



IAS YOUTH COMPUTER CENTRE

DIPLOMA IN JAVA TECHNOLOGY

Practical Textbook • Java Programming • OOP • JDBC • Projects

88/1, Queen Apartment, 2nd Floor,
Dumdum Road, Near Life Line Medical Store,
Kolkata – 700030, West Bengal

Diploma Level • Practical & Job-Oriented Programming Training

About This Book

This practical textbook introduces Java programming from development-environment setup and language fundamentals to object-oriented programming, collections, file handling, JDBC, debugging and project development. The chapters are designed to progress from beginner programming skills to practical application development.

Learning Outcomes

- Understand Java technology, JDK, JRE, JVM and bytecode.
- Set up a Java development environment and run programs.
- Use variables, data types, operators and expressions.
- Write programs using conditions, loops, arrays and strings.
- Create reusable methods and modular programs.
- Apply object-oriented programming with classes and objects.
- Understand inheritance, polymorphism, abstraction and interfaces.
- Use packages, access modifiers and exception handling.
- Work with Java Collections Framework.
- Perform basic file input/output operations.
- Understand JDBC and database connectivity.
- Debug, test and document Java applications.
- Build and present a practical Java project.

Course Contents

No.	Chapter
1	Introduction to Java Technology
2	Java Platform, JDK, JRE & JVM
3	Setting Up the Java Development Environment
4	Java Syntax, Variables & Data Types
5	Operators & Expressions
6	Conditional Statements
7	Loops & Iteration
8	Arrays & String Handling
9	Methods & Modular Programming
10	Object-Oriented Programming Concepts
11	Inheritance & Polymorphism
12	Abstraction, Interfaces & Encapsulation
13	Packages, Access Modifiers & Exception Handling
14	Collections Framework
15	File Handling & Input/Output
16	JDBC & Database Connectivity
17	Java GUI, Web & Application Development Overview
18	Debugging, Testing, Security & Best Practices

No.	Chapter
19	Java Practical Projects & Portfolio
20	Viva, Glossary & Final Assessment

1. Introduction to Java Technology

Learning Outcomes

- Understand Java and its major characteristics.
- Understand the basic Java development lifecycle.

Key Concepts

Java is a general-purpose, object-oriented programming language and platform used for desktop, web, enterprise and many other software applications. This chapter introduces Java, its history, features and development workflow.

Practical Activities

- Identify JDK, JRE and JVM concepts.
- Prepare a simple Java development setup checklist.

Quick Revision: Review the Java syntax, programming concept or development workflow introduced in this chapter, then repeat the practical exercise and save the source code with clear filenames.

2. Java Platform, JDK, JRE & JVM

Learning Outcomes

- Differentiate JDK, JRE and JVM.
- Understand compilation and bytecode execution.

Key Concepts

The Java platform includes tools and runtime components that compile and execute Java programs. Learn the relationship among source code, bytecode, JVM, JRE and JDK.

Practical Activities

- Trace the path from .java source to program execution.
- Identify common Java development tools.

Quick Revision: Review the Java syntax, programming concept or development workflow introduced in this chapter, then repeat the practical exercise and save the source code with clear filenames.

3. Setting Up the Java Development Environment

Learning Outcomes

- Prepare a Java development environment.
- Compile and run a basic Java program.

Key Concepts

A development environment may use the JDK, command line tools or an IDE. Learn installation checks, project folders and basic environment configuration.

Practical Activities

- Create a Java project folder.
- Compile and execute a Hello World program.

Quick Revision: Review the Java syntax, programming concept or development workflow introduced in this chapter, then repeat the practical exercise and save the source code with clear filenames.

4. Java Syntax, Variables & Data Types

Learning Outcomes

- Declare variables correctly.
- Choose suitable Java data types.

Key Concepts

Java programs use classes, methods, statements, variables and data types. Learn primitive types, reference types, identifiers and literals.

Practical Activities

- Create programs using int, double, char, boolean and String.
- Practice variable declaration and output statements.

