

# Contents – C++ Concepts

## Part I: Foundations

- 1.** Introduction to C++
  - 1.1 Structure of a C++ Program
  - 1.2 Compilation and Execution
  - 1.3 Input and Output (cin, cout)
- 2.** Data and Variables
  - 2.1 Data Types
  - 2.2 Variables and Constants
  - 2.3 Type Casting
- 3.** Operators
  - 3.1 Arithmetic Operators
  - 3.2 Relational and Logical Operators
  - 3.3 Assignment Operators
  - 3.4 Bitwise Operators
  - 3.5 Ternary Operator
- 4.** Control Flow
  - 4.1 Conditional Statements (if, else if, switch)
  - 4.2 Loops (for, while, do-while)
  - 4.3 Jump Statements (break, continue, goto)

## Part II: Functions and Data Handling

- 5.** Functions
  - 5.1 Function Definition and Declaration
  - 5.2 Parameters and Return Types
  - 5.3 Inline Functions
  - 5.4 Function Overloading
  - 5.5 Default Arguments
  - 5.6 Recursion
- 6.** Arrays and Strings
  - 6.1 One-Dimensional Arrays
  - 6.2 Multi-Dimensional Arrays
  - 6.3 C-Strings (Character Arrays)
  - 6.4 std::string Class
- 7.** Pointers and References
  - 7.1 Pointer Basics
  - 7.2 Pointer Arithmetic
  - 7.3 Null and Void Pointers
  - 7.4 References
  - 7.5 Dynamic Memory Allocation (new, delete)
  - 7.6 Smart Pointers (C++11)

## **Part III: Object-Oriented Programming (OOP)**

- 8. Classes and Objects**
  - 8.1 Defining Classes
  - 8.2 Access Specifiers
  - 8.3 Constructors and Destructors
  - 8.4 Static Members
  - 8.5 Friend Functions and Classes
- 9. Inheritance**
  - 9.1 Single Inheritance
  - 9.2 Multiple Inheritance
  - 9.3 Multilevel and Hierarchical Inheritance
  - 9.4 Hybrid Inheritance
  - 9.5 Function Overriding
- 10. Polymorphism**
  - 10.1 Virtual Functions
  - 10.2 Abstract Classes and Pure Virtual Functions
  - 10.3 Operator Overloading
- 11. Advanced OOP Concepts**
  - 11.1 Encapsulation
  - 11.2 Aggregation and Composition
  - 11.3 this Pointer

## **Part IV: Advanced Language Features**

- 12. Memory Management**
  - 12.1 Copy Constructor
  - 12.2 Shallow vs Deep Copy
  - 12.3 Move Semantics (C++11)
  - 12.4 RAI (Resource Acquisition Is Initialization)
- 13. Templates**
  - 13.1 Function Templates
  - 13.2 Class Templates
  - 13.3 Template Specialization
- 14. Exception Handling**
  - 14.1 try, catch, throw
  - 14.2 Standard Exceptions
- 15. File Handling**
  - 15.1 File Streams (ifstream, ofstream, fstream)
  - 15.2 File Modes
  - 15.3 Binary File Handling

## Part V: Standard Template Library (STL)

### 16. Containers

- 16.1 Sequence Containers: vector, list, deque, array
- 16.2 Container Adapters: stack, queue, priority\_queue
- 16.3 Associative Containers: set, multiset, map, multimap
- 16.4 Unordered Containers: unordered\_set, unordered\_map

### 17. Iterators

### 18. Algorithms (sort, find, etc.)

### 19. Function Objects (Functors)

### 20. Lambda Functions

## Part VI: Modern and Advanced C++

### 21. Namespaces and Preprocessor

- 21.1 Namespaces
- 21.2 Preprocessor Directives (#define, #include)
- 21.3 Macros
- 21.4 Type Aliases (typedef, using)

### 22. Enumerations and Casting

- 22.1 enum and enum class
- 22.2 Casting Operators: static\_cast, dynamic\_cast, const\_cast, reinterpret\_cast

### 23. Modern C++ Features

- 23.1 auto Keyword
- 23.2 Range-Based For Loops
- 23.3 nullptr
- 23.4 Smart Pointers (unique\_ptr, shared\_ptr, weak\_ptr)
- 23.5 Delegating Constructors
- 23.6 constexpr

### 24. C++17 and C++20 Additions

- 24.1 Structured Bindings
- 24.2 std::optional, std::variant, std::any
- 24.3 Concepts (C++20)
- 24.4 Coroutines (C++20)

### 25. Concurrency

- 25.1 std::thread
- 25.2 Mutexes and Locks
- 25.3 Thread Synchronization

*Jibran*