**Stack Data Structure**

A **Stack** is a linear data structure that follows a particular order in which the operations are performed. The order may be **LIFO(Last In First Out)** or **FILO(First In Last Out)**. **LIFO** implies that the element that is inserted last, comes out first and **FILO** implies that the element that is inserted first, comes out last.

It behaves like a stack of plates, where the last plate added is the first one to be removed. **Think of it this way:**

Pushing an element onto the stack is like adding a new plate on top.

Popping an element removes the top plate from the stack.

**LIFO(Last In First Out) Principle**

The LIFO principle means that the last element added to a stack is the first one to be removed.

* New elements are always pushed on top.
* Removal (pop) also happens only from the top.
* This ensures a strict order: last in → first out.

**Real-world examples of LIFO:**

* **Stack of plates** – The last plate placed on top is the first one you pick up.
* **Shuttlecock box** – The last shuttlecock inserted is the first one taken out, since both operations happen from the same end.

**Types of Stack:**

**Fixed Size Stack**

* A fixed size stack has a predefined capacity.
* Once it becomes full, no more elements can be added (this causes **overflow**).
* If the stack is empty and we try to remove an element, it causes **underflow**.
* Typically implemented using a static array.

Example: Declaring a stack of size 10 using an array.

**Dynamic Size Stack**

* A dynamic size stack can grow and shrink automatically as needed.
* If the stack is full, its capacity expands to allow more elements.
* As elements are removed, memory usage can shrink as well.
* Can be implemented using:  
  -> **Linked List** → grows/shrinks naturally.  
  ->**Dynamic Array** (like vector in C++ or ArrayList in Java) → resizes automatically.

Example: Stack implementation using linked list or resizable array.

**Common Operations on Stack:**

In order to make manipulations in a stack, there are certain operations provided to us.

* **push()**to insert an element into the stack.
* **pop()**to remove an element from the stack.
* **top()**Returns the top element of the stack.
* **isEmpty()**returns true if stack is empty else false.
* **size()** returns the size of the stack.

A screenshot of a computer

AI-generated content may be incorrect.

A table with text and numbers

AI-generated content may be incorrect.

A diagram of a push operation

AI-generated content may be incorrect.

// Driver Code

// creating a stack using array

let st = [];

// pushing elements into the stack

st.push(1); // pushes 1

st.push(2); // pushes 2

st.push(3); // pushes 3

A screenshot of a computer

AI-generated content may be incorrect.

// Driver Code

// creating a stack using array

let st = [];

st.push(1);

st.push(2);

st.push(3);

// Printing current top element

console.log(st[st.length - 1].toString() + " ");

A diagram of a stack

AI-generated content may be incorrect.

// Driver Code

let st = [];

// if stack is empty returns true else false

if (st.length === 0) {

console.log("Stack is empty.");

} else {

console.log("Stack is not empty.");

}

// Inserting value 1 to the stack top

st.push(1);

// if stack is empty returns true else false

if (st.length === 0) {

console.log("Stack is empty.");

} else {

console.log("Stack is not empty.");

}

**Stack using Array**

A stack is a [linear data structure](https://www.geeksforgeeks.org/dsa/introduction-to-linear-data-structures/) that follows the [Last-In-First-Out (LIFO)](https://www.geeksforgeeks.org/dsa/lifo-principle-in-stack/) principle. It can be implemented using an array by treating the end of the array as the top of the stack.

**Declaration of Stack using Array**

A stack can be implemented using an array where we maintain:

* An integer array to store elements.
* A variable capacity to represent the maximum size of the stack.
* A variable top to track the index of the top element. Initially, top = -1 to indicate an empty stack.

A screenshot of a computer code

AI-generated content may be incorrect.