTERVIEW QUESTIONS

1. **What is abstraction?**
   "Abstraction is the process of hiding complex implementation details and showing only the essential features of an object or system."
2. **Difference between interface and abstract class?**
   "An interface defines a contract for behavior without any implementation, allowing a class to implement multiple interfaces. An abstract class provides partial implementation and shared code, but a class can extend only one abstract class. So, interfaces are used for capabilities, while abstract classes are used for base structure and logic."
3. **Explain polymorphism with example?**
   Polymorphism means "many forms." In object-oriented programming, it allows one interface or method to behave differently based on the object that's using it. It makes code more flexible, reusable, and easier to maintain.
   Example:

```
class Animal {
   void sound() {
      System.out.println("Animal makes a sound");
   }
}

class Dog extends Animal {
   void sound() {
      System.out.println("Dog barks");
   }
}

class Cat extends Animal {
   void sound() {
      System.out.println("Cat meows");
   }
}

public class TestPolymorphism {
   public static void main(String[] args) {
      Animal a1 = new Dog();  // Polymorphic reference
      Animal a2 = new Cat();

      a1.sound();  // Output: Dog barks
      a2.sound();  // Output: Cat meows
   }
}
```

4. **What is method overriding?**

Method overriding in Java occurs when a subclass provides its own implementation of a method that is already defined in its superclass**.** The method in the child class must have the same name, return type, and parameters as the one in the parent class. This enables runtime polymorphism, allowing Java to decide which method to execute based on the actual object type at runtime

5. **Explain "IS-A " vs " HAS-A "relationships.**
   Is-a represents inheritance between classes and has-a represent association or composition. Indicates that one class is a type of another and has-a represents that one class contains another class as a field. Example for IS-A relation is Dog is a animal . because dog inherits from animal. Example for HAS-A relation is car has a engine. Because car contains engine.
6. **Why use inheritance?**
   Inheritance allows a class (child/subclass) to acquire properties and behaviors (fields and methods) from another class (parent/superclass). This promotes code reuse**,** modularity**,** and extensibility**.**
7. **What is dynamic binding?**
   Dynamic binding (also called late binding) refers to the process where the method call is resolved at runtime rather than compile time. It's a key part of runtime polymorphism**,** allowing Java to decide which method to execute based on the actual object type—not the reference type.
8. **What is constructor chaining?**
   Constructor chaining in Java is the process of calling one constructor from another within the same class or between parent and child classes. It helps avoid code duplication and ensures proper initialization of objects.
9. **How to implement encapsulation?**
   "Encapsulation in Java is implemented by making class fields private and providing public getter and setter methods to access and modify them. This hides the internal state of the object and allows controlled access, which improves security and maintainability."
10. **Explain super keyword?**
    In Java, the super keyword is used to refer to the immediate parent class. It allows a subclass to access methods, constructors, or variables of its superclass. It's commonly used to call a parent class constructor or to invoke a method that has been overridden in the child class."