

## SEMESTER-III

### COURSE 5: OOPS THROUGH JAVA

Theory

Credits: 3

3 hrs/week

---

#### Course Objectives

1. Introduce core OOP principles and contrast procedural and object-oriented paradigms within the Java ecosystem.
2. Equip learners with foundational and advanced Java syntax including variables, control statements, arrays, strings, and classes.
3. Enable the use of inheritance, polymorphism, interfaces, and exception handling to create maintainable and reusable code.
4. Train students in concurrent programming and stream-based I/O operations including file management and serialization.
5. Empower learners to design, build, and manage GUI programs using Swing components, layout managers, and event handling techniques.

#### Course Outcomes

At the end of the course, students will be able to:

1. Apply OOP principles such as encapsulation, inheritance, and polymorphism in Java applications.
2. Write, compile, and debug Java code using control statements, arrays, classes, and methods effectively.
3. Construct modular code leveraging interfaces, abstract classes, and package hierarchies.
4. Manage thread lifecycles, synchronization, and I/O streams for file handling and console interaction.
5. Design user interfaces using Swing and handle keyboard/mouse input through event-driven programming.

#### Unit 1. OOPs Concepts and Java Programming:

Introduction to Object-Oriented concepts, procedural and object-oriented programming paradigm

Java programming: An Overview of Java, Java Environment, Data types, Variables, constants, scope and life time of variables, operators, type conversion and casting, Accepting Input from the Keyboard, Reading Input with Java.util.Scanner Class, Displaying Output, Displaying Formatted Output with String.format(), Control Statements

#### Unit 2. Arrays and OOP Constructs:

Arrays, Command Line Arguments, Strings-String Class Methods

Classes & Objects: Creating Classes, declaring objects, Methods, parameter passing, static fields and methods, Constructors, and 'this' keyword, overloading methods and access Inheritance:

Inheritance hierarchies, super and subclasses, member access rules, 'super' keyword, preventing inheritance: final classes and methods, the object class and its methods; Polymorphism: Dynamic binding, method overriding, abstract classes and methods;

### **Unit 3. Interfaces, Packages & Exception Handling:**

Interfaces VS Abstract classes, defining an interface, implement interfaces, accessing implementations through interface references, extending interface;

Packages: Defining, creating and accessing a package, importing packages.

Exception Handling: Benefits of exception handling, the classification of exceptions, exception hierarchy, checked exceptions and unchecked exceptions, usage of try, catch, throw, throws and finally, rethrowing exceptions, exception specification, built in exceptions, creating own exception sub classes.

### **Unit 4. Multithreading & Stream based I/O:**

Differences between multiple processes and multiple threads, thread states, thread life cycle, creating threads, interrupting threads, thread priorities, synchronizing threads, inter thread communication.

Stream based I/O (java.io) – The Stream classes-Byte streams and Character streams, Reading console Input and Writing Console Output, File class, Reading and writing Files, The Console class, Serialization

### **Unit 5. GUI Programming with Swing:**

Introduction, MVC architecture, components, containers. Understanding Layout Managers - Flow Layout, Border Layout, Grid Layout, Card Layout, GridBag Layout. Event Handling- The Delegation event model- Events, Event sources, Event Listeners, Event classes, Handling mouse and keyboard events, Adapter classes.

### **Text Books:**

1. Java The complete reference, Herbert Schildt, 9th edition, McGraw Hill.
2. Programming in Java, S. Malhotra, S. Chudhary, 2nd edition, Oxford Univ. Press.

### **Reference Books:**

1. Programming with JAVA - A Primer, E Balaguruswamy, 3rd Edition, McGraw Hill
2. Head First Java: A Brain-Friendly Guide , Katty Sierra, Bert Bates, 2nd Edition, O'Reilly

### **Activities:**

**Outcome:** Apply OOP principles such as encapsulation, inheritance, and polymorphism in Java applications

**Activity:** Develop a class hierarchy for a zoo management system using inheritance and polymorphism (e.g., Animal → Mammal → Dog). Implement encapsulation through private fields and public getters/setters.

**Evaluation Method:** Code review and oral explanation focusing on class relationships, method overriding, and encapsulation practices.

**Outcome:** Write, compile, and debug Java code using control statements, arrays, classes, and methods effectively

**Activity:** Create a console-based student grade calculator using loops, conditionals, arrays, and modular methods.

**Evaluation Method:** Practical test with debugging tasks and output validation across multiple input scenarios.

**Outcome:** Construct modular code leveraging interfaces, abstract classes, and package hierarchies

**Activity:**

Design a payment processing system with abstract classes for Payment, interfaces for Taxable, and organize classes into packages (e.g., com.billing, com.tax).

**Evaluation Method:**

Project submission assessed for modularity, interface implementation, abstraction usage, and package structure.

**Outcome:** Manage thread lifecycles, synchronization, and I/O streams for file handling and console interaction

**Activity:** Build a multithreaded logger that reads input from the console and writes to a file using synchronized threads and buffered streams.

**Evaluation Method:** Lab demonstration with thread state tracing and file output verification under concurrent input.

**Outcome:** Design user interfaces using Swing and handle keyboard/mouse input through event-driven programming

**Activity:** Create a GUI-based quiz application using Swing components (JFrame, JButton, JTextField) with event listeners for mouse clicks and key presses.

**Evaluation Method:** Live demo and rubric-based assessment of UI responsiveness, event handling accuracy, and layout design.

## SEMESTER-III

### COURSE 5: OOPS THROUGH JAVA

**Practical**

**Credits: 1**

**2 hrs/week**

---

#### List of Experiments

1. Write a Java program to print Fibonacci series.
2. Write a Java program to calculate multiplication of 2 matrices.
3. Write a Java program for sorting a given list of names in ascending order.
4. Create a class Rectangle. The class has attributes length and width. It should have methods that calculate the perimeter and area of the rectangle. It should have readAttributes() method to read length and width from the user.
5. Write a Java program that implements method overloading.
6. Write a Java program to implement various types of inheritance
  - i. Single
  - ii. Multi-Level
  - iii. Hierarchical
  - iv. Hybrid
7. Write a java program to implement runtime polymorphism.
8. Write a Java program which accepts withdrawal amount from the user and throws an exception In Sufficient Funds when withdrawal amount is more than available amount.
9. Write a Java program to create three threads and that displays good morning, for every one second, hello for every 2 seconds and welcome for every 3 seconds by using extending Thread class.
10. Write a Java program that creates three threads. The first thread displays OOPS, the second thread displays Through and the third thread displays JAVA by using Runnable interface.
11. Write a Java program that displays the number of characters, lines and words in a text file.
12. Implement a Java program for handling mouse events when the mouse entered, exited, clicked, pressed, released, dragged and moved in the client area.
13. Implement a Java program for handling key events when the key board is pressed, released, typed.
14. Write a Java swing program that reads two numbers from two separate text fields and displays the sum of two numbers in the third text field when the button add is pressed.
15. Write a Java program to design student registration form using Swing Controls. The form which having the following fields and button SAVE  
Form Fields are: Name, RNO, Mailid, Gender, Branch, Address.