|  |  |
| --- | --- |
| **Name** |  |
| **CMS ID** |  |
| **Date** |  |

**Objectives:**

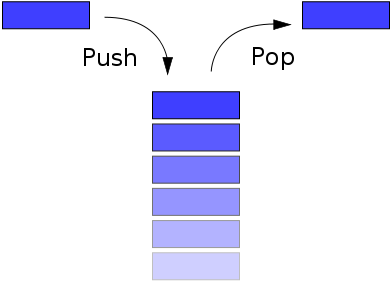
After completing this Lab students will able to

1. Understand the concept and usage of Stack in programming.
2. Implement Stacks using Array.
3. Implement Stacks using linked lists.

**Stacks**

The *stack* is a very common data structure used in programs. By *data structure*, we mean something that is meant to *hold* data and provides certain *operations* on that data.

It is a sequence of items that are accessible at only one end of the sequence. Think of a stack as a collection of items that are piled one on top of the other, with access limited to the topmost item. A stack inserts item on the top of the stack and removes item from the top of the stack. It has LIFO (Last-In First-Out) ordering for the items on the stack.



Type of Stack:

* Linear Stack (Array)
* Linked List Stack

**Objective – 2: *Implementing of Stacks using Arrays.***

**Basic Stack Operations**

There are basic 8 stack operations listed below.

* Create Stack
* Is stack Empty
* Is stack Full
* Push
* Pop
* Top
* Count Stack Elements
* Destroy Stack

**Tasks**

1. **Create a program that takes name of a user and present it in reverse order.**
2. **Implement the balanced parenthesis problem that was discussed in class. You should take combination of different opening and closing parenthesis from user as input, store it in a string object and then evaluate whether the parenthesis are balanced or not. Note that user can enter other characters too, however, you need to work on parenthesis only.**
3. **Implement infix to postfix expression conversion that takes operands and operators as input, store them in a string object and store its postfix conversion in another string object.**
4. **Implement postfix expression evaluator that takes a string object that contain postfix expression as input and return the result of that expression.**