1. What are the advantages of Polymorphism?

- Flexibility and Scalability: Polymorphism allows methods to do different things based on the object it is acting upon, which makes the code more flexible and scalable.

- Code Reusability: It enables the reuse of code through methods in different classes that do similar things but with different inputs.

- Maintainability: Reduces complexity by allowing the same interface to be used for different underlying forms (data types). This makes the code easier to maintain and modify.

- Interchangeability and Extensibility: Objects of different classes can be treated as objects of a common superclass, especially useful in large projects where changing software behavior should be quick and efficient.

2. How is Inheritance useful to achieve Polymorphism in Java?

Inheritance is one of the mechanisms to achieve polymorphism. Here's how it works in Java:

- Inheritance allows a class to inherit properties and methods from another class, known as the superclass. This relationship enables the subclass to override or extend the functionality of its superclass.

- Method Overriding: This is a feature of inheritance where the child class has a method with the same name, return type, and parameters as a method in its parent class. When an overridden method is called from a child class object, the child class version of the method is executed, demonstrating polymorphism.

- Interface Implementation: Java allows classes to implement interfaces which is another way to achieve polymorphism. Classes that implement an interface must provide an implementation for all the methods defined in the interface, thus ensuring a common interface but varied implementations.

3. What are the differences between Polymorphism and Inheritance in Java?

Conceptual Difference:

- Inheritance is a mechanism wherein a new class is derived from an existing class. It focuses on creating a new class using existing classes.

- Polymorphism refers to the ability of different objects to respond to the same method call in different ways. It focuses on using interfaces and methods in diverse ways.

Usage:

- Inheritance is used to establish a relationship between the base class and the derived class where attributes and methods are inherited.

- Polymorphism is used to execute the same action in many different ways.

Purpose:

- Inheritance primarily aims to promote code reuse and establish a hierarchical relationship between classes.

- Polymorphism aims to utilize a unified interface for different underlying forms (data types).

Method Implementation:

- Inheritance may involve extending or modifying behaviors of a superclass in a subclass.

- Polymorphism involves calling the same method on different objects, each of which could belong to different classes, and having them behave differently.