# Example Input

## Introduction to Data Structures

### Arrays

Arrays are fundamental data structures in computer science that allow us to store multiple elements of the same type in contiguous memory locations. They provide efficient random access to elements using indices.

**Key characteristics of arrays:**

- Fixed size (in most programming languages)
- Homogeneous elements (same data type)
- Constant-time access to elements using indices

### Linked Lists

Linked lists are linear data structures where elements are stored in nodes. Each node contains data and a reference (or link) to the next node in the sequence.

**Types of linked lists:**

- Singly linked list
- Doubly linked list
- Circular linked list

Advantages of linked lists include dynamic size and efficient insertion/deletion at the beginning.

### Stacks

Stacks are abstract data types that follow the Last-In-First-Out (LIFO) principle. They support two main operations:

1. Push: Add an element to the top of the stack
2. Pop: Remove the top element from the stack

### Queues

Queues are abstract data types that follow the First-In-First-Out (FIFO) principle. They support two main operations:

1. Enqueue: Add an element to the rear of the queue
2. Dequeue: Remove an element from the front of the queue

# Time Complexity Analysis

Understanding the time complexity of algorithms is crucial for writing efficient code. We use Big O notation to describe the upper bound of an algorithm's running time.

| Complexity | Description |
|---|---|
| O(1) | Constant time |
| O(log n) | Logarithmic time |
| O(n) | Linear time |
| O(n log n) | Linearithmic time |
| O(n^2) | Quadratic time |
| O(2^n) | Exponential time |

# Recursion

Recursion is a problem-solving technique where a function calls itself to solve smaller instances of the same problem. It consists of two main parts:

1. Base case: The condition that stops the recursion
2. Recursive case: The part where the function calls itself

Understanding recursion is essential for solving complex problems and implementing advanced algorithms in computer science.