# **Documentation for question 2**

**Polymorphism in Object-Oriented Programming**

Polymorphism is a fundamental concept in object-oriented programming (OOP) that allows objects of different classes to be treated as objects of a common superclass. It enables a single interface to be used for other data types or objects, providing flexibility and extensibility in code.

In the Java code, several instances of polymorphism are demonstrated:

1. **Method overriding:**

Polymorphism is demonstrated through method overriding, where a subclass provides a specific implementation of a method already defined in its superclass. In the code, the ‘**Patient**’ and ‘**Playgroup**’ classes override the ‘**babySound()**’ method inherited from the ‘**Baby**’ class. Each subclass implements the ‘**babySound()**’ method, reflecting the specific sound associated with a patient or a baby in a playgroup.

1. **Dynamic Method Dispatch:**

Dynamic method dispatch is a mechanism by which a call to an overridden method is resolved at runtime rather than compile time. This allows the JVM to determine the appropriate method implementation to invoke based on the actual type of the object being referred to rather than the reference type. In the provided code, dynamic method dispatch ensures that the overridden ‘**babySound()**’ method in the respective subclass (‘**Patient**’ or ‘**Playgroup**’) is invoked at runtime when called through a reference of type ‘**Baby**’.