**Data Structures Lab**

***Session 7***

**Course:** Data Structures (CS2001) **Semester:** Fall 2021

**Instructor:** Shahbaz Siddiqui  **T.A:** N/A

**Note:**

* + - * Lab manual cover following below Stack and Queue topics

**{Stack with Array and Linked list , Application of Stack, Queue with Array and Linked List , Application of Queue }**

* Maintain discipline during the lab.
* Just raise hand if you have any problem.
* Completing all tasks of each lab is compulsory.
* Get your lab checked at the end of the session.

**Stack with Array**

**Sample Code of Stack in Array**

class Stack {

    int top;

public:

    int a[MAX]; // Maximum size of Stack

    Stack() { top = -1; }

    bool push(int x);

    int pop();

    int peek();

    bool isEmpty();

};

bool Stack::push(int x)

{

    if (top >= (MAX - 1)) {

        cout << "Stack Overflow";

        return false;

    }

    else {

        a[++top] = x;

        cout << x << " pushed into stack\n";

        return true;

    }

}

int Stack::pop()

{

    if (top < 0) {

        cout << "Stack Underflow";

        return 0;

    }

    else {

        int x = a[top--];

        return x;

    }

}

int Stack::peek()

{

    if (top < 0) {

        cout << "Stack is Empty";

        return 0;

    }

    else {

        int x = a[top];

        return x;

    }

}

bool Stack::isEmpty()

{

    return (top < 0);

}

**Task-1:**

A. Design a Main class of upper code which perform the below task

1. Insert 10 Integers values in the stack
2. If the Insert input reach the Highest index of Array display the message Stack overflow
3. Remove the Inserted values till the Last value and print the message that stack is empty

**Stack with Linked list**

**Sample Code of Stack in Array**

struct Node

{

int data;

struct Node\* link;

};

struct Node\* top;

// Utility function to add an element

// data in the stack insert at the beginning

void push(int data)

{

// Create new node temp and allocate memory

struct Node\* temp;

temp = new Node();

// Check if stack (heap) is full.

// Then inserting an element would

// lead to stack overflow

if (!temp)

{

cout << "\nHeap Overflow";

exit(1);

}

// Initialize data into temp data field

temp->data = data;

// Put top pointer reference into temp link

temp->link = top;

// Make temp as top of Stack

top = temp;

}

**Task-2:**

A. Design a Main class of upper code which perform the below task

1. Insert 10 Integers values in the stack
2. Write a utility function for upper code to display all the inserted integer values in the linked list in forward and reverse direction both
3. Write utility function to pop top element from the stack

**Application of Stack (convert infix expression to postfix)**

**Sample Pseudocode**

Begin

initially push some special character say # into the stack

for each character ch from infix expression, do

if ch is alphanumeric character, then

add ch to postfix expression

else if ch = opening parenthesis (, then

push ( into stack

else if ch = ^, then //exponential operator of higher precedence

push ^ into the stack

else if ch = closing parenthesis ), then

while stack is not empty and stack top ≠ (,

do pop and add item from stack to postfix expression

done

pop ( also from the stack

else

while stack is not empty AND precedence of ch <= precedence of stack top element, do

pop and add into postfix expression

done

push the newly coming character.

done

while the stack contains some remaining characters, do

pop and add to the postfix expression

done

return postfix

End

**Code Snippet**

#include<bits/stdc++.h>

using namespace std;

//Function to return precedence of operators

int prec(char c)

{

if(c == '^')

return 3;

else if(c == '\*' || c == '/')

return 2;

else if(c == '+' || c == '-')

return 1;

else

return -1;

}

int main()

{

string exp = "a+b\*(c^d-e)^(f+g\*h)-i";

infixToPostfix(exp);

return 0;

}

**Task-3:**

1. Use the Upper code snippet implement the utility function with the help of array based stack **infixToPostfix** by using sample pseudocode