// University of Arkansas at Little Rock

// Department of Computer Science

// CPSC 2380: Data Structures and Algorithms

// Spring 2020

// Project 2: Infix to Postfix Conversion

// Due Date: April 30, 2020

// Name: Divyajitsinh Mahida

// T-number (Last 4 Digits): T00631929

// Description of the Program (2-3 sentences): This program converts infix

// arithmetic expressions to postfix. Each operator and operand is

// separated by at least one spaces

// Date Written: 04/22/2020

// Date Revised:04/24/2020

#include<iostream>

#include<cstring>

#include<stack>

using namespace std;

// this funtion is used to find precedence of operators

// we assume that only +,-,\*,/ operations are used

int precedence(char op)

{

if (op == '\*' || op == '/')

return 2;

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

return 1;

else

return -1;

}

string InfixToPostfix(char\* str)

{

// stack used for converting infox to postfix

stack<char> st;

int n = strlen(str);

// this string stores the postfix

string res = "";

// prcoess str char by char

for (int i = 0; i < n; i++)

{

// if we find a delimiter sapce then do nothing

if (str[i] == ' ')

continue;

// character is an operand add it to output string.

if ((str[i] >= '0' && str[i] <= '9'))

res += str[i];

// character is an ‘(‘, push it to the stack.

else if (str[i] == '(')

st.push('(');

// if we find a ')' then pop stack to find its matching '(' bracket

else if (str[i] == ')')

{

while (!st.empty() && st.top() != '(')

{

char op = st.top();

st.pop();

res += op;

}

if (st.top() == '(')

{

char x = st.top();

st.pop();

}

}

// if we get a operator then we pop from stack until

// a lower precedence operator is found on the top of stack.

else {

while (!st.empty() && precedence(str[i]) <= precedence(st.top()))

{

char op = st.top();

st.pop();

res += op;

}

//after we have popped lowe precedence operators we push this to stack

st.push(str[i]);

}

}

// after processing infix we simply pop the remaining elements from stack

// and add then to postfix

while (!st.empty())

{

char x = st.top();

st.pop();

res += x;

}

//return postfix

return res;

}

//Driver program to test above functions

int main()

{

char\* str = new char[80];

while (1)

{

cout << "Enter an arithmetic equation..." << endl;

cin.getline(str, 80);

cout << InfixToPostfix(str) << endl;

}

return 0;

}

A screenshot of a computer screen

Description automatically generated