



TECH NEXTGEN ACADEMY

JAVA PROGRAMMING

Think. Code. Design. Build. Grow.

Foundation to Professional Level | Practical Classroom Training | Project-Based Learning

Duration: 5–6 Months



BUILD PROFESSIONAL PROGRAMMING SKILLS WITH JAVA

Java is one of the world's most widely used programming languages for building **enterprise applications, backend systems, web applications, Android applications, APIs, and large-scale software systems.**

Its powerful combination of **Object-Oriented Programming, platform independence, strong memory management, exception handling, Collections Framework, multithreading, database connectivity, and modern language features** makes Java an essential skill for aspiring software developers.

At **Tech NextGen Academy**, Java Programming is designed to take you from programming fundamentals to professional Java development through **hands-on coding, logical problem-solving, OOP, database integration, application development, and real-world projects.**

The course focuses not only on learning Java syntax, but on developing the ability to **design programs, write clean code, work with databases, build applications, and understand professional software development practices.**

Our Goal: From Programming Beginner → Confident Java Programmer → Job-Ready Developer



WHO IS THIS COURSE FOR? & PREREQUISITES

TARGET AUDIENCE

- School & College Students
- BCA / B.Sc. / B.Tech Students
- Computer Science & Engineering Students
- Commerce & Non-IT Students
- Job Seekers & Freshers
- Aspiring Software & Java Developers
- Beginners in Programming
- Students Preparing for Coding Interviews / Academic Java
- Students Interested in Enterprise Application Development
- Professionals Looking to Upgrade Skills

PREREQUISITES

- Basic computer knowledge is recommended.
- No previous Java programming experience is required.
- Basic programming knowledge is helpful but **not mandatory**.
- A willingness to practice, solve problems, and build projects is essential.

Course Duration: 5 to 6 Months (Comprehensive Practical Training)



COURSE CURRICULUM & MODULES

01 PROGRAMMING FUNDAMENTALS & LOGICAL THINKING

Learn to Think Before You Code: What is programming? | Programming languages | Why Java? | Features & Applications | Java editions overview | Program development process | Source code & bytecode | Compiler & interpreter concepts | Algorithms, Flowcharts & Pseudocode | Problem decomposition | Input → Process → Output | Logical thinking & Problem-solving strategies

PRACTICAL: Convert real-world problems into: *Problem → Algorithm → Flowchart → Java Program → Output*

02 JAVA PLATFORM & DEVELOPMENT ENVIRONMENT

Understand How Java Works: Java history & evolution | Java features | JDK, JRE & JVM structure | Java Development Kit components | Bytecode & Platform independence | Compilation & Execution process | .java & .class files | Classpath basics & Environment variables | IDE introduction

Development Tools: Installing/configuring JDK | IDE setup | Project creation | Source files | Compilation, Execution & Debugging basics

PRACTICAL: Master the complete Java workflow: *Write* → *Compile* → *Bytecode* → *JVM* → *Execute* → *Test* → *Debug*

03 JAVA SYNTAX, VARIABLES, DATA TYPES & OPERATORS

Data Types: Primitive (byte, short, int, long, float, double, char, boolean) | Reference data types | Literals & Constants

Variables: Declaration, Initialization & Assignment | Local, Instance & Static variables | Variable scope & Naming conventions

Operators & Type Conversion: Arithmetic, Relational, Logical, Assignment, Increment/Decrement, Conditional, Bitwise, Shift, instanceof | Precedence & Associativity | Widening & Narrowing conversion | Type casting

PRACTICAL: Build Calculator, Marks Calculator, Salary Calculator, Area Calculator, and Simple Billing Application.

04 INPUT, OUTPUT & STRING HANDLING

Interactive Programs: System.out.print(), println(), printf() | Scanner class | User input & Formatted output | Escape sequences & Input validation

String Handling: String fundamentals & objects | String immutability | String comparison & concatenation | Methods (searching, substrings, length, character access)

StringBuilder & StringBuffer: Mutable vs Immutable strings | Performance considerations

PRACTICAL: Create Student Information System, Employee Information Program, Text-Processing Application, and Simple Billing Application.

05 DECISION MAKING & CONDITIONAL STATEMENTS

Make Programs Think & Decide: if, if-else, Nested if, else-if | switch & modern switch expressions | case, default, break | Conditional operators & logical expressions

PROBLEM-SOLVING & CHALLENGE: Build Even/Odd checker, Positive/Negative checker, Largest/Smallest of numbers, Grade calculator, Eligibility checker, Electricity bill calculation, Salary calculation, and Menu-driven programs independently.

