ANSWER 6–

Polymorphism, in the context of object-oriented programming, is the ability of objects or classes to take on different forms or behave differently based on the context in which they are used. It allows objects or classes of different types to be treated as objects of a common superclass, providing flexibility and interchangeability. In simple words, polymorphism can be understood as follows:

Many Forms: Polymorphism refers to the ability of objects or classes to take on many forms or have many behaviors. It allows different objects or classes to respond differently to the same method call based on their specific implementation.

Method Overriding: Polymorphism is often achieved through method overriding. Method overriding occurs when a subclass provides its own implementation of a method that is already defined in its superclass. The method in the subclass overrides the behavior of the same-named method in the superclass.

Interface Compatibility: Polymorphism also allows different classes to implement the same interface or inherit from the same superclass, enabling them to be used interchangeably. This allows you to write code that can work with objects of different types as long as they adhere to a common interface or superclass.

Code Flexibility and Extensibility: The purpose of polymorphism is to provide code flexibility and extensibility. By treating objects of different types as instances of a common superclass or interface, you can write more generic and reusable code. This allows you to write code that can work with a variety of objects without needing to know their specific types.

Dynamic Binding: Polymorphism is often associated with dynamic binding or late binding. Dynamic binding means that the specific

method implementation to be executed is determined at runtime based on the actual type of the object, rather than at compile time. This allows for dynamic and flexible behavior based on the actual objects involved in the program.

The purpose of polymorphism is to promote code flexibility, reusability, and extensibility. It allows you to write more generic code that can work with objects of different types, reducing code duplication and promoting code organization. Polymorphism enables you to write code that focuses on the common behaviors and interfaces shared by objects, rather than their specific implementations, leading to more maintainable and scalable software systems