Quick Revision: Review the Java syntax, programming concept or development workflow introduced in this chapter, then repeat the practical exercise and save the source code with clear filenames.

Practice Note: Compile and test programs in a controlled development environment. Validate inputs and review code before using it in a real application.

5. Operators & Expressions

Learning Outcomes

- Use common Java operators.
- Build valid expressions and calculations.

Key Concepts

Java provides arithmetic, relational, logical, assignment, unary and conditional operators for calculations and decision-making.

Practical Activities

- Create a calculator-style program.
- Practice relational and logical expressions.

Quick Revision: Review the Java syntax, programming concept or development workflow introduced in this chapter, then repeat the practical exercise and save the source code with clear filenames.

Practice Note: Compile and test programs in a controlled development environment. Validate inputs and review code before using it in a real application.

6. Conditional Statements

Learning Outcomes

- Use decision-making structures.
- Design programs with multiple conditions.

Key Concepts

Conditional statements allow programs to make decisions based on conditions. Learn if, if-else, nested conditions and switch.

Practical Activities

- Create a marks/grade program.
- Build a menu-driven example using switch.

Quick Revision: Review the Java syntax, programming concept or development workflow introduced in this chapter, then repeat the practical exercise and save the source code with clear filenames.

Practice Note: Compile and test programs in a controlled development environment. Validate inputs and review code before using it in a real application.

7. Loops & Iteration

Learning Outcomes

- Use appropriate loop structures.
- Control repeated program execution.

Key Concepts

Loops repeat statements efficiently. Learn for, while and do-while loops along with break and continue.

Practical Activities

- Print number and multiplication tables.
- Create a program to calculate totals or repeated input.

Quick Revision: Review the Java syntax, programming concept or development workflow introduced in this chapter, then repeat the practical exercise and save the source code with clear filenames.

Practice Note: Compile and test programs in a controlled development environment. Validate inputs and review code before using it in a real application.

8. Arrays & String Handling

Learning Outcomes

- Work with arrays and strings.
- Process collections of values using loops.

Key Concepts

Arrays store collections of values, while String objects represent text. Learn one-dimensional arrays, common string operations and basic traversal.

Practical Activities

- Calculate total and average from an array.
- Perform common String operations and comparisons.

Quick Revision: Review the Java syntax, programming concept or development workflow introduced in this chapter, then repeat the practical exercise and save the source code with clear filenames.

Practice Note: Compile and test programs in a controlled development environment. Validate inputs and review code before using it in a real application.

9. Methods & Modular Programming

Learning Outcomes

- Create reusable methods.
- Use parameters and return values effectively.

Key Concepts

Methods divide programs into manageable units and support reuse. Learn parameters, return values, method overloading and scope.

Practical Activities

- Write methods for arithmetic operations.
- Create overloaded methods for related tasks.

Quick Revision: Review the Java syntax, programming concept or development workflow introduced in this chapter, then repeat the practical exercise and save the source code with clear filenames.

Practice Note: Compile and test programs in a controlled development environment. Validate inputs and review code before using it in a real application.

10. Object-Oriented Programming Concepts

Learning Outcomes

- Create classes and objects.
- Understand core object-oriented terminology.

Key Concepts

Java is strongly associated with object-oriented programming. Learn classes, objects, fields, methods, constructors and the principles of encapsulation.

Practical Activities

- Build a Student or Employee class.
- Create objects and call their methods.

Quick Revision: Review the Java syntax, programming concept or development workflow introduced in this chapter, then repeat the practical exercise and save the source code with clear filenames.

Practice Note: Compile and test programs in a controlled development environment. Validate inputs and review code before using it in a real application.

11. Inheritance & Polymorphism

Learning Outcomes

- Understand inheritance relationships.
- Use method overriding and polymorphism.

Key Concepts

Inheritance allows classes to derive behavior from other classes. Polymorphism allows common interfaces to represent different implementations.

Practical Activities

- Create a simple parent-child class hierarchy.
- Demonstrate method overriding with a practical example.

