



TECH NEXTGEN ACADEMY

C++ 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 C++

C++ is a powerful, versatile programming language used for **software development, game development, system programming, competitive programming, application development and high-performance computing.**

It combines the power and performance of C with modern programming concepts such as **Object-Oriented Programming, classes, objects, inheritance, polymorphism, templates, STL and exception handling.**

At **Tech NextGen Academy**, C++ Programming is designed to take you from programming fundamentals to confident, structured and object-oriented programming through **hands-on coding, problem-solving challenges, debugging exercises and real-world projects.**

The course focuses not only on learning syntax, but on developing the ability to **design programs, organize code, solve problems and build maintainable applications.**

Our Goal: From Programming Beginner → Confident C++ Programmer



WHO IS THIS COURSE FOR?

- 🎓 School & College Students
- 🎓 BCA / B.Sc. / B.Tech Students
- 🎓 Computer Science Students
- 🎓 Engineering Students
- 🎓 Commerce & Non-IT Students
- 📁 Job Seekers & Freshers
- 🧑‍💻 Aspiring Software Developers
- 🧠 Beginners in Programming
- 🧩 Students Preparing for Coding Interviews
- 🏆 Competitive Programming Beginners
- 🚀 Students Planning to Learn DSA
- 📖 Anyone wanting strong programming & OOP skills

Prerequisites:

- Basic computer knowledge is recommended.
- No previous C++ programming experience is required.
- Basic knowledge of C programming is helpful but **not mandatory.**
- A willingness to practice, solve problems and write code is essential.



WHAT YOU WILL MASTER

01 — PROGRAMMING FUNDAMENTALS & LOGICAL THINKING

LEARN TO THINK BEFORE YOU CODE

- What is programming? & Languages
- Why C++? Features & Applications
- Program development process
- Source code & executable code
- Compiler & compilation concepts
- Syntax & semantics
- Algorithms & Flowcharts
- Pseudocode & Problem decomposition
- Input → Process → Output
- Logical thinking & Strategies

PRACTICAL: Convert real-world problems into:

Problem → Algorithm → Flowchart → C++ Program → Output



02 — C++ PROGRAM STRUCTURE & DEVELOPMENT ENVIRONMENT

WRITE YOUR FIRST C++ PROGRAM

- Introduction to C++ & Program structure
- `main()` function, Header files, Namespaces (`#include`, `using namespace`)
- Statements, Comments, Keywords, Identifiers, Constants
- Compilation, Execution, Compiler errors & Basic debugging
- Input & Output: `cout`, `cin`, `endl`, Stream-based I/O & Formatted output

PRACTICAL: Create interactive programs & master workflow:

Write → Compile → Execute → Test → Debug



03 — VARIABLES, DATA TYPES & OPERATORS

BUILD THE CORE OF YOUR PROGRAMS

Data Types & Variables:

- `int`, `float`, `double`, `char`, `bool`, `void`
- Signed/unsigned & Type modifiers
- Data type sizes & Type casting
- Declaration, Initialization, Scope, Constants

Operators:

- Arithmetic, Relational, Logical
- Assignment, Increment/Decrement
- Conditional, Bitwise, `sizeof`
- Operator precedence & Associativity

PRACTICAL PROJECTS: Calculator, Marks Calculator, Salary Calculator, Area Calculator, Simple Billing Application.

04 — INPUT, OUTPUT & STRING BASICS

MAKE YOUR PROGRAMS INTERACTIVE

- cin, cout, getline(), Character input, Formatted output, Escape sequences
- Multiple inputs, Input validation, Basic string handling
- **C++ String Class (std::string):** Declaration, Initialization, Input, Concatenation, Comparison, Length, Character access, Searching inside strings, Substrings

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

05 — DECISION MAKING & CONDITIONAL STATEMENTS

MAKE PROGRAMS THINK & DECIDE

- if, if-else, Nested if, else-if, switch, case, default, break
- Conditional operator, Multiple conditions, Logical expressions

PROBLEM-SOLVING: Even/Odd, Positive/Negative, Largest/Smallest of numbers, Grade calculator, Eligibility checker, Electricity bill, Income calculation, Menu-driven programs.

06 — LOOPS & ITERATION

AUTOMATE REPETITIVE OPERATIONS

- for, while, do-while, Nested loops, break, continue, Infinite loops, Loop tracing
- Counter-controlled loops vs Condition-controlled loops

PRACTICE: Number patterns, Character patterns, Tables, Factorial, Fibonacci series, Prime numbers, Armstrong numbers, Palindrome numbers, Reverse numbers, Sum of digits.

07 — FUNCTIONS & MODULAR PROGRAMMING

WRITE REUSABLE & ORGANIZED CODE

- Function declaration, Definition, Call, Parameters, Arguments, Return values
- Default arguments, Function prototypes, Function overloading
- Local vs Global variables, Scope, Pass by value vs Pass by reference, Inline functions

PRACTICAL: Build a menu-driven application using multiple functions and demonstrate function overloading.

