

Infix to Postfix conversion using stack

Infix to Postfix

Infix: $A + (B * C)$

↓
 $A + (B * C)$

↓
 $A (B * C) +$

↓
Postfix: $A B C * +$

Order of operation

- 1) Parentheses $() \{ \} []$
- 2) Exponents (right to left)
- 3) Multiplication and division (left to right)
- 4) Addition and Subtraction (left to right)

</>

$(a - b / c) * (a / k - l)$

→



01/24

- If operand
print
- If '('
push to stack
- If ')'
pop from stack and print until '(' is found
- If operator
pop from stack and print until an operator with less precedence is found

push

Infix to Postfix

```
#include<iostream>
#include<string>
#include<stack>
using namespace std;
int precedence(char a)
{
    if(a=='^') return 3;
    else if(a=='/' || a=='*') return 2;
    else if(a=='+' || a=='-') return 1;
    return -1;
}
void infixToPostFixConversion(string str)
{

```

```

    stack<char> stk;
    string result="";
    for(int i=0;i<str.length();i++)
    {
        //operand
        if(str[i]>='a' && str[i]<='z' || str[i]>='A'&&str[i]<='Z' || str[i]>='0' && str[i]<='9')
        {
            result+=str[i];
        }else
        {
            //(
            if(str[i]=='(')
            {
                stk.push(str[i]);
            }else
            {
                // )
                if(str[i]==')')
                {
                    while(!stk.empty() && stk.top()!='(')
                    {
                        result+=stk.top();
                        stk.pop();
                    }
                    stk.pop();
                }else
                {
                    // operator
                    {
                        while(!stk.empty() && precedence(stk.top())>=precedence(str[i]))
                        {
                            result+=stk.top();
                            stk.pop();
                        }
                        stk.push(str[i]);
                    }
                }
            }
        }
    }
    while(!stk.empty())
    {
        result+=stk.top();
        stk.pop();
    }
    cout<<"Result :"<<result;
}

int main()
{
    //string g="a+b*(c^d-e)^(f+g*h)-i";
    string g="(a-b/c)*(a/k-l)";
    infixToPostFixConversion(g);
    return 0;
}

```

Infix to Prefix

Sabse phele string ko reverse kiya

For '(' → ')' or ')' → '(' replace liya

Or postfix wale logic mai ek jagah change kiya highlighted hai

Or at the end waps string ko reverse kiya

```
#include<stack>
#include<algorithm>
using namespace std;
int precedence(char a)
{
    if(a=='^') return 3;
    else if(a=='/' || a=='*') return 2;
    else if(a=='+' || a=='-') return 1;
    return -1;
}
string infixToPostFixConversion(string str)
{
    stack<char> stk;
    string result="";
    for(int i=0;i<str.length();i++)
    {
        //operand
        if(str[i]>='a' && str[i]<='z' || str[i]>='A' && str[i]<='Z' || str[i]>='0' && str[i]<='9')
        {
            result+=str[i];
        }else
        {
            //(
            if(str[i]=='(')
            {
                stk.push(str[i]);
            }else
            {
                //)
                if(str[i]==')')
                {
                    while(!stk.empty() && stk.top()!='(')
                    {
                        result+=stk.top();
                        stk.pop();
                    }
                    stk.pop();
                }else
                {
                    // operator
                    {
                        while(!stk.empty() && (precedence(stk.top())>precedence(str[i]) ||
                        precedence(stk.top())>precedence('^')))
                        {
                            result+=stk.top();
                            stk.pop();
                        }
                    }
                }
            }
        }
    }
    result+=stk.top();
    stk.pop();
    reverse(result.begin(), result.end());
    return result;
}
```

```

    }
    stk.push(str[i]);
}

}
while(!stk.empty())
{
    result+=stk.top();
    stk.pop();
}
return result;
}

string infixToPrefixConversion(string str)
{
    reverse(str.begin(),str.end());
    for(int i=0;i<str.length();i++)
    {
        if(str[i]=='(') str[i]=')';
        else if(str[i]==')') str[i]='(';
    }
    string g=infixToPostFixConversion(str);
    reverse(g.begin(),g.end());
    return g;
}

int main()
{
    //string g="a+b*(c^d-e)^(f+g*h)-i";
    string g="x+y*z/w+u";
    cout<<infixToPrefixConversion(g);
    return 0;
}

```