The Structured C

Introduction

- Background Part 1 (IO Devices, CPU and Memory)
- Background Part 2 (Computer Organization & Operating System)
- Why Do We Need Programming Languages
- C Introduction
- C Standards and Implementation
- Basic C Programming Terminology
- How do C Programs Run?
- First C Program
- Comment in C

Input Output in C

- Variables in C
- Variable Naming Rules
- Data types in C
- Ranges of Data Types in C
- Operator size of() in C
- Global Variables and Scope
- Const in C
- Static Variables in C
- Literals in C
- Type Conversion in C
- Swap Two Numbers

Operator

- Arithmetic Operators
- Unary Arithmetic Operators
- Comparison Operators
- Assignment Operators
- Logical Operators
- Bitwise Operator in C (AND, OR and XOR)
- Bitwise Operator in C (Left Shift, Right Shift and NOT)
- Signed Number Representation and Bitwise NOT
- Operator Precedence & Associativity
- Day Before N Day
- Sum of Natural Numbers
- Last Digit of a Number
- Practice Problem on Operators

Flow control

- If Else in C
- IF Else Example in C
- Nested If Else with Example
- Else if with example in C
- Switch in C
- Even Odd
- Largest of Three Numbers
- Leap Year
- Simple Calculator

Function

- Functions in C
- Applications of Functions
- Function Declaration & Definition
- How Functions Work
- Inline Function
- Practice Problems on C Functions
- First Digit of a Number
- Prime Factorization

Loop

- While Loop in C
- For Loop in C
- Do While Loop in C
- Break in C
- Continue in C
- Nested Loops in C
- Patterns
- Square Pattern
- Triangle Pattern
- Inverted Triangle Pattern
- Factorial of a Number
- Check for Prime
- Next Prime Number
- All Divisor of a Number
- GCD of Two Numbers
- LCM of Two Numbers
- Fibonacci Numbers
- Count Digits of a Number
- Table of a Number

Array

- Introduction to Arrays in C
- Declaring and Initializing Arrays
- Accessing Array Elements
- Size of an Array in C
- Array Traversal in C
- Different Types of Arrays
- Check if Array is Sorted
- Count Distinct in an Array
- Sum of an Array
- Average of an Array
- Maximum in an Array

Pointer

- Address and Dereference Operators in C
- Introduction to Pointers in C
- Applications of Pointers in C
- Function Parameters and Pointers
- Array Parameters and Pointers
- Pointer Arithmetic
- Void Pointer in C
- NULL in C
- Pointers versus Arrays
- Pointer to Pointer in C
- Pointer Practice Questions

Dynamic Memory Allocation

- Memory Structure of a Program
- Dynamic Memory Allocation (malloc(), calloc(), and free())
- Memory Leak

String

- String In C (Introduction)
- String Syntax, Size and Length in 'C'
- String Comparison in C
- String Copy In C
- String concatenation in C
- Pattern Searching
- strncat(), strncmp(), and strncpy()
- Substring search in C
- String Tokenization in C
- Reverse a String
- Check for Palindrome
- String: Binary to Decimal
- String: Decimal to Binary

Multi Direction Array

- Multidimensional Array in 'C'
- Passing 2D Arrays as Arguments to functions
- Transpose of a Matrix
- Matrix Multiplication

Struct and Union

- Struct in C
- Structure Variable Initialization
- Structure Arrays
- Structure Pointer
- Structure Alignment
- Structure Alignment Reason for Structure Alignment in C
- Union In C

Advanced

- Function Pointer in C
- Function Pointer in C Passing Functions as Parameters
- File Handling in C
- Reading from a File
- Writing to a file in C

Introduction to DSA

- Analysis of Algorithms(Background)
- Asymptotic Analysis
- Order of Growth
- Best, Average and Worst cases
- Asymptotic Notations
- Big O Notation
- Omega Notation
- Theta Notation
- Analysis of Common Loops in C
- Recursion Tree Method for Solving Recurrences
- Recursion Tree Method for Solving Recurrences
- More Example Recurrences
- Upper Bounds Using Recursion Tree Method
- Space Complexity

Recursion

- Recursion Introduction
- Applications of Recursion
- Recursion Practice Questions
- Recursion vs Iteration
- N to 1 Recursion
- Print 1 to N Using Recursion
- Tail Recursion
- Writing Base Case in Recursion in C

Arrays

- Introduction to Arrays in C
- Declaring and Initializing Arrays
- Accessing Array Elements
- Size of an Array in C
- Dynamic Memory Allocation (malloc(), calloc(), and free())
- Operations on Arrays - Part 1 in C
- Operations on Arrays (Part 2)
- Average of an Array
- Get Smaller Elements
- Maximum in an Array
- Second largest element in array in C
- Check if Array is Sorted
- Find the Only ODD number
- Reverse an Array
- Left Rotate an Array by One
- Left Rotate an Array by D places

