**Experiment Number: 3c**

**Title:** Conversion from Prefix to Postfix Expression Using Recursion.

**Problem Statement:** Write a C++ program to convert a prefix expression into postfix expression using recursion.

**Algorithm:**

1. Read the Prefix expression in reverse order (from right to left).
2. If the symbol is an operand, then push it onto the Stack.
3. If the symbol is an operator, then pop two operands from the Stack  
   Create a string by concatenating the two operands and the operator after them.  
   **string = operand1 + operand2 + operator**  
   And push the resultant string back to Stack.
4. Perform the above steps recursively until end of Prefix expression.

**Code:**

#include<iostream>

#include<stack>

using namespace std;

string prefix;

string postfix;

int i;

int l;

stack<string> s;

bool isoperator(char ch)

{

if(ch=='+'||ch=='-'||ch=='\*'||ch=='/'||ch=='^')

return true;

return false;

}

void convert(int i)

{

if(i<0)

{

postfix = s.top();

s.pop();

cout<<"Postfix expression :"<<postfix<<endl;

return;

}

if(isoperator(prefix[i]))

{

string p1 = s.top();

s.pop();

string p2 = s.top();

s.pop();

string temp = p1+p2+prefix[i];

s.push(temp);

}

else{

s.push(string(1,prefix[i]));

}

i = i - 1;

convert(i);

}

int main()

{

cout<<"Enter the prefix expression :\n";

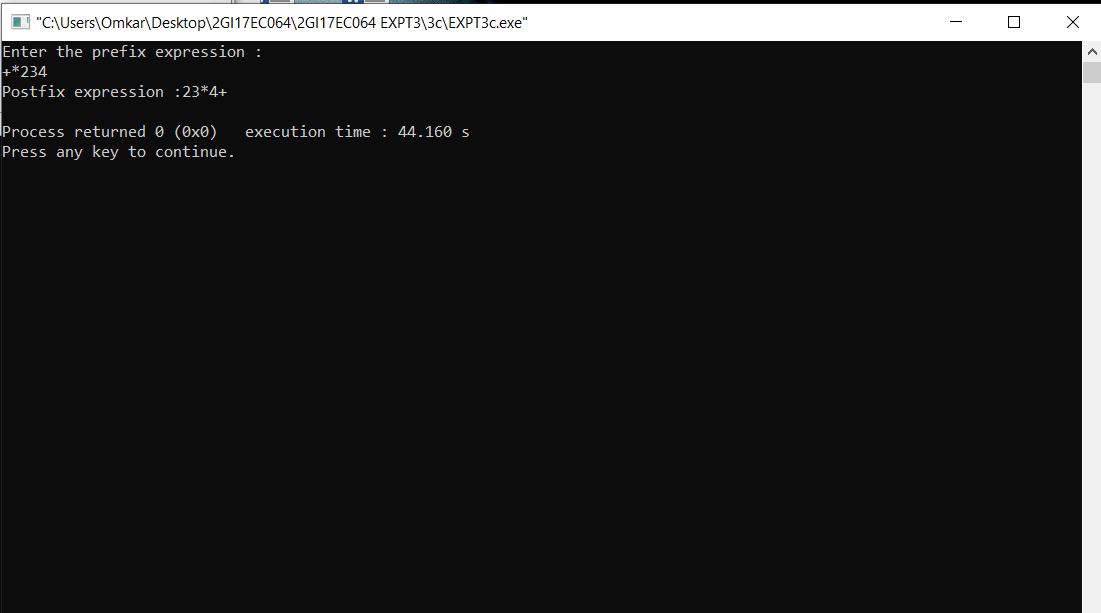
cin>>prefix;

l = prefix.size();

convert(l-1);

}

**Output:**



**Analysis:**

We have implemented the conversion from prefix to postfix expression using recursive approach but it has some limitations associated with it:

1. Recursion takes a lot of stack space, usually not considerable when the program is small and running on a PC.
2. Recursion uses more processor time.
3. Recursive solution is always logical and it is very difficult to trace (debug and understand).
4. This implementation considers a single digit we can extend it to many digits by modifying the code.