Quick Revision: Review the Java syntax, programming concept or development workflow introduced in this chapter, then repeat the practical exercise and save the source code with clear filenames.

Practice Note: Compile and test programs in a controlled development environment. Validate inputs and review code before using it in a real application.

12. Abstraction, Interfaces & Encapsulation

Learning Outcomes

- Understand abstraction and interfaces.
- Apply encapsulation using access modifiers and methods.

Key Concepts

Abstraction focuses on essential behavior, interfaces define contracts, and encapsulation protects object state through controlled access.

Practical Activities

- Create an abstract class example.
- Create and implement a simple interface.

Quick Revision: Review the Java syntax, programming concept or development workflow introduced in this chapter, then repeat the practical exercise and save the source code with clear filenames.

Practice Note: Compile and test programs in a controlled development environment. Validate inputs and review code before using it in a real application.

13. Packages, Access Modifiers & Exception Handling

Learning Outcomes

- Organize Java classes into packages.
- Handle common exceptions safely.

Key Concepts

Packages organize classes, while access modifiers control visibility. Exception handling allows programs to respond to runtime problems in a structured way.

Practical Activities

- Create a package-based mini project.
- Use try-catch-finally for a controlled error scenario.

Quick Revision: Review the Java syntax, programming concept or development workflow introduced in this chapter, then repeat the practical exercise and save the source code with clear filenames.

Practice Note: Compile and test programs in a controlled development environment. Validate inputs and review code before using it in a real application.

14. Collections Framework

Learning Outcomes

- Understand common collection interfaces.
- Choose a suitable collection for a task.

Key Concepts

Collections provide flexible structures for storing and processing groups of objects. Learn List, Set, Map and common collection operations.

Practical Activities

- Store records using ArrayList.
- Practice HashSet and HashMap with simple examples.

Quick Revision: Review the Java syntax, programming concept or development workflow introduced in this chapter, then repeat the practical exercise and save the source code with clear filenames.

Practice Note: Compile and test programs in a controlled development environment. Validate inputs and review code before using it in a real application.

15. File Handling & Input/Output

Learning Outcomes

- Read and write simple files.
- Understand basic Java I/O concepts.

Key Concepts

Java I/O APIs support reading and writing data. Learn basic file operations, streams/readers/writers and safe resource handling.

Practical Activities

- Create a text file from a Java program.
- Read records from a file and display them.

Quick Revision: Review the Java syntax, programming concept or development workflow introduced in this chapter, then repeat the practical exercise and save the source code with clear filenames.

Practice Note: Compile and test programs in a controlled development environment. Validate inputs and review code before using it in a real application.

16. JDBC & Database Connectivity

Learning Outcomes

- Understand JDBC architecture.
- Perform basic database operations from Java.

Key Concepts

JDBC allows Java applications to communicate with relational databases. Learn the concepts of drivers, connections, statements, prepared statements and result sets.

Practical Activities

- Connect a training application to a database.
- Practice a parameterized query using PreparedStatement.

Quick Revision: Review the Java syntax, programming concept or development workflow introduced in this chapter, then repeat the practical exercise and save the source code with clear filenames.

Practice Note: Compile and test programs in a controlled development environment. Validate inputs and review code before using it in a real application.

17. Java GUI, Web & Application Development Overview

Learning Outcomes

- Understand major Java application areas.
- Recognize how Java applications interact with users and services.

Key Concepts

Java can be used in multiple application environments. This chapter introduces GUI concepts and gives an overview of Java web and enterprise development without requiring a specific framework.

Practical Activities

- Design a simple application screen on paper or in an available IDE.
- Map a Java application into UI, logic and data layers.

Quick Revision: Review the Java syntax, programming concept or development workflow introduced in this chapter, then repeat the practical exercise and save the source code with clear filenames.

18. Debugging, Testing, Security & Best Practices

Learning Outcomes

- Identify and fix common programming errors.
- Apply basic coding, testing and security practices.

Key Concepts

Good Java development includes debugging, validation, testing, readable code, secure input handling and maintainable project structure.

