C and C++ Combined Programming Syllabus

Module 1: Introduction to C and C++

1. **Overview of C and C++**
   - History and differences between C and C++
   - Why learn C and C++?
   - Applications of C and C++ in software development, embedded systems, game development, etc.
2. **Setting Up Development Environment**
   - Installing C and C++ compilers (e.g., GCC, MinGW)
   - IDEs: Code::Blocks, Visual Studio, Dev-C++, Visual Studio Code, CLion
3. **First Program in C and C++**
   - Writing, compiling, and running a simple C program
   - Writing, compiling, and running a simple C++ program
   - Differences in syntax between C and C++

Module 2: Basic Syntax and Data Types

1. **Variables, Constants, and Data Types**
   - C: int, char, float, double
   - C++: New data types (bool, string, vector in C++)
   - Type modifiers (signed, unsigned, long, short)
2. **Operators**
   - C: Arithmetic, relational, logical, and bitwise operators
   - C++: In addition to C operators, C++ supports operator overloading
   - Assignment operators, increment/decrement operators
3. **Input/Output in C and C++**
   - C: printf(), scanf()
   - C++: cout, cin with the <<, >> operators
   - Formatting output in both C and C++

Module 3: Control Flow and Functions

1. **Control Flow Statements**
   - Conditional statements: if, else, switch
   - Loops: for, while, do-while
   - C++: for-each loop (range-based loop)
2. **Functions**
   - C: Defining and calling functions, function overloading not supported
   - C++: Function overloading, default arguments, inline functions

- Recursion

3. **Function Pointers in C**
   - C: Using function pointers for dynamic function calls
   - C++: Function pointers and lambda functions

Module 4: Arrays and Strings

1. **Arrays**
   - C: Defining and using arrays, multidimensional arrays
   - C++: Vectors (dynamic arrays), array bounds checking
   - Passing arrays to functions
2. **Strings**
   - C: C-style strings (character arrays) and functions (e.g., strcpy(), strlen(), strcmp())
   - C++: string class and its functions (e.g., concatenation, comparison, substr)

Module 5: Pointers and Memory Management

1. **Pointers**
   - C: Pointer basics, pointer arithmetic, arrays and pointers
   - C++: Pointers to objects, classes, functions, and dynamic memory management
   - Memory allocation: malloc(), calloc(), free() (C), new, delete (C++)
2. **Dynamic Memory Allocation**
   - C: Memory allocation with malloc(), calloc()
   - C++: new and delete operators, handling memory leaks with smart pointers (C++11 onwards)

Module 6: Object-Oriented Programming (OOP) in C++

1. **Introduction to OOP**
   - Differences between procedural and object-oriented programming
   - Key concepts: Classes, Objects, Encapsulation, Inheritance, Polymorphism, Abstraction
2. **Classes and Objects**
   - C: Using structures to simulate objects
   - C++: Classes, member functions, constructors, destructors
   - Access modifiers: private, public, protected
3. **Inheritance and Polymorphism**
   - C: Not applicable (only using structures)
   - C++: Single, multiple, and multilevel inheritance
   - Virtual functions, pure virtual functions, and abstract classes
4. **Operator Overloading and Function Overloading**
   - C++: Overloading operators like +, -, <<, >>
   - C++: Function overloading (same function name, different parameter types)

Module 7: Advanced Topics in C and C++

1. **Memory Management in C and C++**
   - C: Manual memory management with malloc(), free()
   - C++: Smart pointers (unique_ptr, shared_ptr, weak_ptr), RAII principle

2. **File Handling**
   - C: File operations using fopen(), fclose(), fprintf(), fscanf(), and binary file handling
   - C++: File streams (fstream, ifstream, ofstream), file pointer manipulation
3. **Exception Handling in C++**
   - C: Error codes, errno, and error handling mechanisms
   - C++: try, catch, throw for exception handling
4. **Templates**
   - C: Not applicable
   - C++: Function templates, class templates, template specialization, and generic programming

Module 8: Advanced Data Structures and Algorithms

1. **Data Structures**
   - C: Arrays, linked lists, stacks, queues, trees, and graphs (implemented manually)
   - C++: Using Standard Template Library (STL) containers: vector, stack, queue, list, map, set
2. **Algorithms**
   - Searching: Linear search, binary search (C and C++)
   - Sorting: Bubble sort, selection sort, quicksort, mergesort, and using STL sort in C++
   - Graph algorithms: BFS, DFS, Dijkstra's algorithm (C and C++)

Module 9: Multithreading and Concurrency in C++

1. **Introduction to Multithreading**
   - C: Not supported natively, external libraries required (e.g., POSIX threads)
   - C++: Multithreading using std::thread, synchronization with mutex, lock_guard, and condition_variable
2. **Lambda Expressions in C++**
   - Using anonymous functions (lambdas) for short-term functional operations
3. **Concurrency and Synchronization**
   - Handling concurrent execution, avoiding deadlocks, race conditions

Module 10: Projects and Applications

1. **Bank Management System** (using C++)
2. **Library Management System** (using C++)
3. **Tic-Tac-Toe Game** (using C)
4. **Simple File Compression Tool** (C and C++)
5. **Student Record Management System** (C and C++)
6. **Inventory System using OOP** (C++)
7. **Sorting and Searching Algorithms Visualizer** (C++)
8. **Multithreaded Web Scraper** (C++)

Tools and IDEs

- **Compilers**: GCC (C and C++), MinGW, Visual Studio, Clang
- **IDEs**: Code::Blocks, Dev-C++, Visual Studio, CLion, Eclipse, VS Code

- **Debugger**: GDB

More Courses: