



# Computing Lab

**CS 69201**

**Department of Computer Science and Engineering  
IIT Kharagpur**

# Fundamentals of Data Structures & Sorting

Unlocking the building blocks of efficient programming.





# C Programming

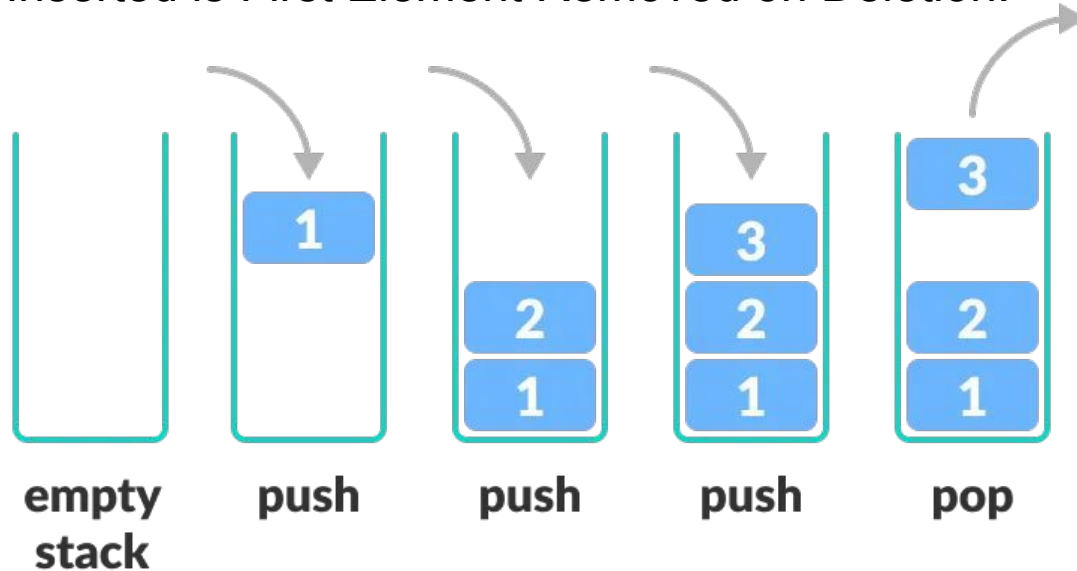
By now, you must be already familiar with these core C concepts:

- Variables, data types, and initializations
- Unary, relational, logical, and ternary operators
- Conditional statements (if/else, switch)
- Loops (for, while, do/while)
- Pointers and dynamic memory allocation
- Functions
- struct definitions
- File Handling

Let's now build on this foundation by revisiting C's data structures.

# Stacks

- Stacks operate on the Last In First Out (LIFO) Mechanism.
- Last Element Inserted is First Element Removed on Deletion.





# Stacks

## Operations:

- `void push(int x);`
- `void pop();`
- `int top();`
- `bool isEmpty();`
- `int size();`

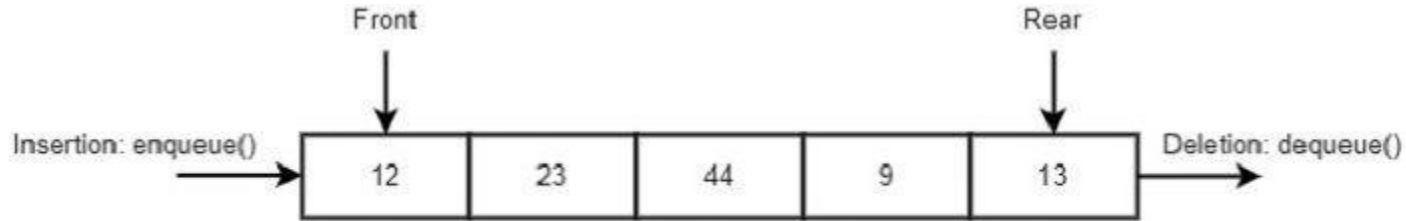
## Use-Cases / Example Questions:

- Parentheses matching
- Depth-First Search (DFS) graph traversal
- "Next Greater Element" problems

# Queues



- Queues operate on the First In First Out (FIFO) Mechanism.
- First element inserted is First Element Removed on Deletion.



Queue: FIFO Operation



# Queues

## Operations:

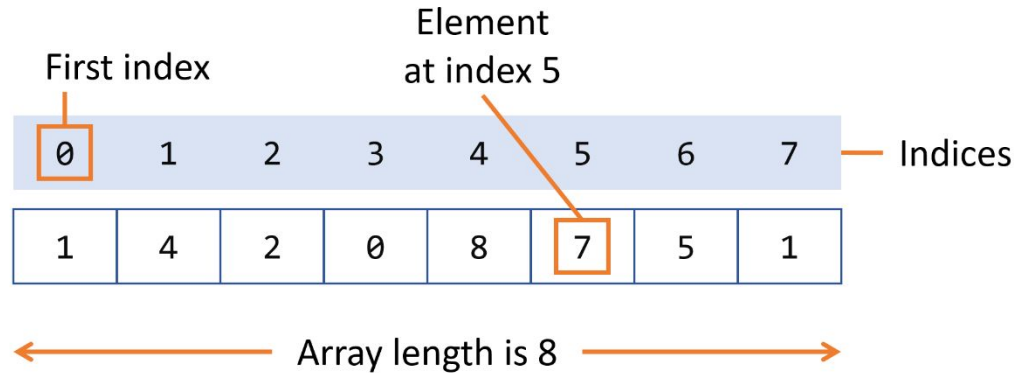
- `void enqueue(int x);`
- `void dequeue();`
- `int front();`
- `bool isEmpty();`
- `int size();`

## Use-Cases / Example Questions:

- Breadth-First Search (BFS) for graph traversal
- Scheduling algorithms
- Implementing "Sliding Window" maximum problems

# Arrays

- Arrays store elements in a contiguous memory block, allowing constant-time access by index ( $O(1)$ ).







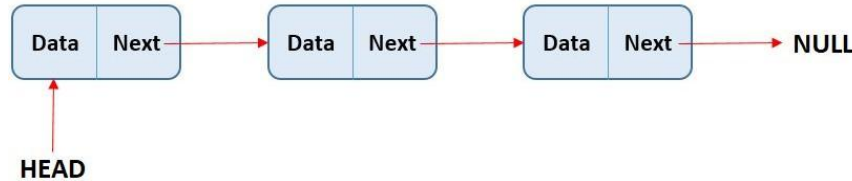
# Arrays

## Use-Cases / Example Questions:

- Two-pointer techniques for sum/partition problems
- Binary search in sorted arrays for efficient lookup
- Frequency counting and prefix sum calculations

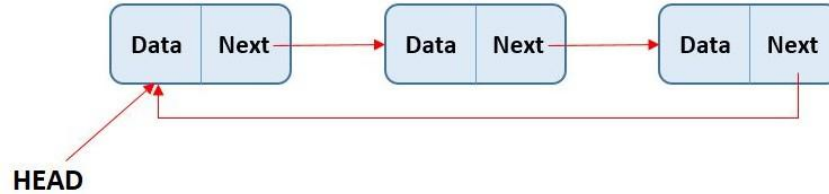
# Linked Lists

- Data structure consisting of nodes each of which contains information and address/pointer to the next node.
- Types : Single Linked List, Circular Single Linked List, Double Linked List, Circular Doubly Linked List

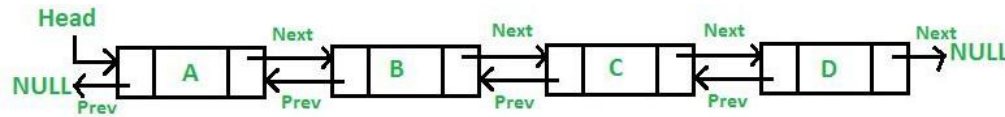


Single Linked List

# Linked Lists



## Circular Linked List



## Doubly Linked List

# Linked Lists



## Use-Cases / Example Questions:

- Reversing a singly linked list
- Detecting and removing cycles in lists
- Merging two sorted linked lists efficiently

# Sorting



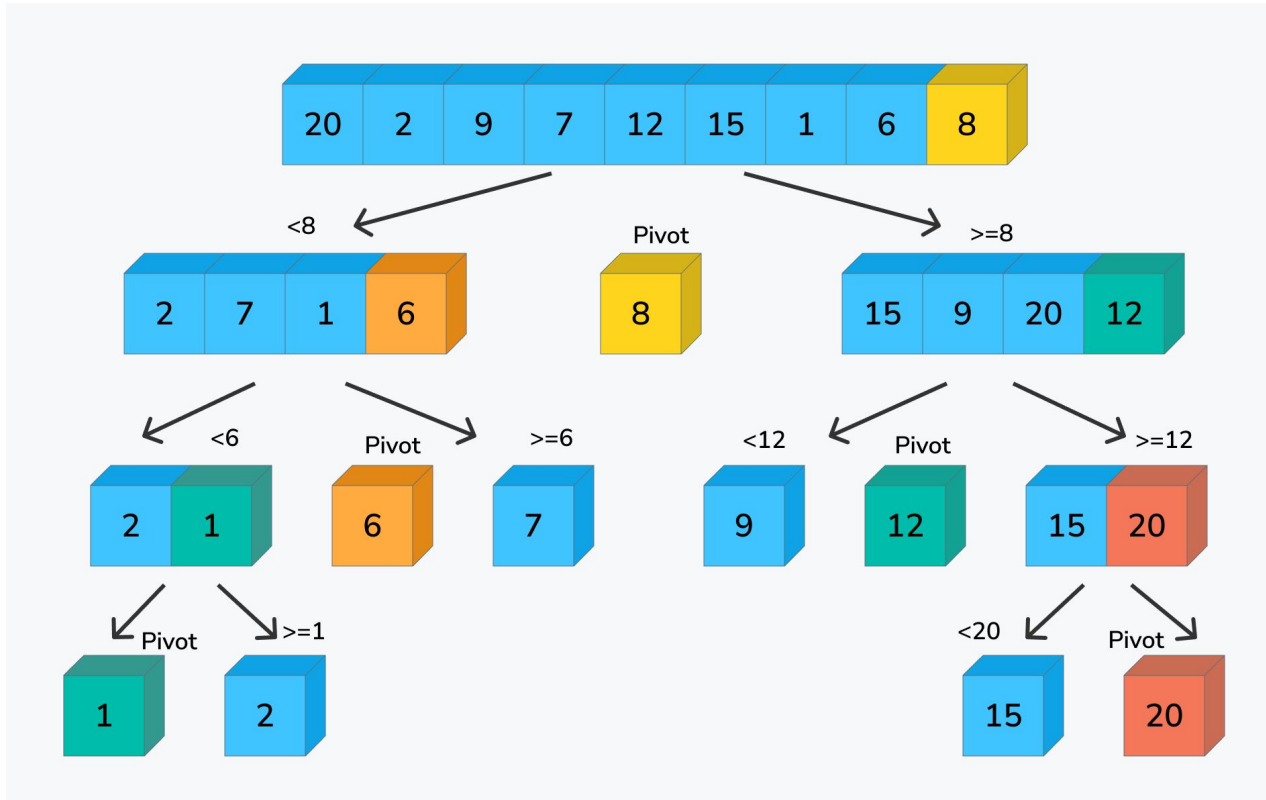
- Rearrange a given array or list of elements according to a comparison operator on the elements.
- Examples : Quick Sort, Merge Sort, Selection Sort, Bubble Sort, Radix Sort, Insertion Sort



# Quick Sort

- Sorting algorithm based on Divide and Conquer Approach
- Procedure:
  - Choose a pivot element from the array.
  - Partition the array into two parts:
    1. Elements  $\leq$  pivot on the left
    2. Elements  $>$  pivot on the right
  - Recursively apply Quick Sort to each part until sub-arrays have size  $\leq 1$ .
  - Since partitioning rearranges in place, simply concatenating the left part, pivot, and right part yields the fully sorted array.

# Quick Sort



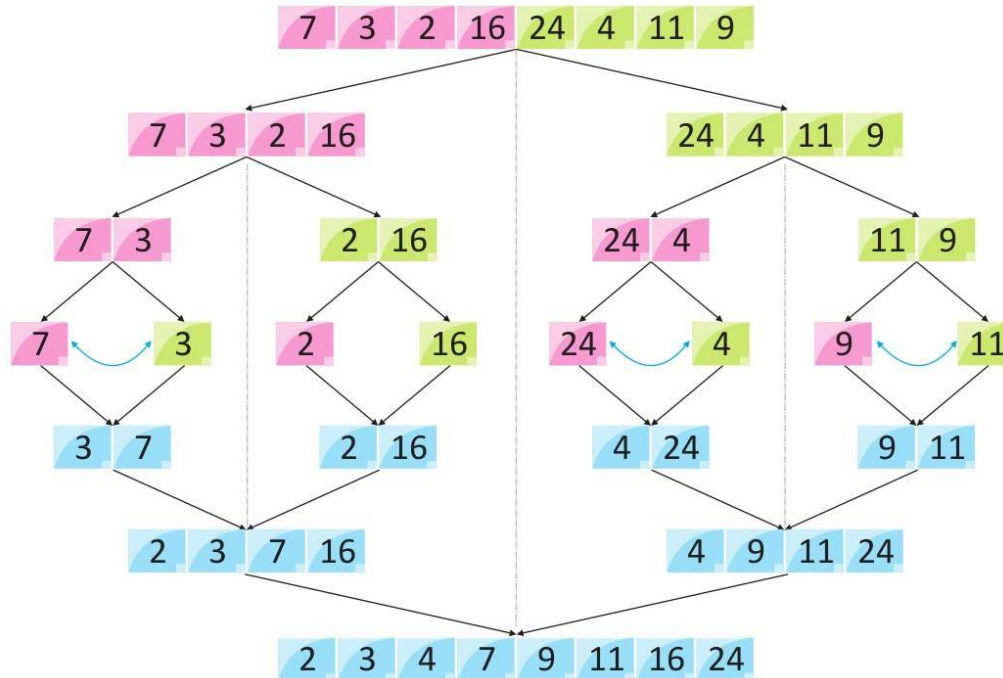


# Merge Sort

- Sorting algorithm based on Divide and Conquer Approach
- Procedure:
  - Divide the array into two halves, sort each half, and then merge the sorted halves back together.
  - For each half do the above till the array is sorted.



# Merge Sort



# How to run C program

- Save your program with `.c` extension [ex. Prog1.c]
- Use the C compiler (GCC) to compile your program as `gcc prog1.c -o prog1`
- Execute the compiled program as `./prog1`

```
[(base) → ~ gcc prog1.c -o prog1  
[(base) → ~ ./prog1  
Hello, World!  
(base) → ~ █
```

output

executable



# Additional Resources

- Stack: [mlsu.ac.in/econtents/325\\_04Stacks.pdf](https://mlsu.ac.in/econtents/325_04Stacks.pdf)
- Queue: [mlsu.ac.in/econtents/326\\_05Queues.pdf](https://mlsu.ac.in/econtents/326_05Queues.pdf)
- Arrays: [mlsu.ac.in/econtents/323\\_02Arrays.pdf](https://mlsu.ac.in/econtents/323_02Arrays.pdf)
- Linked List: [mlsu.ac.in/econtents/327\\_06Linked\\_List.pdf](https://mlsu.ac.in/econtents/327_06Linked_List.pdf)
- Strings: [Strings in C \(With Examples\)](#)
- Sorting: [Quick Sort Algorithm, Merge Sort Algorithm](#)



# Sample Questions

- Given a string containing brackets (), [] and {}, write a function to determine if the brackets are balanced and properly nested.
- Given an array of integers, find the maximum sum of any contiguous subarray
- Write a function to detect if a linked list has a cycle and, if so, return the node where the cycle begins.



# Sample Questions

- Given an array of integers, count the number of inversions (i.e., pairs  $(i, j)$  such that  $i < j$  and  $a[i] > a[j]$ ) in  $O(n \log n)$  time by modifying Merge Sort.
- Write a function to generate and print all those strings of length  $n$  over the alphabet  $\{a, b, c\}$ , in which no two consecutive symbols are the same.

For example, your function should print `ababcba`, but not `ababcca` for  $n=7$ .



# Sample Questions

- You are given an  $8 \times 8$  chessboard. Your task is to place eight queens on the board such that no queen can attack any other queen, that is, no two of the queens share the same row, column or diagonal (forward or backward).
- Generalize the problem to an  $n \times n$  board.
- Consider each cell coordinate of form  $(i,j)$  where  $1 \leq i,j \leq n$  and print those cell coordinates such that the queens cannot attack each other. First do for  $8 \times 8$  and then generalise it



# Exam on Basics of C Programming

- Next Monday: July 28, 2025
- Duration : 2 hours [2-4 PM].
- Evaluation: 4:15 PM onwards.
- Exam Syllabus : Basics of C Programming, Stacks, Queues, Arrays, Linked Lists, Sorting Techniques



**THANK YOU!**