Stack

What is a Stack?

A **Stack** is a data structure where you can only:

- **Add** elements from the **top**
- **Remove** elements from the **top**

This rule is called:

LIFO – Last In, First Out

The **last item you put** is the **first item you take out**.

Real-Life Examples of a Stack

1. Stack of Plates

Imagine plates stacked in a cafeteria:

```
Top →     [ Plate 3 ]
          [ Plate 2 ]
Bottom→[ Plate 1 ]
```

- You **add** a plate on top
- You **take** a plate from top
- You **cannot** remove the bottom plate directly

This is **exactly how a Stack works**.

---

2. Undo feature in Mobile Apps / PC

When you type:

A
AB
ABC

Each action is pushed into a **stack**.

When you press **Undo**:

- Last action (C) is removed first
- Then B
- Then A

Undo uses **LIFO stack**.

---

3. Browser Back Button

Every webpage you visit is stored in a stack:

Home
→ About Us
→ Services
→ Contact

If you click **Back**, it goes:

Contact ← removed first
Services
About Us
Home

Last visited page → removed first.

---

4. Pile of Books

Books stacked one on top of another.

You always:

- Put new book on top
- Remove book from top

You can't pull a book from the bottom.

Simple Definition

A **stack** is a data structure that stores elements in **Last In, First Out (LIFO)** order.

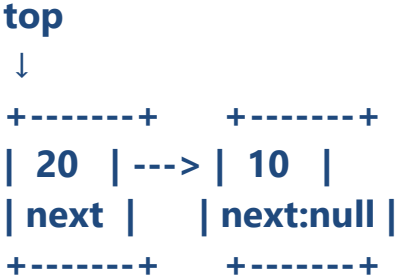
Operations in Stack

Operation	Meaning
push(x)	Add item to top
pop()	Remove top item
peek()	See top item without removing
isEmpty()	Check if stack has no items
isFull()	Check if stack is full (for array stack)

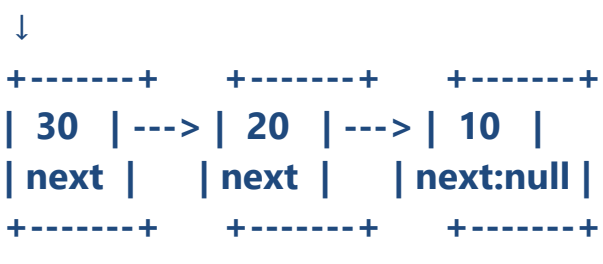
After stack.push(10)

After stack.push(20)

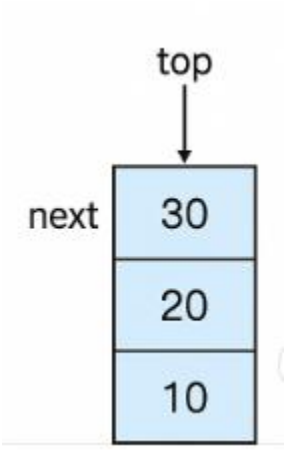
New node (20) becomes the new top → it points to old top (10):



After stack.push(30)



Stack (top → bottom) = 30, 20, 10



peek() → returns 30

top points to node with value 30.

After pop() (removes 30)

Top moves to next node (20):

```
top
 ↓
+-------+     +-------+
|  20   | ---> |  10   |
| next  |     | next:null |
+-------+     +-------+
```

Popped value = **30**

---

peek() after pop → returns 20

Top now points to 20.