C++ Development Mastery (6 Months/ 24 Weeks)

Phase 1: Comprehensive C++ Training (4 Months / 16 Weeks)

Week 1: C++ Fundamentals Refresher

- Overview of C++ syntax, structure, and standard libraries
- Key concepts: Data types, variables, control flow (if, loops), functions, and error handling
- Hands-on exercises: Simple programs to reinforce basic concepts

Week 2: Object-Oriented Programming (OOP) in C++

- Introduction to classes and objects
- Encapsulation, inheritance, and polymorphism
- Hands-on: Create basic class-based applications

Week 3: Advanced OOP and Design Patterns

- Advanced object-oriented techniques: Abstract classes, interfaces, and virtual functions
- Introduction to design patterns: Singleton, Factory, Strategy
- Hands-on: Develop a small application using design patterns

Week 4: Memory Management in C++

- Pointers, dynamic memory allocation, and memory leak prevention
- Introduction to smart pointers (unique_ptr, shared_ptr)
- Hands-on: Build an application with dynamic memory management

Week 5: Data Structures and Algorithms in C++

- In-depth exploration of data structures: Linked lists, stacks, queues, and trees
- Algorithms: Sorting, searching, recursion
- Hands-on: Implement a mini-project utilizing these data structures

Week 6: Advanced C++ Programming Techniques

- Templates and generic programming
- Metaprogramming and type traits
- Hands-on: Build a custom data structure library using templates

Week 7: Multithreading and Concurrency

ction to threads, synchronization (mutexes, condition variables)
- Best practices for concurrent programming and thread safety
- Hands-on: Build a multithreaded calculator or concurrent task manager

Week 8: File I/O and Serialization

- File handling in C++: Reading from and writing to files
- Serialization techniques for storing and retrieving complex objects
- Hands-on: Develop a program to read and write structured data to files

Week 9: Real-Time Application Development

- Building responsive, low-latency applications
- Performance considerations in real-time systems (CPU optimization, latency control)
- Hands-on: Create a real-time data processing application

Week 10: Introduction to Frameworks - Qt for GUI Development

- Basics of Qt for GUI-based application development
- Event-driven programming and layout management
- Hands-on: Develop a simple GUI tool or app using Qt

Week 11: Using Boost Libraries for Complex Tasks

- Overview of the Boost libraries for threading, regex, filesystem, and more
- Applying Boost in real-world applications
- Hands-on: Integrate Boost functionalities into an existing project

Week 12: Networking in C++

- Basics of socket programming and network communication
- Introduction to REST APIs, HTTP, and TCP/IP
- Hands-on: Build a simple networked application (client-server)

Week 13: Debugging, Profiling, and Optimization

- Debugging techniques using GDB, Valgrind
- Performance profiling and code optimization
- Hands-on: Debug and optimize a real-world project

Week 14: Software Testing in C++

- Writing unit tests with Google Test framework
- Best practices for code coverage and test-driven development (TDD)
- Hands-on: Implement unit tests and integration tests for a project

oduction to High-Performance Systems

- Key concepts for building high-performance systems (low-level optimizations, caching)
- Real-time operating systems (RTOS) and embedded systems
- Hands-on: Develop a high-performance mini-project focusing on speed and optimization

Week 16: Capstone Project

- A real-world project where students integrate learned concepts and build a fully functional C++ application
- Project ideas: A high-performance app, a multithreaded server, a real-time processing tool
- Code reviews, feedback, and final adjustments

Phase 2: Job Training and Real-World Experience (2 Months / 8 Weeks)

Week 17: Job Preparation & Real-World Project Introduction

- Guidance on job readiness: Resume building, interview preparation, and industry best practices
- Introduction to real-world projects with clients (developing specific features based on the course)

Week 18-20: Hands-On Client Projects

- Students work on real-world projects, applying learned skills
- Focus on performance, scalability, and system-level optimizations
- Weekly checkpoints with mentors to refine skills

Week 21-22: Advanced Job Training

- Industry tools: Version control (Git), collaborative coding tools (GitHub, GitLab), continuous integration (CI/CD)
- Students integrate into client teams, contributing to active projects

Week 23-24: Final Project & Job Placement Assistance

- Completion of client projects or independent advanced mini-projects
- Final review and debugging
- Job placement preparation: Mock interviews, portfolio preparation, and networking