PROFESSIONAL CORE JAVA PROGRAM | CAREER-FOCUSED CURRICULUM

Learning Roadmap

A visual path through the complete 8-module curriculum.

2 Months	8 Modules	Core Java
Program Duration	Complete Curriculum	Applied Learning

01 **Computer Fundamentals & Career Orientation** Foundation	02 **Core Java Basics** Programming Foundation
03 **Object-Oriented Programming** Core Java	04 **Exception Handling, Generics & Collections** Core Java
05 **File Handling, I/O & Core APIs** Core Java	06 **Modern Java (8 to 21)** Core Java
07 **Multithreading, Concurrency & JVM** Core Java	08 **Build Tools, Testing & Capstone Project** Applied Practice

Program at a Glance

Key information about the Core Java program.

Program	Professional Core Java Program
Duration	2 Months (approx. 9 weeks), 8 modules
Learning Approach	Instructor-led sessions, hands-on coding labs, practice programs and a capstone project
Eligibility	Graduates and final-year students from any stream; basic computer knowledge. No prior programming experience required
Technology Stack	Java 21 (LTS), IntelliJ IDEA, Maven, JUnit 5, Git and GitHub
Program Outcome	Strong Core Java developer who can build well-structured, concurrent and tested Java applications confidently

Program Structure & Timeline

Module-wise duration, phase and primary tools, progressing from Java fundamentals to modern Java, concurrency and a capstone project.

No.	Module	Phase	Duration	Key Tools
01	Computer Fundamentals & Career Orientation	Foundation	1 Week	JDK 21, IntelliJ IDEA, VS Code, Git
02	Core Java Basics	Core Java	1 Week	Java 21, IntelliJ IDEA
03	Object-Oriented Programming	Core Java	2 Weeks	Java 21, IntelliJ IDEA
04	Exception Handling, Generics & Collections	Core Java	1 Week	Java Collections Framework
05	File Handling, I/O & Core APIs	Core Java	1 Week	java.io, java.nio.file
06	Modern Java (8 to 21)	Core Java	1 Week	Java 21, Streams
07	Multithreading, Concurrency & JVM	Core Java	1 Week	Java 21, ExecutorService
08	Build Tools, Testing & Capstone Project	Practice	1 Week	Maven, JUnit 5, IntelliJ IDEA, GitHub
	Total Program Duration		**9 Weeks**	**Approx. 2 months**

MODULE 01

Computer Fundamentals & Career Orientation

01

Foundation-level orientation to programming, how Java works, the Java ecosystem and career paths, and environment setup.

DURATION	LEVEL	KEY TOOLS	MODULE OUTCOME
1 Week	Beginner	JDK 21, IntelliJ IDEA, VS Code, Git	Ready development environment and a clear learning roadmap

Topics Covered

- Introduction to Programming and Problem Solving
- Algorithms and Flowcharts
- How Java Works (Source Code, Bytecode, JVM)
- Compiled vs Interpreted Languages
- Java Editions and LTS Releases
- Java Ecosystem and Career Paths
- Number Systems and Binary Basics
- JDK vs JRE vs JVM
- Command Line Basics
- Version Control Basics

Software to Install

- JDK 21 (LTS)
- IntelliJ IDEA or Eclipse
- Visual Studio Code
- Git and GitHub

Assignment

- Install and verify the JDK and IDE
- Create and configure a GitHub account
- Write, compile and run a Hello World program
- Create a repository for practice solutions

WHAT THIS MODULE SUPPORTS

UNDERSTAND
Know how Java source code becomes a running program.

SET UP
Prepare a professional development environment.

PLAN
Follow a clear roadmap toward a Java career.

MODULE 02

Core Java Basics

02

Java syntax and programming logic: variables, operators, control flow, methods, arrays and strings.

