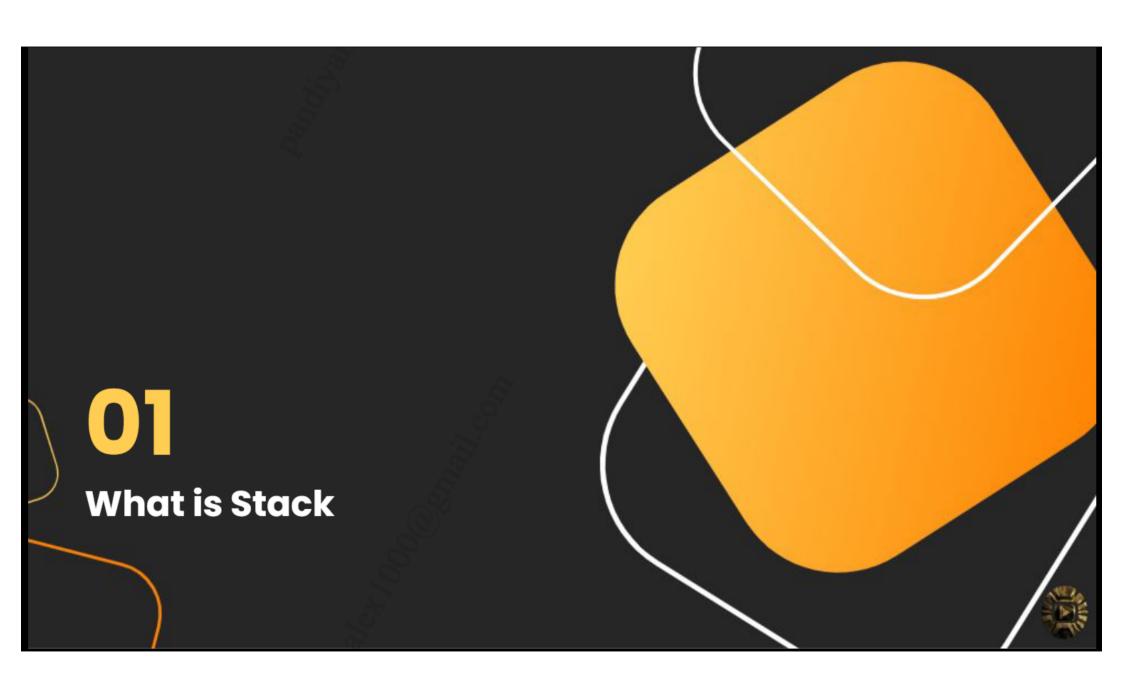


# CONTENTS

- 1. What is Stack
- **2.** Types of Stack
- **3.** Operations on Stack
- **4.** Implementations of Stack
- **5.** Advantages and Disadvantages
- **6.** Advanced Topics in stack
- **7.** Activity





# What is Stack

- A stack is a linear data structure.
- > Follows the Last In First Out (LIFO) principle.
- > The last element added is the first to be removed.
- > Efficient for managing data in constrained access patterns.

#### Examples:

- Stack of plates
- Stack of books
- Undo/Redo functionality
- Call stack in programming





