East West University
Department of Computer Science & Engineering
A/2, Jahurul Islam Avenue, Jahurul Islam City, Aftabnagar, Dhaka-1212

Lab Manual :
Course Code : CSE207
Course Title : Data Structure
Instructor : Amit Kumar Das, Senior Lecturer, CSE

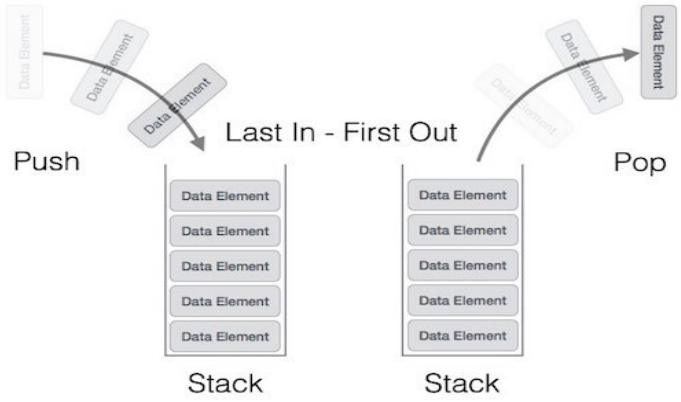
Objective:

The objective of this lab is to provide basic concept of stack. At the end of the lab, students are able:

- To learn how to create a stack
- To learn how to perform push and pop operation in stack
- To learn how to use stack for parsing unmatched parenthesis in an algebraic expression
- To learn how to use stack for reversing data.

Stack:

A stack is an Abstract Data Type (ADT), commonly used in most programming languages. It is named stack as it behaves like a real-world stack, for example – a deck of cards or a pile of plates, etc. A real-world stack allows operations at one end only. This feature makes it LIFO data structure. In stack terminology, insertion operation is called PUSH operation and removal operation is called POP operation.



Now you have to perform the following lab task on stack:

Exercise 1:

Create a Menu

Create a menu that will display all the exercises given below (Exercise 2 to Exercise 5) as a list and prompt user to select any desired option. The menu can be designed in below format.

1. Insert data/ push stack
2. Print stack
3. Pop stack

Exercise 2:

Push Operation

Adding a new data/node in stack is a more than one step activity. First, create a node using structure and find the location where it has to be inserted. Then input the data and store it in the allocated memory space. Insert the node at the beginning of the previously inserted node.

Exercise 3:

Pop Operation

After completing exercise 1 you have a newly created stack. Now perform the pop operation on it.

Exercise 4:

Parsing Unmatched Parenthesis

One of the most important applications of stack is parsing. Parsing is any logic that breaks data into independent piece for further processing. So parsing unmatched parenthesis is a common problem of parsing. When parentheses are unmatched then there will be two types of error: the opening parentheses are unmatched or the closing parenthesis are missing. No write a program using stack that will make sure that all parentheses are well pared. For example,

Input	**Output**
((A+B)/C	Opening parentheses not end
(A+B)/C)	Closing parentheses not matched

Exercise 5:

Reversing Data

Reversing data requires that a given set of data be reordered so that the first and last elements are exchanged. The idea of reversing data can be used in solving classical problem such as converting a decimal number to a binary number. Now write a program using stack that will convert decimal number to binary number. For example:

Input	**Output**
45	101101
4	100