

# Documentation Tabs

Please open the provided documentation file on your laptop/PC. The file contains separate tabs for each data structure:

- Instruction
- Array
- Linked List
  - Singly Link List
  - Doubly Linked List
- Stack
- Queue
- Trees
- Sorting Algorithms

Each tab contains specific requirements and examples for the corresponding data structure implementation

## General Requirements

Students are required to implement comprehensive programs covering the following core Data Structure topics:

- Arrays
- Linked Lists
- Stacks
- Queues
- Trees
- Sorting Algorithms

Each implementation must be menu-driven, allowing users to perform all standard operations relevant to each data structure.

# Implementation Requirements

## 1. Full Implementation Required

Each program must include all standard operations pertinent to the respective data structure:

- Arrays: Insertion, deletion, display, etc.
- Stacks: Push, pop, peek operations
- Queues: Enqueue, dequeue, display functions
- Linked Lists: Insertion (beginning, middle, end), deletion, traversal
- Sorting: Multiple algorithms (Bubble, Insertion, Selection)

## 2. Menu-Driven Format

All programs must provide a clear, user-friendly menu interface. Example for Array operations:

```
===== ARRAY OPERATIONS MENU =====
1. Push to Stack
2. Pop from Stack
3. Peek Stack
4. Enqueue to Queue
5. Dequeue from Queue
6. Display Array
7. Delete Element (Pointer-Based)
8. Find Third Minimum Value
9. Eliminate Duplicates
10. Exit
```

## 3. File Structure

Submit one code file per topic, containing all related operations for that data structure. Use the following naming convention:

- <rollno>\_Array.cpp
- <rollno>\_Stack.cpp
- <rollno>\_LinkedList.cpp
- <rollno>\_Queue.cpp

Important Note: For each data structure, implement a single comprehensive program with different functions for each operation, integrated through the menu system.

## 4. Code Standards

- Include explanatory comments throughout your code
- Follow proper indentation and formatting practices
- Use meaningful variable and function names
- Implement appropriate error handling

# Documentation Requirements

## Dry Run Documentation

A manual dry run of each implementation is mandatory:

- Document step-by-step execution processes
- Record memory/stack/queue state changes
- Track variable values and transitions
- Include your full name and roll number on each page
- Submit clear images or scans compiled into a single PDF named `<rollno>_DryRun.pdf` or you can upload a dry run PDF for each data-structure.

**Important:** A sample dry run document (`sampleDryRun.pdf`) has been provided as reference. You must follow the same step-by-step

# Evaluation Process

## Viva Examination:

Viva will be conducted to evaluate your understanding.

Be prepared to:

- Explain your implementation logic in detail
- Justify your dry run steps
- Demonstrate understanding of pointer usage and operations
- Address questions regarding performance considerations

**Note:** The assignment grade will be determined solely through the viva performance.