DURATION	LEVEL	KEY TOOLS	MODULE OUTCOME
1 Week	Beginner	Java 21, IntelliJ IDEA	Write correct Java programs using control flow, arrays and strings

Part 1: Java Syntax

- Structure of a Java Program
- Variables and Data Types
- Operators
- Type Casting
- Input and Output (Scanner)
- Comments and Naming Conventions

Part 2: Control Flow

- if / else
- switch
- for Loop
- while Loop
- do-while Loop
- Enhanced for Loop
- break and continue
- Nested Loops
- Pattern Printing

Part 3: Methods, Arrays & Strings

- Methods and Overloading
- Variable Arguments (varargs)
- Arrays (1D)
- Arrays (2D)
- Strings and String Methods
- StringBuilder and StringBuffer
- Command-line Arguments
- Wrapper Classes and Autoboxing

Practice

- Number Programs (Prime, Fibonacci, Palindrome)
- Pattern Programs
- Array Programs
- String Programs

WHAT THIS MODULE SUPPORTS

WRITE
Write correct Java using variables, operators and input/output.

CONTROL
Direct program flow with conditions and loops.

ORGANIZE
Structure code with methods, arrays and strings.

MODULE 03

Object-Oriented Programming

03

Object-oriented design in Java: classes, the four pillars of OOP, interfaces and advanced class features.

DURATION	LEVEL	KEY TOOLS	MODULE OUTCOME
2 Weeks	Beginner to Intermediate	Java 21, IntelliJ IDEA	Design object-oriented programs with reusable classes

Part 1: Classes & Objects

- Classes and Objects
- Constructors
- this Keyword
- static Members
- Initialization Blocks
- Object Class
- Packages
- Access Modifiers

Part 2: Pillars of OOP

- Encapsulation
- Inheritance
- super Keyword
- Method Overriding
- Polymorphism
- Abstraction
- Abstract Classes
- Interfaces
- Default and Static Interface Methods

Part 3: Advanced OOP

- Inner and Nested Classes
- Anonymous Classes
- Enums
- Records
- final Keyword
- Immutable Classes
- equals and hashCode
- toString
- Composition vs Inheritance
- SOLID Principles Basics

Practice Programs

- Library Management System
- Banking System
- Inventory Management

WHAT THIS MODULE SUPPORTS

MODEL

Represent real-world entities as classes and objects.

REUSE

Apply inheritance, polymorphism and interfaces for reusable code.

PROTECT

Use encapsulation and immutability to keep data safe.

MODULE 04

Exception Handling, Generics & Collections

04

Robust error handling, type-safe generics and the Java Collections Framework.

DURATION	LEVEL	KEY TOOLS	MODULE OUTCOME
1 Week	Intermediate	Java Collections Framework	Handle errors, use generics and work with collections confidently

Exception Handling

- try, catch, finally
- Multiple catch Blocks
- throw and throws
- Checked vs Unchecked Exceptions
- Custom Exceptions
- try-with-resources
- Reading Stack Traces

Generics

- Generic Classes
- Generic Methods
- Bounded Types
- Wildcards

Collections Framework

- ArrayList
- LinkedList
- HashSet
- LinkedHashSet
- TreeSet
- HashMap
- LinkedHashMap
- TreeMap
- Queue and Deque
- PriorityQueue
- Iterator and ListIterator
- Comparable and Comparator
- Collections Utility Class
- Immutable Collections (List.of, Map.of)
- How HashMap Works Internally

Practice

- Student Record Manager using Collections
- Contact Book with Search and Sort

WHAT THIS MODULE SUPPORTS

HANDLE
Handle errors gracefully with well-designed exceptions.

STORE
Choose the right collection for each problem.

ORDER
Sort and compare objects with Comparable and Comparator.

MODULE 05

File Handling, I/O & Core APIs

05

Working with files and streams, text processing, serialization and everyday utility APIs.