06 LOOPS & ITERATION

Automate Repetitive Operations: for, Enhanced for, while, do-while | Nested loops | break & continue | Infinite loops | Loop tracing | Counter-controlled & Condition-controlled loops

PROGRAMMING PRACTICE & CHALLENGE: Number & Character patterns, Tables, Factorial, Fibonacci series, Prime numbers, Armstrong numbers, Palindrome numbers, Reverse numbers, Sum of digits, and Number counting.

07 METHODS & MODULAR PROGRAMMING

Write Reusable & Organized Code: Method declaration, definition & invocation | Parameters, Arguments & Return values | void methods | Method overloading | Static vs Instance methods | Argument passing & Scope | Modular programming

PRACTICAL & METHOD DESIGN: Learn to break large problems into smaller, reusable methods and build a menu-driven application using multiple methods and method overloading.

08 ARRAYS & MULTI-DIMENSIONAL DATA

One-Dimensional Arrays: Array declaration, creation, initialization, indexing, traversal, searching, sorting, updating elements, and operations (Sum, Average, Max, Min, Frequency, Reversing, Merging)

Multi-Dimensional Arrays: Two-dimensional arrays | Matrix representation | Matrix Addition, Subtraction, Multiplication, and Transpose

PRACTICAL PROJECT: Build a **Student Marks Analysis System** using arrays and methods.

09 OBJECT-ORIENTED PROGRAMMING FUNDAMENTALS

Enter Professional Java Development: Procedural vs Object-Oriented Programming | What is an object & class? | Class members (Fields & Methods) | Object creation & references | new keyword | Access modifiers (public, private, protected, Package-level access) | Encapsulation & Data hiding | Abstraction

REAL-WORLD MODELING & PRACTICAL: Convert *Real-World Entity* → *Class* → *Object* → *Properties* → *Behaviors*. Create classes for Student, Employee, Bank Account, Product, Customer, and Library Book.

10 CONSTRUCTORS, 'this' & OBJECT LIFECYCLE

Constructors & 'this' Keyword: Default & Parameterized constructors | Constructor overloading & chaining | Constructor initialization | this() usage | Resolving variable conflicts | Calling methods/constructors | Returning current object

Object Lifecycle: Object creation, initialization, usage, object references | Garbage collection concepts & eligibility

PRACTICAL: Design classes using multiple constructors and demonstrate object creation and lifecycle management.

11 ENCAPSULATION & ABSTRACTION

Protect Data & Control Access: Data hiding | Private fields, Getter & Setter methods | Controlled data access & validation inside methods | Class design principles | Abstract classes & Abstract methods | Interfaces | Implementation hiding | Designing contracts

PRACTICAL: Build a secure **Bank Account** → **Customer** → **Account Operations** system using encapsulation and abstraction.

12 INHERITANCE & CODE REUSABILITY

Build Relationships Between Classes: Superclass & Subclass | extends & super keywords | Single, Multilevel, & Hierarchical inheritance | Constructor behavior in inheritance | Method overriding | Access control & Reusability

REAL-WORLD EXAMPLES & PRACTICAL: Build class hierarchies for Employee → Manager, Vehicle → Car, Person → Student, and Account → Savings Account for real-world business applications.

13 POLYMORPHISM & ADVANCED OOP

Compile-Time & Run-Time Polymorphism: Method & Constructor overloading | Compile-time binding | Method overriding & Dynamic method dispatch | Upcasting & Downcasting concepts | Parent references & Child objects | @Override annotation

Abstract Design: Abstract classes & methods | Interfaces & Multiple interface implementation | Interface-based programming

PRACTICAL: Build a polymorphic application using different types of real-world objects.

14 PACKAGES, ACCESS CONTROL & INTERFACES

Organize Java Applications Professionally: Packages fundamentals | Creating & importing packages | Built-in & User-defined packages | Naming conventions & structure | Access control (public, private, protected, Default) across packages | Interfaces | Multiple interfaces | Default & Static interface methods | Functional interface introduction

PRACTICAL: Create a multi-package application using interfaces and controlled access.

