C And C++ Syllabus

C Programming

Introduction

- ✓ History and Evolution of C
- ✓ Features of C
- ✓ Structure of a C Program
- ✓ Compilation and Execution Process (Preprocessor, Compiler, Linker)
- ✓ Basic Syntax (Statements, Identifiers, Keywords, Data Types)

Variables, Constants, and Data Types

- ✓ **Variables:** Declaration, Initialization, Scope, and Lifetime
- ✓ **Constants:** #define and const keyword
- ✓ **Data Types:** Basic (int, char, float, double), Derived (arrays, pointers), and User-defined (struct, union, enum)
- ✓ **Type Conversion:** Implicit and Explicit Type Conversion (Casting)

Operators

- ✓ **Arithmetic Operators**: Addition, subtraction, multiplication, division, modulus.
- ✓ **Relational Operators**: Equal to, greater than, less than.
- ✓ **Logical Operators**: AND, OR, NOT.
- ✓ **Bitwise Operators**: AND, OR, XOR, NOT, left and right shifts.
- ✓ **Assignment Operators**: Simple, compound assignment.
- ✓ **Miscellaneous Operators**: sizeof, comma operator, pointer operator.

Control Structures

- ✓ **Conditional Statements:** if, else, switch-case, and nested conditions.
- ✓ **Loops:** for, while, do-while, infinite loops, and loop control statements (break, continue).
- ✓ **Goto Statement:** Usage and limitations.

Functions

- ✓ **Defining Functions:** Function declaration, definition, and calling.
- ✓ **Function Arguments:** Passing arguments by value and reference.
- ✓ **Return Types:** Returning values, void functions.
- ✓ **Recursion:** Recursive functions, base case, and recursion stack.
- ✓ **Function Overloading:** Not supported in C but useful for understanding C++.

Arrays and Strings

- ✓ **Arrays:** One-dimensional and multi-dimensional arrays, initialization, accessing elements.
- ✓ **String Handling:** Working with strings in C, functions like strcpy, strcat, strlen, strcmp.
- ✓ **Pointer and Array Relationship:** Accessing array elements using pointers.

Pointers

- ✓ **Pointer Basics**: Pointer declaration, initialization, and dereferencing.
- ✓ **Pointer Arithmetic**: Pointer increment, decrement, and memory address manipulation.
- ✓ **Pointers and Arrays**: Relationship between arrays and pointers.
- ✓ **Dynamic Memory Allocation**: malloc, calloc, free, realloc.

Structures and Unions

- ✓ **Structures**: Defining, accessing structure members, nested structures.
- ✓ **Unions**: Defining and using unions, differences between structures and unions.
- ✓ **Arrays of Structures**: Usage and access.
- ✓ **Pointers to Structures**: Using pointers with structures.

File Handling

- ✓ **File Operations**: fopen, fclose, fread, fwrite, fprintf, fscanf.
- ✓ **File Modes**: Reading, writing, and appending to files.
- ✓ **Error Handling**: Handling errors in file operations.
- ✓ **Binary Files**: Reading and writing binary data.

Advanced Topics

- ✓ **Command-Line Arguments**: Parsing arguments passed to the program.
- ✓ **Preprocessor Directives**: #include, #define, #ifdef, #pragma.
- ✓ **Bitwise Manipulation**: Bit masking, setting, clearing, and toggling specific bits.

C++ Programming

Introduction

- ✓ **History and Evolution**: Origin of C++ and its object-oriented features.
- ✓ **C++ vs C**: Key differences between C and C++ (object-oriented paradigm).
- ✓ **Structure of a C++ Program**: Basic structure, main function, and output using cout.

Object-Oriented Programming

- ✓ **Classes and Objects**: Defining and instantiating classes, using objects.
- ✓ **Access Modifiers**: Public, private, and protected access.
- ✓ **Member Functions**: Definition, declaration, and access to member functions.

Constructors and Destructors

- ✓ **Constructors:** Default, parameterized, copy constructors, constructor overloading.
- ✓ **Destructors:** Purpose, defining destructors, calling destructors automatically.

Inheritance

- ✓ **Single Inheritance:** Inheriting properties from a base class.
- ✓ **Multiple Inheritance:** C++ allows multiple base classes.
- ✓ **Access Control in Inheritance:** Using public, private, and protected access specifiers.
- ✓ **Constructor and Destructor in Inheritance:** Order of calling constructors and destructors.

Polymorphism

- ✓ **Compile-**Time Polymorphism: Function overloading, operator overloading.
- ✓ **Runtime Polymorphism:** Virtual functions, dynamic method dispatch, abstract classes.
- ✓ **Pure Virtual Functions:** Abstract base classes.

Encapsulation and Abstraction

- ✓ **Encapsulation:** Hiding data and exposing only essential functionality.
- ✓ **Abstraction:** Using abstract classes and interfaces to hide complexity.

Function Overloading and Operator Overloading

- ✓ **Function Overloading:** Same function name with different parameters.
- ✓ **Operator Overloading:** Overloading operators to perform operations on user-defined data types.

Exception Handling

- ✓ **Try, Catch, Throw:** Basic exception handling syntax.
- ✓ **Custom Exception Classes:** Defining and throwing user-defined exceptions.
- ✓ **Exception Propagation:** Handling exceptions across multiple function calls.

STL (Standard Template Library)

- ✓ **Containers**: vector, list, stack, queue, map, set.
- ✓ **Iterators**: Working with iterators to traverse containers.
- ✓ **Algorithms**: Using algorithms like sort(), find(), and reverse() with containers.

File Handling

- ✓ **File Streams**: ifstream, ofstream, fstream for reading and writing.
- ✓ **Binary Files**: Working with binary files in C++.
- ✓ **Error Handling**: Handling file errors using C++ exception handling.

Additional Topics

- ✓ **Smart Pointers**: Using unique_ptr, shared_ptr, and weak_ptr for automatic memory management.
- ✓ **Multithreading**: Introduction to multithreading using std::thread.
- ✓ **Namespaces**: Avoiding name conflicts with namespaces.
- ✓ **Typecasting**: static_cast, dynamic_cast, const_cast, reinterpret_cast.
- ✓ **Memory Management**: new, delete, handling memory leaks.