# **Types of Stack**

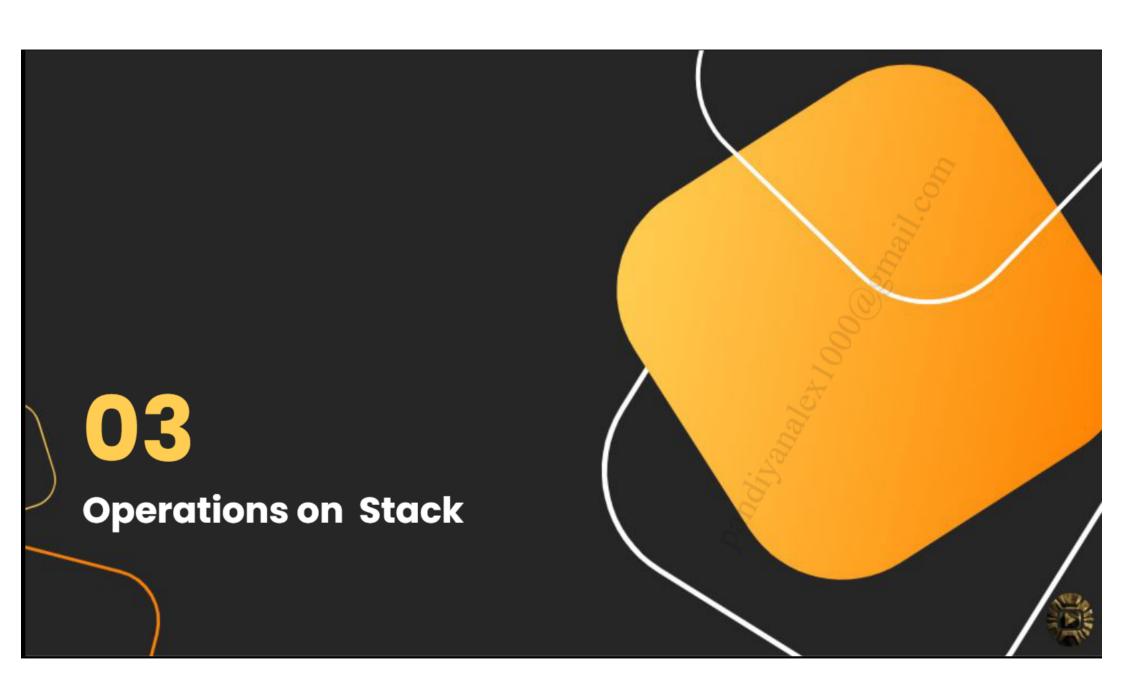
#### **Static Stacks**

- > These stacks have a fixed size defined at the time of creation.
- > They are implemented using arrays, making memory allocation straightforward and fast.
- Limitation: If the maximum size is exceeded, it results in a stack overflow.
- > Suitable for scenarios where the number of elements is known in advance.

#### **Dynamic Stacks**

- > These can grow or shrink in size during runtime, offering more flexibility.
- Implemented using linked lists where each element points to the next.
- No fixed limit, hence no overflow unless system memory is exhausted.
- Ideal for applications requiring unpredictable or varying stack sizes.





# **Operations on Stack**

### Push:

> Add an element to the top of the stack

### Pop:

Remove the top element from the stack

### Peek:

- > View the top element without removing it
- ➤ Useful to check the current top value.

Time Complexity: O(1)





# Implementations of Stack

### **Array-based Implementation**

- Uses a fixed-size array
- Quick access and manipulation
- Risk of overflow if size limit is exceeded

## **Linked List-based Implementation**

- ➤ Each node points to the next
- Dynamic memory allocation
- > No overflow unless memory is exhausted





# Advantages and Disadvantages

## **Memory Usage:**

- Arrays may waste space if underutilized
- ➤ Linked lists use extra memory for pointers

### **Performance Considerations:**

- Arrays offer fast access, but less flexible
- ➤ Linked lists are flexible but have overhead





# Advanced Topics in Stack

### **Stacks in Data Processing**

- > Parsing Expressions (e.g., converting infix to postfix)
- > Backtracking Algorithms (e.g., maze solving, recursion)

#### **Stack Overflow and Underflow**

#### Causes:

- Overflow: pushing onto a full stack
- ➤ Underflow: popping from an empty stack

#### **Prevention Techniques:**

- Check capacity before push
- Check emptiness before pop
- Use try-catch for exception handling



# Advanced Topics in Stack

## **Error Handling Techniques**

- Validate operations with conditions
- Use exception handling to prevent crashes
- > Provide meaningful error messages to users

#### **Variants of Stacks**

#### **Double-Ended Stacks:**

- Allow insertion/deletion from both ends
- > Useful for palindrome checking and special applications

