***Stack Data Structure***

**Stack**

It is a linear data structure that follows a particular order in which the operations are performed.

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

*This strategy states that the element that is inserted last will come out first.* ***You*** *can take a pile of plates kept on top of each other as a real-life example. The plate which we put last is on the top and since we remove the plate that is at the top, we can say that the plate that was put last comes out first.*

**Basic 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 stack.

**Push:**

Adds an item to the stack. If the stack is full, then it is said to be an Overflow condition.

**Algorithm for push:**

begin

if stack is full

return

endif

else

increment top

stack[top] assign value

end else

end procedure

**Pop:**

Removes an item from the stack. The items are popped in the reversed order in which they are pushed. If the stack is empty, then it is said to be an Underflow condition.

**Algorithm for pop:**

begin

if stack is empty

return

endif

else

store value of stack[top]

decrement top

return value

end else

end procedure

**Top:**

Returns the top element of the stack.

**Algorithm for Top:**

begin

return stack[top]

end procedure

**isEmpty:**

Returns true if the stack is empty, else false.

**Algorithm for isEmpty**:

begin

if top < 1

return true

else

return false

end procedure

**Understanding stack practically:**

*There are many real-life examples of a stack. Consider the simple example of plates stacked over one another in a canteen. The plate which is at the top is the first one to be removed, i.e. the plate which has been placed at the bottommost position remains in the stack for the longest period of time. So, it can be simply seen to follow the LIFO/FILO order.*

**Complexity Analysis:**

* **Time Complexity**

| **Operations** | **Complexity** |
| --- | --- |
| push() | O(1) |
| pop() | O(1) |
| isEmpty() | O(1) |
| size() | O(1) |

**Types of Stacks:**

* **Register Stack:** This type of stack is also a memory element present in the memory unit and can handle a small amount of data only. The height of the register stack is always limited as the size of the register stack is very small compared to the memory.
* **Memory Stack:** This type of stack can handle a large amount of memory data. The height of the memory stack is flexible as it occupies a large amount of memory data.