08 — ARRAYS & MULTI-DIMENSIONAL DATA

STORE & PROCESS MULTIPLE VALUES

- **1D Arrays:** Declaration, Initialization, Indexing, Traversal, I/O, Searching, Sorting, Updating
- **Array Operations:** Sum, Average, Max, Min, Frequency, Reversing, Merging, Sorting
- **2D Arrays:** Matrix representation, Addition, Subtraction, Multiplication, Transpose

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

09 — POINTERS & REFERENCES

UNDERSTAND MEMORY & DATA ACCESS

- Memory addresses, Pointer fundamentals, Address operator &, Dereference operator *
- Pointer arithmetic, Pointers with variables/arrays/functions, Pointer to pointer, Null pointers, Safety
- **C++ References:** Reference variables, References with functions, Pass/Return by reference, Pointers vs References

10 — OBJECT-ORIENTED PROGRAMMING FUNDAMENTALS

ENTER THE WORLD OF OOP

- Procedural vs Object-Oriented Programming
- Classes & Objects, Data members, Member functions, Object creation, Accessing members
- Access modifiers: `public`, `private`, `protected`
- Encapsulation, Data hiding, Abstraction

PRACTICAL: Create classes for Student, Employee, Bank Account, Product, Customer, Library Book.

11 — CONSTRUCTORS, DESTRUCTORS & OBJECT LIFECYCLE

CONTROL HOW OBJECTS ARE CREATED & DESTROYED

- **Constructors:** Default, Parameterized, Overloading, Initialization, Copy constructor, Default arguments
- **Destructors:** Syntax, Object destruction, Resource management fundamentals
- **Object Lifecycle:** Creation → Initialization → Usage → Destruction

12 — ENCAPSULATION & ABSTRACTION

PROTECT DATA & CONTROL ACCESS

- Data hiding, Access modifiers, Getter & Setter methods, Class design principles
- Abstraction necessity, Abstract design, Interface-like behavior, Hiding implementation details

PRACTICAL: Build secure classes for: **Bank Account** → **Customer** → **Account Operations**.

13 — INHERITANCE & CODE REUSABILITY

BUILD RELATIONSHIPS BETWEEN CLASSES

- Base class & Derived class
- Public, Protected, and Private inheritance modes
- Single, Multilevel, Multiple, Hierarchical, and Hybrid inheritance
- Constructor behavior in inheritance & Function overriding

EXAMPLES: Employee → Manager | Vehicle → Car | Person → Student | Account → Savings Account

14 — POLYMORPHISM & ADVANCED OOP

ONE INTERFACE → MULTIPLE BEHAVIORS

- **Compile-Time Polymorphism:** Function overloading, Operator overloading (Unary & Binary), Compile-time binding
- **Run-Time Polymorphism:** Function overriding, Virtual functions, Base class pointers, Dynamic binding, Pure virtual functions, Abstract classes, Virtual destructors

15 — FRIEND FUNCTIONS & ADVANCED CLASS DESIGN

DESIGN MORE FLEXIBLE C++ PROGRAMS

- Friend functions & Friend classes, Static data members & Static member functions
- `this` pointer, Nested classes, Constant member functions, Mutable concepts
- Object relationships: Composition, Aggregation, Association (Inheritance vs Composition)

16 — EXCEPTION HANDLING & ERROR MANAGEMENT

BUILD MORE RELIABLE APPLICATIONS

- Runtime errors & Exception concepts: `try`, `catch`, `throw`
- Multiple catch blocks, Catch-all handler, Custom exceptions, Propagation & Strategies

PRACTICAL: Safely handle Invalid Input, Division by Zero, Invalid Account Operations, File Errors.

17 — TEMPLATES & GENERIC PROGRAMMING

WRITE REUSABLE CODE FOR DIFFERENT DATA TYPES

- **Function Templates:** Generic functions & Template parameters
- **Class Templates:** Generic classes & Template-based data structures
- Generic programming principles to eliminate code duplication

18 — STANDARD TEMPLATE LIBRARY (STL)

MASTER THE POWER OF MODERN C++

- **Containers:** `vector`, `list`, `deque`, `stack`, `queue`, `priority_queue`, `set`, `multiset`, `map`, `multimap`, `unordered_set`, `unordered_map`
- **Iterators:** `begin()`, `end()`, Traversal, Operations
- **Algorithms:** `sort()`, `reverse()`, `find()`, `count()`, `min_element()`, `max_element()`
- **Utilities:** `pair`, Lambda expressions introduction

19 — FILE HANDLING & DATA PERSISTENCE

STORE APPLICATION DATA PERMANENTLY

- File streams: ifstream, ofstream, fstream
- Opening, Closing, Reading, Writing, Appending data
- Text files vs Binary files, File modes & position concepts, Error handling

PRACTICAL: Build a File-Based Student / Employee / Inventory Management System.