15 EXCEPTION HANDLING & ERROR MANAGEMENT

Build Reliable Applications: Checked vs Unchecked exceptions | Exception hierarchy | try, catch, finally, throw, throws | Multiple catch blocks & Nested try blocks | Custom exceptions & Exception propagation | Best practices

Common Exceptions: ArithmeticException, NullPointerException, ArrayIndexOutOfBoundsException, NumberFormatException, InputMismatchException, IOException

PRACTICAL: Build applications that safely handle invalid input, division by zero, invalid account operations, file errors, and invalid menu choices.

16 WRAPPER CLASSES, ENUMS & GENERICS

Write Flexible & Type-Safe Code: Primitive vs Object types | Wrapper classes (Integer, Double, Character, Boolean) | Autoboxing & Unboxing | Enum fundamentals, constants & methods | Generic programming, classes & methods | Type parameters, bounded types, wildcards & Type safety

PRACTICAL: Create reusable and type-safe classes for handling different types of data.

17 COLLECTIONS FRAMEWORK

Master Java's Most Important Professional Skill:

- **List:** ArrayList, LinkedList (Operations, adding, removing, searching, updating, iteration)
- **Set:** HashSet, LinkedHashSet, TreeSet (Duplicate handling, sorted collections)
- **Map:** HashMap, LinkedHashMap, TreeMap (Key-value storage, searching by key, updating)
- **Queue & Iteration:** Queue, PriorityQueue, Deque | Iterator, ListIterator, Enhanced for

PRACTICAL: Build an **Employee / Product / Student Management System** using Java Collections.

18 COLLECTION ALGORITHMS, LAMBDA & STREAM API

Functional Programming Concepts: Functional interfaces | Lambda expressions | Method references | Predicate, Consumer, Function, Supplier

Stream API & Optional: Stream creation | Pipelines | `filter()`, `map()`, `sorted()`, `distinct()`, `limit()`, `count()`, `forEach()`, `reduce()` | Collecting results | Optional to avoid null checks

PRACTICAL: Analyze employee, student, or sales data using: *Collections* → *Lambda* → *Streams* → *Filtering* → *Sorting* → *Aggregation*

19 FILE HANDLING & INPUT/OUTPUT

Store & Retrieve Data: File handling fundamentals | File class | Byte vs Character streams | `FileInputStream`, `FileOutputStream`, `FileReader`, `FileWriter`, `BufferedReader`, `BufferedWriter` | Reading & writing files | Appending data | File paths & Directory handling | Resource management with `try-with-resources`

PRACTICAL: Build a persistent file-based **Student / Employee / Inventory Management System**.

20 MULTITHREADING & CONCURRENCY

Understand Concurrent Execution: Process vs Thread | Thread fundamentals | Thread class & `Runnable` interface | Thread lifecycle & priorities | `sleep()` & `join()` | Synchronization, Race conditions & Thread safety | Synchronized methods & blocks | Inter-thread communication & Executor framework overview

PRACTICAL: Create applications performing multiple concurrent tasks (Background processing, Timer operations, Concurrent data processing).

22 DEBUGGING, TESTING & CODE QUALITY

Error Types & Debugging: Syntax, Runtime, Logical, Input & Database errors | Reading compiler messages | Breakpoints, Step-by-step execution, Variable inspection, Call stack | Identifying logical mistakes | Module testing & Input validation

Code Quality & Testing: Meaningful naming | Indentation | Modular clean code | Reusability | Avoiding duplication | Test cases (Positive, Negative, Boundary, Functional testing)

PRACTICAL: Debug intentionally broken Java programs and prepare a basic test report.

13-Step Professional Development Lifecycle followed in classroom projects:

Step 01: Requirement Analysis



Step 02: Problem Design



Step 03: Class Design



Step 04: Application Structure (Packages)



Step 05: OOP Implementation



Step 06: Data Management (Collections)



Step 07: Error Management (Exceptions)

Step 08: File / Database Storage



Step 09: Concurrent Processing



Step 10: Scenario & Input Testing



Step 11: Debugging & Error Correction



Step 12: Code Optimization & Refactoring



Step 13: Architecture & Viva Presentation

CAPSTONE PROJECT — COMPLETE JAVA MANAGEMENT SYSTEM

Students develop a complete Java application using the concepts learned throughout the course.