DURATION	LEVEL	KEY TOOLS	MODULE OUTCOME
1 Week	Intermediate	java.io, java.nio.file	Read, write and process data using files and core Java APIs

File Handling & I/O

- File and Path
- Reading Files
- Writing Files
- Byte and Character Streams
- Buffered Streams
- NIO Files API
- Serialization

Strings & Text Processing

- String Pool and Immutability
- String Formatting
- Regular Expressions
- Scanner and Formatter
- StringBuilder Internals

Core Utilities

- Properties Files
- Logging Basics (java.util.logging)
- Shallow vs Deep Copy

Practice

- CSV Reader and Writer
- Log File Analyzer
- Text Search Utility

WHAT THIS MODULE SUPPORTS

PERSIST

Save and load application data using files and serialization.

PARSE

Process text with strings and regular expressions.

TRACE

Add basic logging to follow program behaviour.

MODULE 06

Modern Java (8 to 21)

06

Functional programming and the modern language features introduced from Java 8 to Java 21.

DURATION	LEVEL	KEY TOOLS	MODULE OUTCOME
1 Week	Intermediate to Advanced	Java 21, Streams	Write concise, modern Java using lambdas, streams and new language features

Functional Programming

- Lambda Expressions
- Functional Interfaces
- Method References
- Stream API
- Collectors and Grouping
- flatMap and Reduction
- Optional

Language Features

- var Keyword
- Text Blocks
- Switch Expressions
- Records
- Sealed Classes
- Pattern Matching

Date and Time

- LocalDate, LocalTime and LocalDateTime
- Duration, Period and Formatting

Practice

- Employee Data Analysis with Streams
- Modeling Data with Records and Sealed Types

WHAT THIS MODULE SUPPORTS

SIMPLIFY
Write concise code with lambdas and streams.

TRANSFORM
Process data with stream pipelines and collectors.

MODERNIZE
Use records, sealed classes and pattern matching.

MODULE 07

Multithreading, Concurrency & JVM

07

Concurrent programming, the java.util.concurrent toolkit and JVM fundamentals.

DURATION	LEVEL	KEY TOOLS	MODULE OUTCOME
1 Week	Intermediate to Advanced	Java 21, ExecutorService	Write safe concurrent Java and understand how the JVM works

Multithreading Basics

- Thread and Runnable
- Thread Lifecycle
- Synchronization
- wait and notify
- volatile and Visibility
- Race Conditions
- Deadlock

Concurrency Utilities

- ExecutorService
- Callable and Future
- CompletableFuture
- Locks and Atomic Variables
- Concurrent Collections
- CountDownLatch and Semaphore
- Virtual Threads

JVM Basics

- JVM Architecture
- Stack and Heap Memory
- Garbage Collection
- Class Loading

Practice

- Producer-Consumer Problem
- Thread-safe Bank Transfer

WHAT THIS MODULE SUPPORTS

PARALLELIZE
Run tasks safely with threads and executors.

SYNCHRONIZE
Prevent race conditions and deadlocks.

UNDERSTAND
Know how the JVM manages memory and execution.

MODULE 08

Build Tools, Testing & Capstone Project

08

Professional development practices: reflection, build tools, unit testing and a capstone project built end to end.

DURATION	LEVEL	KEY TOOLS	MODULE OUTCOME
1 Week	Intermediate to Advanced	Maven, JUnit 5, GitHub	Deliver a tested, well-structured Core Java application

Reflection & Annotations

- Reflection API
- Custom Annotations
- Retention and Target

Build & Testing

- Maven Basics
- Project Structure
- Managing Dependencies
- JUnit 5
- Testing Exceptions

Professional Practices

- Clean Code Practices
- Debugging with IntelliJ
- Git Branching and Pull Requests
- Code Review Basics

Capstone Project (choose one)

- Library Management System with File Persistence
- Employee Management App using Streams
- Multi-threaded Task Scheduler

WHAT THIS MODULE SUPPORTS

BUILD

Structure and build projects with Maven.

TEST

Verify code with JUnit 5 unit tests.

DELIVER

Ship a complete, documented project on GitHub.