

## ■ Certificate Course in Data Structures in C++ (CCDSC++)

**Duration – 2 Months**

### Course Overview

This course introduces learners to the **fundamental data structures** and their implementation in **C++**, a widely used object-oriented programming language. The program emphasizes practical coding with C++ while covering the concepts of arrays, linked lists, stacks, queues, trees, graphs, hashing, and algorithm analysis. Students will also learn how to solve real-world computational problems efficiently using C++ features like classes, templates, and the Standard Template Library (STL).

**Duration:** 2 Months (Approx. 40 Sessions)

**Prerequisites:** Basic knowledge of C++ (variables, loops, functions, classes, and objects).

---

### Course Objectives

By the end of this course, learners will be able to:

1. Understand the **importance of data structures** and their role in software development.
  2. Implement data structures using **object-oriented principles** of C++.
  3. Analyze the **time and space complexity** of algorithms.
  4. Apply different data structures to solve **real-world computational problems**.
  5. Gain familiarity with **STL containers** for efficient programming.
- 

### Course Outcomes

On successful completion, learners will:

- CO1: Demonstrate understanding of **basic and advanced data structures**.
  - CO2: Apply **C++ concepts (OOP, pointers, dynamic memory, templates)** to implement data structures.
  - CO3: Develop **problem-solving skills** using stacks, queues, linked lists, trees, and graphs.
  - CO4: Evaluate algorithms using **Big-O notation**.
  - CO5: Build **mini projects** involving multiple data structures.
-

## **Course Syllabus**

### **Module 1: Introduction & C++ Basics Refresher**

- Importance of Data Structures & Algorithms
- Recap of C++ fundamentals: Classes, Objects, Inheritance, Polymorphism
- Pointers, References, and Dynamic Memory Allocation (new, delete)
- Templates & Exception Handling in C++

### **Module 2: Arrays & Strings**

- Array Representation and Operations
- Searching Techniques: Linear Search, Binary Search
- Sorting Techniques: Bubble, Selection, Insertion, Merge, Quick Sort
- String manipulation in C++

### **Module 3: Stack**

- Concept & Applications of Stack
- Implementation using Arrays & Linked Lists
- Applications: Infix to Postfix, Postfix Evaluation
- STL Stack usage

### **Module 4: Queue**

- Concept & Applications of Queue
- Implementation using Arrays & Linked Lists
- Circular Queue, Priority Queue, Deque
- STL Queue usage

### **Module 5: Linked List**

- Singly Linked List – Creation, Insertion, Deletion, Traversal
- Doubly Linked List – Insertion, Deletion, Applications
- Circular Linked List – Operations and Applications
- Use of C++ OOP for Linked Lists

### **Module 6: Trees**

- Introduction to Trees & Terminology

- Binary Tree & Binary Search Tree (BST) – Creation, Traversals
- Balanced Trees – AVL Tree Basics
- STL Map & Set (Tree-based structures)

### **Module 7: Graphs**

- Graph Representation: Adjacency Matrix & List
- Graph Traversals: BFS & DFS
- Applications: Shortest Path, Networking Concepts

### **Module 8: Hashing & Complexity Analysis**

- Hash Functions & Hash Tables
- Collision Handling – Chaining, Open Addressing
- Algorithm Complexity Analysis: Big-O, Big-Ω, Big-Θ

### **Module 9: Mini Project & Assessment**

- Student Database using Linked List
- Expression Solver using Stack & Queue
- Simple Library/Inventory System using Trees & Hashing

---

☒ By the end of **2 months**, learners will be able to:

- Design efficient algorithms.
- Implement & test real-world applications using data structures in **C++**.
- Prepare for **coding interviews, competitive programming, and advanced courses**.