1. What are the benefits of Polymorphism?

- Polymorphism allows code to be reused, making it easier and faster to develop programs. Once classes are written and tested, they can be used multiple times without rewriting them. It also allows changes to be made in the subclass without affecting the original class.
- A single variable can hold different types of data. Subclasses can modify inherited variables without altering the same variables in the superclass or other subclasses.
- Writing fewer lines of code leads to simpler debugging and maintenance for developers.

2. How does Inheritance help enable Polymorphism in Java?

- Inheritance creates a class hierarchy, where subclasses are seen as specialized versions of their superclasses. This setup is crucial for enabling polymorphism in Java.
- When a subclass overrides a method and is referenced through a superclass type, Java determines the correct method at runtime.
- This allows objects of different subclasses to be treated uniformly using the superclass type, while still executing their specific behaviors.
- This approach simplifies handling groups of objects and leads to more general and reusable code.
- By coding to an abstract superclass rather than concrete subclasses, developers can create systems that are easier to modify and extend, with better modularity and less dependency.

3. What distinguishes Polymorphism from Inheritance in Java?

- **Concept**:
  Inheritance is the concept of one class gaining properties and methods from another.
  Polymorphism means the same method or interface can work differently depending on the object.
- **Purpose**:
  Inheritance is used for code reuse and defining “is-a” relationships.
  Polymorphism allows for dynamic behavior and flexible code execution.

- **Types**:
  Inheritance includes single, multilevel, and hierarchical types.
  Polymorphism includes compile-time (method overloading) and runtime (method overriding).
- **Focus**:
  Inheritance focuses on shared structure and functionality across related classes.
  Polymorphism emphasizes how the same method can perform differently based on the object calling it.
- **Interrelation**:
  Inheritance supports polymorphism by connecting classes.
  Polymorphism often relies on inheritance, especially for overriding methods.
- **Reusability vs. Flexibility**:
  Inheritance offers code reuse by passing down features.
  Polymorphism provides flexibility by allowing different object types to respond in their own way.