National University of Computer and Emerging Sciences

Lab Manual

for

Data Structures

Course Instructor	Ma'am Abeeda Akram
Lab Instructor(s)	Mr. Sohaib Ahmad Ms. Ammara Nasir
Section	BCS-3G
Semester	FALL 2022

Department of Computer
Science FAST-NU, Lahore,
Pakistan

Lab Manual 04

Objectives:

After performing this lab, students shall be able to revise:

✔ Implementation of a Stack ADT using Linked List and Array

✔ Different applications of Stack ADT

Problem

Question1:

Part A: (Stack using Link List & Array)

1. Implement a template Stack class using Link List with one top pointer.
   - **bool Push (T Val)** // Add an element in Stack. Returns False if push operation is unsuccessful otherwise True.
   - **bool Pop ()** // Remove top element from Stack. Returns true if the operation is successful otherwise false if the stack is empty with some error message.
   - **bool Top(T&) //returns** the top element but does not remove it from the stack, the topmost element from the stack via the parameter passed by reference. It returns false via a return statement if there is no element in the stack, else it returns true and assigns the topmost element to the parameter passed by reference.
   - **bool IsEmpty()**
   - **bool IsFull()**
   - **Stack()** //default constructor. Creates a stack of default size 10
   - **Stack(int size**) // Parameterized Constructor. Creates a stack of size = size
   - **~Stack() //** Destructor
2. Implement a template Stack Using an Array and implement all the above functionalities.

Part B: (Expression Evaluation)

Design and Implement the following functions for Expression Evaluation. You will take an expression of variable size as input from the user in the form of a string.

1. **Parenthesis_Check:**
   Implement a function **bool check (char expression[])** using the Stack class, determining if a given expression is correctly parenthesized. The function takes in a character array. The function returns true if the brackets are applied correctly and properly balanced and false otherwise. The expression contains: 3 type of brackets ( ) , [] and {}, English alphabets, numbers and operators.
   For example,

(x + y) * (w / z)	TRUE
A * { B / C } – () (0)	TRUE
x + y) + (a – c)	FALSE
) a + b * 3 (	FALSE

2. **Infix_to_Postfix:**
   Implement a function **void infixtopostfix(char infix[])** that will convert the infix notation to postfix notation and print the result.
   For example:
   Input Expression: 2+3*4
   Output Expression: 234*+

3. **Evaluate_Postfix:**
   Implement a function **void evaluate_postfix(char postfix[])** to evaluate a postfix expression and output the final results. Evaluation of postfix expression using stack has been explained in the figure below as well.
   For example
   Input Expression: 234*+
   Output Expressoin:14

Input	**Stack** *(top is on the left)*	
2 3 4 * +	*empty*	Push 2
3 4 * +	2	Push 3
4 * +	3 2	Push 4
* +	4 3 2	Pop 4, pop 3, do 3 *4 , push 12
+	12 2	Pop 12, Pop 2, do 2 + 12, push 14
	14	

4. **Menu:** This function will provide appropriate options to users and perform operations 1 to 3 accordingly.

Good Luck!