C++ Programming Assignments – L1

Expression Evaluator

Design and implement an expression evaluator to evaluate arithmetic expression.

Expression can have unary and binary operators.

To start with, you can assume following as possible operators.

Unary +, Unary -, Binary +, Binary -, multiplication * and division /

Example of valid expressions:

- -10+20*8-90/2
- (-10+20)*8-90/2
- -10
- +5

Hint: two components –

a. Parser – To scan the input string, check for syntax, and possibly/optionally produce an intermediate representation
b. Evaluator – To evaluate the expression tree.

Code:

```
/*****************************************************************************

                    Online C++ Compiler.
          Code, Compile, Run and Debug C++ program online.
Write your code in this editor and press "Run" button to compile and execute it.

*****************************************************************************/

#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;
}

// The main function to convert infix expression
//to postfix expression
void infixToPostfix(string s)
{
   std::stack<char> st;
   st.push('N');
   int l = s.length();
   string ns;
   for(int i = 0; i < l; i++)
   {
      // If the scanned character is an operand, add it to output string.
      if((s[i] >= 'a' && s[i] <= 'z')||(s[i] >= 'A' && s[i] <= 'Z'))
      ns+=s[i];

      // If the scanned character is an '(', push it to the stack.
      else if(s[i] == '(')

      st.push('(');

      // If the scanned character is an ')', pop and to output string from the stack
      // until an '(' is encountered.
      else if(s[i] == ')')
      {
         while(st.top() != 'N' && st.top() != '(')
         {
            char c = st.top();
            st.pop();
           ns += c;
         }
         if(st.top() == '(')
         {
            char c = st.top();
            st.pop();
         }
      }
```

```
        //If an operator is scanned
        else{
            while(st.top() != 'N' && prec(s[i]) <= prec(st.top()))
            {
                char c = st.top();
                st.pop();
                ns += c;
            }
            st.push(s[i]);
        }

    }
    //Pop all the remaining elements from the stack
    while(st.top() != 'N')
    {
        char c = st.top();
        st.pop();
        ns += c;
    }

    cout << ns << endl;

}

//main() function implementation
int main()
{
    string exp;
    cout<<"Enter the expression";
    cin>>exp;
    //= "a+b*(c^d-e)^(f+g*h)-i";
    infixToPostfix(exp);
    return 0;
}
```

Code: Output

Run Debug Stop Share Save { } Beautify Language C++

main.cpp

```
            ns += c;
        }
        if(st.top() == '(')
        {
            char c = st.top();
            st.pop();
        }
    }

    //If an operator is scanned
    else{
        while(st.top() != 'N' && prec(s[i]) <= prec(st.top()))
        {
            char c = st.top();
            st.pop();
            ns += c;
        }
        st.push(s[i]);
    }

}
//Pop all the remaining elements from the stack
```

input

```
Enter the expression-10+20*8-90/2
-1+02*0-89/02


...Program finished with exit code 0
Press ENTER to exit console.
```