PROJECT DOMAIN OPTIONS (CHOOSE ONE)

Student Management System | Employee Management System | Library Management System | Banking Management System | Inventory Management System | Hospital Management System | Hotel Management System | Billing & Sales Management System

PROJECT FEATURES & DELIVERABLES

- Object-Oriented Architecture & Multiple Classes
- Encapsulation, Abstraction, Inheritance & Polymorphism
- Interfaces, Packages & Collections Framework
- Generics, Lambda Expressions & Stream API
- Input Validation & Exception Handling
- File Handling, JDBC Integration & Data Persistence
- Complete Java Source Code
- Requirement & Project Documentation
- Search, Sorting & Report Generation Functionality
- Menu-driven / User Interactive Interface
- Class & Package Structure Architecture
- Database Design (where applicable)
- Test Cases & Debugging Record
- Persistent Data Storage
- Comprehensive Project Report
- Final Viva & Live Presentation

FINAL PROJECT FLOW: Requirement → Design → Class Structure → Coding → OOP → Collections → Exception Handling → File/Database → JDBC → Testing → Debugging → Optimization → Presentation

TECH NEXTGEN LEARNING METHOD & JOURNEY

LEARN → THINK → CODE → DESIGN → TEST → DEBUG → BUILD → PRESENT → GROW

Your Learning Progression:

Foundation (Logic) → Platform (JVM/JDK) → Core Java → Interaction → Conditional Logic → Automation (Loops) → Modularity (Methods) → Arrays → OOP Foundation → Object Lifecycle → Encapsulation & Abstraction → Inheritance → Polymorphism & Interfaces → Packages & Access Control → Exception Handling → Wrapper Classes, Enums & Generics → Collections Framework → Modern Java (Lambda & Streams) → Storage (File I/O) → Concurrency (Multithreading) → Quality (Debugging/Testing) → Real-World Project → Capstone + Assessment + Certification.

🎯 WHAT YOU WILL BE ABLE TO DO & CERTIFICATION

KEY SKILL OUTCOMES

- Understand Java fundamentals, JDK, JRE, JVM & platform independence.
- Write Java programs from scratch, work with variables, data types, operators.
- Implement conditional logic, loops, iteration & recursion.
- Design reusable methods, process strings & arrays effectively.
- Create classes, objects & apply Encapsulation, Abstraction, Inheritance, Polymorphism.
- Use constructors, handle runtime problems using exceptions & custom exceptions.
- Work with Wrapper classes, Enums, Generics, List, Set, Map, Queue & Iterators.
- Write Lambda expressions, use Functional Interfaces & Stream API.
- Read/write files, manage basic threads, synchronization & thread safety.
- Connect Java apps to databases using JDBC, perform CRUD operations & PreparedStatement.
- Debug Java programs, write test cases & build database-driven applications.
- Build a strong foundation for Spring Boot & backend development.

ASSESSMENT & CERTIFICATION

Progress evaluated via: Assignments → Logic Challenges → Coding Exercises → OOP Projects → Practical Tests → Capstone & Viva.

Score	Grade
90–100%	A+
80–89%	A
70–79%	B+
60–69%	B
Below 60%	Needs Improvement

 **TECH NEXTGEN ACADEMY**
Java Programming Course Completion Certificate

🌟 WHY TECH NEXTGEN ACADEMY & CAREER PROGRESSION

WHY CHOOSE US?

🧠 Logic-Building Focus | ☕ Hands-On Java | 🏗️ Strong OOP Foundation | 📦 Collections & Modern Java (Streams/Lambda) | 🛡️ Exception Handling & Generics | 📁 Multithreading & File Handling | 🔍 Debugging & Testing | 🏢 Real-World Projects & Capstone | 🎤 Viva & Presentation | 📝 Assessment Certification

CONTINUE YOUR JAVA DEVELOPMENT JOURNEY

Progress toward: Advanced Java → Advanced OOP & Design Principles → Data Structures & Algorithms → SQL & Database Development → JDBC & Advanced DB Programming → Java Web Development (Servlets/JSP) → REST API Development → Spring Framework & Spring Boot → Spring Data JPA / Hibernate → Backend & Full Stack Java Development → Microservices & Git/GitHub.

JOIN TECH NEXTGEN ACADEMY

Learn Today. Lead Tomorrow. | Admissions Open | Limited Seats

📍 **Location:** Jadavpur, Kolkata
☎️ **Phone:** +91 6292009436 / +91 6292009536
📱 **WhatsApp:** +91 6292009436

✉️ **Email:** technextgenacademy@gmail.com
🌐 **Website:** technextgenacademy.com
🕒 **Duration:** 5 – 6 Months

START CODING. MASTER JAVA. BUILD REAL SOFTWARE.