#### **Concurrent Stacks:**

- Support multi-threaded access
- Critical for high-performance software systems

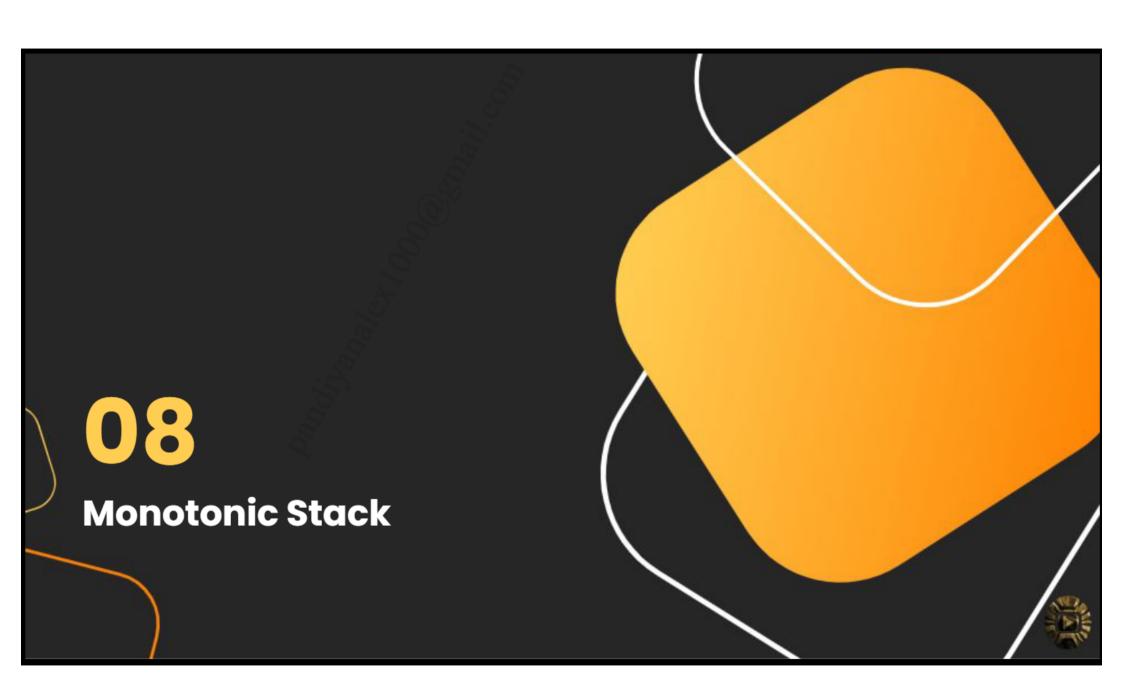




# Activity

- **∢/>** Min Stack
- Basic Calculator
- Decode String
- **▲ | Maximum Frequency Stack**





# Monotonic Stack

A stack that maintains its elements in either increasing or decreasing order — hence the term monotonic.

A powerful algorithmic pattern used primarily for problems involving

- > next greater/smaller element in an array,
- histogram problems,
- interval-related tasks.

### **Types**

- > Monotonic Increasing Stack: Each new element is greater than or equal to the top of the stack.
  - Used to find next smaller elements.
- Monotonic Decreasing Stack: Each new element is less than or equal to the top of the stack.
  - Used to find next greater elements.



# • Monotonic Stack

## Classic Problems Using Monotonic Stack

- Next Greater Element
- ➤ Largest Rectangle in Histogram
- Daily Temperatures
- Trapping Rain Water
- ➤ 132 Pattern
- > Sliding Window Maximum

#### **Benefits**

- > Time complexity often becomes **O(n)**, because each element is pushed and popped at most once.
- > Easy to implement with Stack or Deque.





# Activity 2

- Next Greater Element
- Daily Temperatures
- Largest Rectangle in Histogram
- **≺ |** ► <u>Asteroid Collision</u>
- Implement Queue using Stack
- **◆ |** ► Backspace String compare

