Course Code	**:**	**MCS-024**
Course Title	**:**	**Object Oriented Technologies and Java Programming**
Assignment Number	**:**	**BCA (IV)/024/Assignment/2026-27**
Maximum Marks	**:**	**100%**
Last Date of Submission	**:**	**31st October, 2026 (for July 2026 Session)**
	:	**30th April, 2027 (for January 2027 Session)**

There are Three questions in this assignment which carry 80 marks, in total. Rest 20 marks are for viva-voce. Answer all the questions. Give appropriate comments in programs to increase understandability. Wherever required, you may write java program, run it on machine and take its output as part of solution. Please go through the guidelines regarding assignments given in the Programme Guide for the format of presentation.

Q1: **(30 Marks)**

What is Object Oriented Programming (OOP)? Explain the four fundamental principles of OOP, namely, Encapsulation, Abstraction, Inheritance, and Polymorphism with suitable examples.

Q2: **(30 Marks)**

Explain the concept of Inheritance in Java. Describe the different types of inheritance supported by Java (single, multilevel, hierarchical) and explain why Java does not support multiple inheritance through classes. How is this limitation overcome using interfaces?

Q3: **(20 Marks)**

Explain multithreading in Java. Describe the life cycle of a thread with a neat diagram, and write a Java program to create a thread using both the **Thread** class and the **Runnable** interface.

Q1: (30 Marks) What is Object Oriented Programming (OOP)? Explain the four fundamental principles of OOP, namely, Encapsulation, Abstraction, Inheritance, and Polymorphism with suitable examples.

Answer

Object Oriented Programming (OOP) is a way of designing software around objects rather than around a long sequence of procedures. An object combines data that represents its state with methods that operate on that data. Objects are normally created from classes, where a class acts as a blueprint and an object is a particular instance of that blueprint. In Java, for example, a class Account may describe the data and operations common to bank accounts, while acc1 and acc2 are separate objects with their own balances.

OOP is useful when a program contains many related entities and their behaviour has to be organised in a clear and maintainable form. Instead of allowing every part of the program to manipulate all data directly, responsibilities are distributed among classes. This makes a large program easier to understand, test, extend and reuse. The four principles that give OOP this structure are encapsulation, abstraction, inheritance and polymorphism. They are related, but each solves a different design problem.

1. Encapsulation

Encapsulation means keeping the data of an object and the methods that work on that data together inside one class, while controlling how outside code can access the internal state. In Java this is commonly achieved with access modifiers such as private, protected and public. Important fields are often declared private, and public methods provide the permitted operations. This prevents unrelated code from placing an object in an invalid state.

Encapsulation is more than simply writing getter and setter methods. A class should protect its own rules. For example, a bank account should not allow a negative deposit. If the balance field were public, any statement could assign an impossible value. With a private field and a checked deposit() method, the class itself enforces the rule. The internal representation can also be changed later without forcing every user of the class to change, provided the public operations remain compatible.