Practical Activities

- Debug a deliberately faulty Java program.
- Create a small set of test cases and validate expected outputs.

Quick Revision: Review the Java syntax, programming concept or development workflow introduced in this chapter, then repeat the practical exercise and save the source code with clear filenames.

19. Java Practical Projects & Portfolio

Learning Outcomes

- Plan and build a Java application.
- Document source code and project functionality.

Key Concepts

Integrated projects help learners combine programming, OOP, collections, file/database handling and documentation into usable applications.

Practical Activities

- Build a mini project such as Student Management, Inventory, Billing or Library Management.
- Prepare source code, test data, screenshots and a project report.

Quick Revision: Review the Java syntax, programming concept or development workflow introduced in this chapter, then repeat the practical exercise and save the source code with clear filenames.

20. Viva, Glossary & Final Assessment

Learning Outcomes

- Demonstrate integrated Java programming skills.
- Explain program design and coding decisions in a viva.

Key Concepts

The final assessment evaluates Java fundamentals, programming logic, object-oriented programming, collections, I/O, database connectivity and project implementation.

Practical Activities

- Complete the final Java project.
- Present the application and explain key classes, methods and data flow.

Quick Revision: Review the Java syntax, programming concept or development workflow introduced in this chapter, then repeat the practical exercise and save the source code with clear filenames.

Final Practical Assessment

Students should complete an integrated Java project demonstrating programming logic, OOP, collections, file/database handling, testing and documentation.

Assessment Area	Marks
Java fundamentals, syntax & data types	10
Conditions, loops, arrays & strings	15
Methods & modular programming	10
OOP: classes, inheritance, polymorphism & abstraction	20
Packages, exceptions & collections	10
File handling & JDBC	10
Debugging, testing & coding practices	5
Project documentation & presentation	5
Viva & final practical project	15
Total	100

Suggested Final Project

- Choose a practical application such as Student Management, Library Management, Inventory, Billing or Employee Management.
- Design appropriate classes, fields and methods.
- Use constructors, encapsulation and suitable object-oriented relationships.
- Implement menus or a clear user interaction flow.
- Use collections to store and process records.
- Add file storage or JDBC database connectivity where appropriate.
- Validate input and handle common exceptions.
- Test the application with positive and negative test cases.
- Prepare source-code documentation, screenshots and a short project report.
- Present the application and explain its main classes, methods and data flow.

Viva / Interview Questions

- What is Java?
- What are JDK, JRE and JVM?
- What is bytecode?
- What is the difference between primitive and reference data types?
- What is a class and what is an object?
- What is a constructor?
- What is encapsulation?
- What is inheritance?
- What is polymorphism?
- What is abstraction?
- What is an interface?
- What is method overloading?
- What is method overriding?
- What is exception handling?

- What is the difference between ArrayList and HashSet?
- What is a HashMap?
- What is JDBC?
- What is a PreparedStatement?
- Why is testing important?

Glossary

Java: General-purpose programming language and platform widely used for application development.

JDK: Java Development Kit containing tools needed to develop and run Java applications.

JRE: Java Runtime Environment used to run Java applications.

JVM: Java Virtual Machine that executes Java bytecode.

Class: Blueprint defining data and behavior for objects.

Object: Runtime instance of a class.

Method: Named block of code that performs an operation.

Constructor: Special class member used to initialize objects.

Inheritance: Mechanism through which a class derives characteristics from another class.

Polymorphism: Ability to use a common interface or reference for different implementations.

Encapsulation: Bundling data and methods while controlling access to internal state.

Interface: Contract defining methods that implementing classes provide.

Exception: Event or condition that disrupts normal program execution.

Collection: Java framework structure used to store and manage groups of objects.

JDBC: Java API used to connect applications with relational databases.

IDE: Integrated Development Environment used to write, run and debug software.



IAS YOUTH COMPUTER CENTRE

Diploma in Java Technology

88/1, Queen Apartment, 2nd Floor,
Dumdum Road, Near Life Line Medical Store,
Kolkata – 700030, West Bengal

Learn • Code • Test • Build