DAILY DSA | DAY-13 | Stacks|
-GOPALKRISHNA A

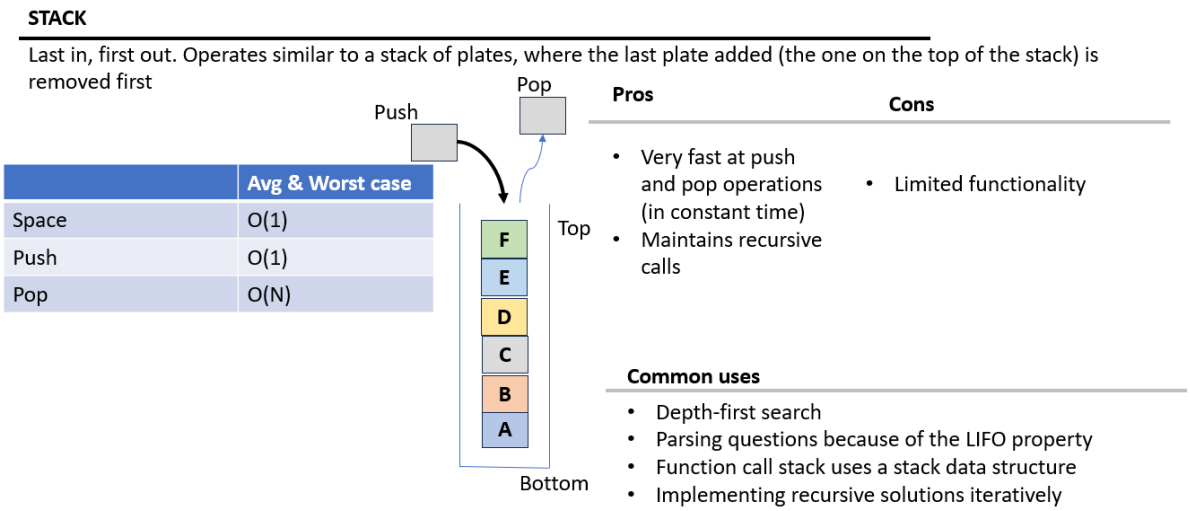
The tasks that need to be completed have piled up and become a stack? Let's see how computers handle stacks

Stacks is an Abstract data type, that is popular used to organize the information/Data. They provide means of storing different types of data in unique ways and methods to access either all or distinct parts of it.

Stack: A stack is a linear data structure used for storing data that follows either the LIFO (Last in First Out) or FILO (First in Last Out) principle. This means that **the last element that is inserted is the first element to be removed.**

Main operations in the stack:

- **push()** - To insert an element into the stack the operation
- **pop()** - To remove an element from the stack the operation
- **top()** - Returns to the top element of the stack
- **is Empty()** - Return true if the stack is empty else false
- **size()** - Returns the size of the stack



Types of stack:

1. Fixed-size stack: In a Fixed-size stack, the size of the stack is predetermined and cannot be changed during runtime. These are implemented using arrays.

2. Dynamic size stack: A stack whose size and capacity can change during runtime. These are implemented using linked lists.

Top of the stack:

- When a new element is added to the stack, it is placed on top of the existing elements. Similarly, when an element is removed from the stack, the topmost element is removed first. The top of the stack is always the element that is currently accessible for viewing or manipulation
- The pointer through which the elements are accessed, inserted, and deleted in the stack is called the **"Top of the stack"**. It is the pointer to the topmost element of the stack.

Calculating memory usage:
The memory footprint of a stack or queue depends on its underlying implementation - typically arrays are used to build fixed-size stacks and queues, whereas linked lists are used to build variable-sized stacks and queues.

Advantages of the stack:

- Easy implementation: Uses arrays or linked lists and its operations are simple to understand and implement.
- Efficient memory utilization: Stacks use a contiguous block of memory, making it more efficient in memory utilization.

Disadvantages of the stack:

- Stack data structures are not suitable for applications that require accessing elements in the middle of the stack, like searching or sorting algorithms
- The stack data structure can result in stack overflow if too many elements are pushed onto the stack, and it can result in stack underflow if too many elements are popped from the stack.