Searching

- Linear Search
- Analysis of Linear Search
- Binary Search (Iterative)
- Binary Search (Recursive)
- Analysis of Binary Search
- Index of first Occurrence in Sorted
- Count Occurrences in Sorted

Sorting

- Bubble Sort
- Selection Sort
- Insertion Sort
- Merge sort introduction in C
- Merge Two Sorted Arrays in C
- Merge Sorting Algorithm
- Merge Sort Analysis
- Naive partition
- Lomuto Partition
- Hoare partition
- Quick Sort Introduction in C
- QuickSort using Lomuto Partition
- QuickSort using Hoare Partition
- QuickSort analysis
- Space Analysis of QuickSort
- Tail call elimination in QuickSort

Matrix

- Multidimensional Array in 'C'
- Passing 2D Arrays as Arguments to functions
- Matrix in snake pattern in C
- Matrix Boundary Traversal
- Transpose of a Matrix
- Rotate Matrix Anti-clockwise by 90
- Spiral Traversal of Matrix
- Search in Row-wise and Column-wise sorted matrix in C

Hashing

- Introduction to Hashing
- Hashing Application
- Direct Address Table
- Hashing Functions
- Collision Handling
- Chaining in C
- Open Addressing
- Double Hashing
- Chaining vs Open Addressing

Strings

- String In C (Introduction)
- String Syntax, Size and Length in 'C
- Escape Sequence in C
- String Comparison in C
- String Copy In C
- String concatenation in C
- strncat(), strncmp(), and strncpy()
- Reverse a String
- Substring search in C
- Pattern Searching
- Check for Palindrome
- Check for Anagram in C

Linked List

- Introduction to Linked List
- Simple Linked List Implementation in C
- Applications of Linked List
- Traversing a Linked List in C
- Recursive Traversal of Singly Linked List
- Insert at Begin of Singly Linked List
- Insert at the end of Singly Linked List
- Insert at given position in Singly Linked List
- Sorted Insert in a Singly Linked List
- Delete First Node of Singly Linked List
- Delete Last of Singly Linked List
- Search in a Linked List (Iterative and Recursive)
- Reverse a linked list iterative
- Recursive reverse a linked list (Part 2)

Doubly Linked list

- Singly Vs Doubly Linked List (Advantages & Disadvantages)
- Insert at Begin of Doubly Linked List
- Insert at End Doubly Linked List
- Delete Head of a Doubly Linked List
- Delete Last of a Doubly Linked List
- Reverse a Doubly Linked List

Circular Linked List

- Circular Linked List in C
- Circular Linked List (Advantages & Disadvantages)
- Circular Linked List Traversal in C
- Insert at Begin of Circular Linked List
- Insert at the end of Circular Linked List
- Delete Head of Circular Linked List
- Delete Kth of a Circular Linked List

Stack

- Stack Data Structure in C
- Stack Applications
- Array Implementation of Stack in C
- Implementing stack using queue
- Implementation part of Implementing stack using queue
- Infix, Prefix and Postfix Introduction
- Infix to Postfix (Simple Solution)
- Infix to Postfix (Efficient Solution)
- Evaluation of Postfix
- Infix to Prefix (Simple Solution)
- Infix to Prefix (Efficient Solution)
- Evaluation of Prefix

Queue

- Queue Data Structure
- Application of Queue Data structure
- Implementation of Queue using Linked List
- Insertion in Queues Program in C (Enqueuing)
- Deletion (Removal) in Queues Program in C (Dequeuing)
- Implementing queue using stack
- Implementation part of Implementing queue using stack
- Reverse Queue
- Implementation part of Reverse Queue
- Circular queue introduction
- Circular queue using Array
- Implementation part of Circular queue using Array
- Circular Queue using Linked Lists

Dequeue

- Deque Data Structure in C
- Deque Applications
- Array Implementation of Deque
- Implementation part of Array Implementation of Deque

Tree

- Tree Data Structure in C
- Application of Tree
- Binary Tree
- Tree Traversal
- Implementation of Inorder Traversal
- Implementation of Preorder Traversal
- Implementation of Postorder Traversal
- BFS Traversal in C
- Iterative Inorder Traversal
- Implementation of Iterative Inorder Traversal
- Iterative Preorder Traversal
- Implementation of Iterative Preorder Traversal
- Maximum in Binary Tree
- Height of Binary Tree
- Size of Binary Tree
- Search Binary Tree

Binary Search Tree

- Binary Search Tree(Background)
- Binary Search Tree (Introduction) in C
- Search in BST
- Insert in BST
- Floor in BST
- Ceil in BST in C

Heap

- Binary Heap Introduction
- Binary Heap Insert
- Binary Heap (Heapify and Extract)
- Binary Heap (Decrease Key, Delete and Build Heap)
- Build heap

Graph

- Introduction to Graph
- Graph Representation (Adjacency Matrix)
- Graph Representation (Adjacency List)
- Adjacency List implementation in C
- Adjacency Matrix and List Comparison
- Applications of BFS
- Applications of DFS