20 — MODERN C++ CONCEPTS

MOVE TOWARD PROFESSIONAL C++ DEVELOPMENT

- auto, nullptr, Range-based for, enum class, constexpr basics
- Lambda expressions, Smart pointers (unique_ptr, shared_ptr)
- Move semantics & Rvalue references fundamentals, RAII principles

21 — PROBLEM-SOLVING & CODING CHALLENGES

TURN KNOWLEDGE INTO SKILLS

Progressive coding challenge levels: **Beginner** → **Intermediate** → **Advanced**

Understand → **Plan** → **Code** → **Test** → **Debug** → **Improve**

22 — REAL-WORLD C++ PROGRAMMING SIMULATION

BUILD A COMPLETE OBJECT-ORIENTED APPLICATION



CAPSTONE PROJECT

Complete Object-Oriented Management System Using C++

Students will develop a complete menu-driven application using the concepts learned throughout the course.

PROJECT 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

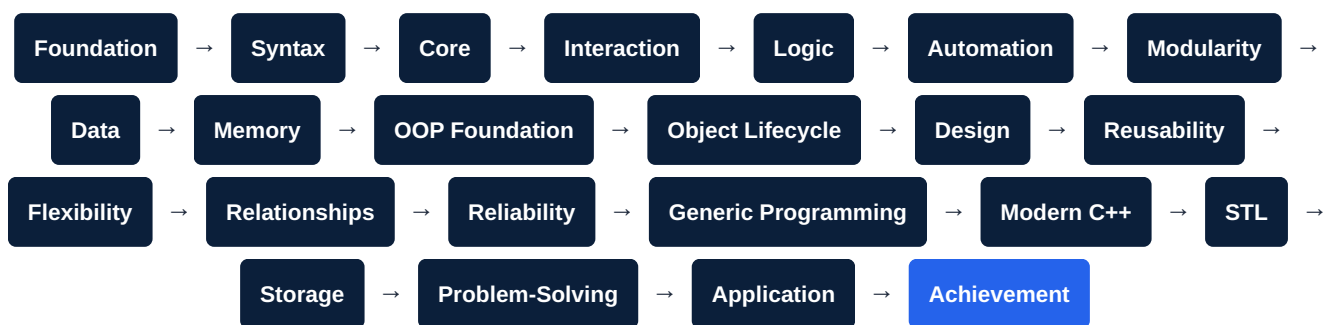
FINAL DELIVERABLES:

-  Complete C++ Source Code &  Requirement Documentation
-  Class Design &  Program Flow / Design Diagrams
-  Test Cases,  Debugging Record &  File Data Storage
-  Project Report &  Final Viva & Presentation

TECH NEXTGEN LEARNING METHOD

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

YOUR LEARNING JOURNEY



WHAT YOU WILL BE ABLE TO DO

- ✓ Understand programming fundamentals
- ✓ Develop logical & computational thinking
- ✓ Write C++ programs from scratch
- ✓ Work with variables, operators & data types
- ✓ Implement conditional logic & loops
- ✓ Design reusable functions & overloading
- ✓ Work with arrays, strings & pointers
- ✓ Create classes & apply encapsulation
- ✓ Master constructors & destructors
- ✓ Implement inheritance & polymorphism
- ✓ Apply operator overloading & friend functions
- ✓ Handle runtime errors using exceptions
- ✓ Create function & class templates
- ✓ Use STL containers, iterators & algorithms
- ✓ Use modern C++ features & RAI
- ✓ Store & retrieve data using file streams
- ✓ Implement searching & sorting algorithms
- ✓ Build menu-driven real-world applications
- ✓ Debug & write maintainable clean code
- ✓ Ace technical interviews & DSA preparation

ASSESSMENT & CERTIFICATION

Score Range	Grade
90–100%	A+
80–89%	A
70–79%	B+

Score Range	Grade
60–69%	B
Below 60%	Needs Improvement



TECH NEXTGEN ACADEMY

C++ Programming Course Completion Certificate



WHY TECH NEXTGEN ACADEMY?

Not Just C++ Classes. A Complete Programming & OOP Skill-Building Experience.

- **Logic-Building Focus:** Plan & solve before coding.
- **Hands-On C++:** Live coding sessions.
- **Problem Solving:** Progressive challenges.
- **OOP Mastery:** Full structural design.
- **Exception Handling:** Build reliable code.
- **STL & Modern C++:** Industry standard code.
- **File Handling:** Real data persistence.
- **Real Projects:** Capstone management systems.
- **Viva & Presentation:** Technical confidence.
- **Certified:** Practical evaluation.



CONTINUE YOUR PROGRAMMING JOURNEY

After completing C++ Programming, progress toward:

Data Structures & Algorithms • Advanced C++ • Competitive Programming • Object-Oriented Design • Java Programming • Python Programming • SQL & Database Development • Software Development • Backend Development • Full Stack Development • System Programming • Game Development Fundamentals



JOIN TECH NEXTGEN ACADEMY

Learn Today. Lead Tomorrow.



Jadavpur, Kolkata



+91 6292009436 / +91 6292009536



WhatsApp: +91 6292009436



technextgenacademy@gmail.com | technextgenacademy.com

Admissions Open | Limited Seats

START CODING. MASTER OOP. BUILD REAL SOFTWARE.