OOP(Object Oriented Programming)

C++

Syllabus :- Basics and Fundamentals

1. Hello World and Setup

Topic - Introduction to C++

History, compilers (GCC/Clang), IDE setup, basic program structure, comments.

2. Variables and Data Types

Topic - Basic Data Storage

Primitive types (int, char, float, double, bool), type modifiers (signed, unsigned, short, long), sizeof operator.

3. Operators and Expressions

Topic - Computation

Arithmetic, relational, logical, bitwise, assignment, and conditional (ternary) operators, operator precedence.

4. Input/Output (I/O)

Topic - Interacting with the User

Standard streams (cin, cout), basic I/O manipulation (\n, endl), <iostream> header.

5. Control Flow: Conditionals

Topic - Decision Making

if, else, else if statements, nested if, switch statement.

6. Control Flow: Loops

Topic - Repetition

for loop, while loop, do-while loop, loop control statements (break, continue).

7. Functions

Topic - Code Reusability

Function definition, declaration (prototype), parameters, return types, local and global scope, function overloading.

8. Arrays

Topic - Storing Collections

One-dimensional and multi-dimensional arrays, array initialization, accessing elements.

9. Pointers

Topic - Memory Management Basics

What is a pointer, declaring and initializing pointers, the address-of (&) and dereference (*) operators, pointers and arrays.

10. References

Topic - Aliases for Variables

What is a reference, reference declaration, references vs. pointers, const references.

11. Strings

Topic - Working with Text

C-style strings (character arrays) and the std::string class (from <string>).

12. Structures and Unions

Topic - Custom Data Types

Defining a struct, accessing members, nested structures, unions, difference between struct and class (initial discussion).**13. Enumerations (Enums)**Named Constants Simple enums (enum), strongly-typed enums (enum class).

Syllabus :- Intermediate and Object-Oriented Programming (OOP)

14. Introduction to OOP

Topic - Core Concepts

Abstraction, Encapsulation, Inheritance, Polymorphism.

15. Classes and Objects

Topic - Defining Custom Types

Class definition, objects, access specifiers (public, private, protected), member functions, and member variables.

16. Constructors and Destructors

Topic - Object Lifecycle

Default, parameterized, and copy constructors, initializer lists, the destructor (~ClassName()).

17. Encapsulation and Access Control

Topic - Data Hiding

Role of private and public, **Getters and Setters**, **const correctness**.

18. Static Members

Topic - Class-level Data/Functions

static member variables and static member functions, their memory allocation and usage.

19. The this Pointer

Topic - Self-Reference

Understanding this, returning *this for chaining.

20. Friends

Topic - Breaking Encapsulation

friend functions and **friend classes**.

21. Operator Overloading

Topic - Customizing Operators

Overloading binary and unary operators, stream insertion (<<) and extraction (>>) operators.

22. Inheritance

Topic - Code Extension

Base and derived classes, types of inheritance (public, private, protected), order of constructor/destructor calls.

23. Polymorphism

Topic - Runtime Behavior

Virtual functions, **Abstract base classes**, pure virtual functions, interfaces, **v-table** (briefly).

24. Exception Handling

Topic - Error Management

try, catch, throw, creating custom exception classes, exception safety.

25. File Input/Output (I/O)

Topic - Persistent Storage

fstream (ifstream, ofstream), text and binary file operations, manipulators (setw, setprecision).

Syllabus :- Advanced and Modern C++

26. Templates

Generic Programming

Function templates and **Class templates**, template specialization.

27. Standard Template Library (STL) - Part 1

Containers

Vectors, **Lists**, **Deques**, their use cases and underlying structure.

28. Standard Template Library (STL) - Part 2

Associative Containers

Maps, **Sets**, **Multimaps**, **Multisets**, ordered vs. unordered containers.

29. Standard Template Library (STL) - Part 3

Adaptors, Iterators, and Algorithms

Stacks, **Queues**, **Priority Queues**, **Iterators** (input, output, forward, bidirectional, random access), common **Algorithms** (sort, find, copy).

30. Smart Pointers (Modern C++)

Automated Memory Management

std::unique_ptr, **std::shared_ptr**, **std::weak_ptr**, RAII principle.

31. Move Semantics (C++11)

Efficiency

Rvalue references (&&), **Move constructor** and **Move assignment operator**, std::move.

32. Lambdas (C++11)

Anonymous Functions

Lambda syntax, capture list ([]), implicit and explicit captures.

33. Concurrency (C++11 and later)

Parallelism

Basic concepts of **Threads**, **std::thread**, Mutexes (std::mutex) and Locks, basic **std::future** and **std::promise**.

34. Advanced Pointers and Memory

Deeper Dive

Pointer arithmetic, dynamic memory allocation (new, delete, new[], delete[]), shallow vs. deep copy.

35. Preprocessor and Compilation

The Build Process

#include, #define, conditional compilation (#ifdef, #ifndef), linking process.

36. Best Practices and Design

Professional Code

Coding standards, **RAII (Resource Acquisition Is Initialization)**, separation of concerns.