#include <iostream>

#include <stack>

using namespace std;

int priority (char alpha){

if(alpha == '+' || alpha =='-')

return 1;

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

return 2;

if(alpha == '^')

return 3;

return 0;

}

string convert(string infix)

{

int i = 0;

string postfix = "";

// using inbuilt stack< > from C++ stack library

stack <int>s;

while(infix[i]!='\0')

{

// if operand add to the postfix expression

if(infix[i]>='a' && infix[i]<='z'|| infix[i]>='A'&& infix[i]<='Z')

{

postfix += infix[i];

i++;

}

// if opening bracket then push the stack

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

{

s.push(infix[i]);

i++;

}

// if closing bracket encounted then keep popping from stack until

// closing a pair opening bracket is not encountered

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

{

while(s.top()!='('){

postfix += s.top();

s.pop();

}

s.pop();

i++;

}

else

{

while (!s.empty() && priority(infix[i]) <= priority(s.top())){

postfix += s.top();

s.pop();

}

s.push(infix[i]);

i++;

}

}

while(!s.empty()){

postfix += s.top();

s.pop();

}

cout << "Postfix is : " << postfix; //it will print postfix conversion

return postfix;

}

int main()

{

string infix = "((a+(b\*c))-d)";

string postfix;

postfix = convert(infix);

return 0;

}

Method 2

Output

Postfix is : abc\*+d-