PRN No. : 124B2B012

Name : Khairnar Atharva Anil.

Title: Write a program to convert infix expression to postfix, infix expression to prefix and evaluation of postfix and prefix expression.

Code:

a) Infix to Prefix conversion:

```
#include<iostream>
#include<string>
using namespace std;

int priority(char c)
{
   if((c=='*')||(c=='/'))
   {
      return 3;
   }
   if((c=='+')||(c=='-'))
   {
      return 2;
   }
   return 0;
}

string infixToPostfix(string s)
{
   string postfix="";
   int top=-1;
   char stack[10];
   for(char c:s)
   {
      if((c=='+')|| (c=='*')|| (c=='-')|| (c=='/'))
      {
         if(priority(stack[top]) >= priority(c))
         {
            postfix+=stack[top];
            top--;
         }
         top++;
         stack[top]=c;
      }
```

```
        else
        {
            postfix+=c;
        }
    }

    while(top!=-1)
    {
        postfix+=stack[top];
        top--;
    }
    cout<<"\npostfix expression:"<<postfix;
    return postfix;
}


string reverse1(string s)
{
    string rev="";
    for(int i=s.size()-1;i>=0;i--)
    {
        rev+=s[i];
    }
    return rev;
}

int main()
{
    string s="a+b*c";
    cout<<"Original String:"<<s;
    string rev=reverse1(s);
    cout<<"\nReverse of original:"<<rev;
    string i=infixToPostfix(rev);
    string f=reverse1(i);
    cout<<"\nAgain reverse of postfix:"<<f;
}
```

b) Infix to Postfix conversion:

```
#include<iostream>
#include<string>
using namespace std;

int priority(char c)
{
   if((c=='*')||(c=='/'))
   {
      return 3;
   }
   if((c=='+')||(c=='-'))
   {
      return 2;
   }
   return 0;
}

void infixToPostfix(string s)
{
   string postfix="";
   int top=-1;
   char stack[10];
   for(char c:s)
   {
      if((c=='+')|| (c=='*')|| (c=='-')|| (c=='/'))
      {
         if(priority(stack[top]) >= priority(c))
         {
            postfix+=stack[top];
            top--;
         }
          top++;
         stack[top]=c;
      }
      else
      {
         postfix+=c;
      }
   }

   while(top!=-1)
   {
```

```
        postfix+=stack[top];
        top--;
    }
    cout<<"postfix string:"<<postfix;
}

int main()
{
    string s="a+(b*c)+d";
    infixToPostfix(s);
}
```

Output:

a)

```
Output

/tmp/jaVhaCQOr1.o
Original String:a+b*c
Reverse of original:c*b+a
postfix expression:cb*a+
Again reverse of postfix:+a*bc

=== Code Execution Successful ===
```

b)

```
Output

/tmp/xzDJCrnAic.o
postfix string:a(bc)*d++

=== Code Execution Successful ===
```