Department : Computer Engineering

Class : SE

Subject : Data Structure Lab

Name: Kartiki Uday Khare.

Roll no: 21494

Batch: H4

Assignment No : 10

- **Problem Statement:**
  Implement C++ program for expression conversion as infix to postfix and its evaluation using stack based on given conditions: 1. Operands and operator, both must be single character. 2. Input Postfix expression must be in a desired format. 3. Only '+', '-', '*' and '/ ' operators are expected.

- **Code:**

```
#include <iostream>
#define max 100
using namespace std;

template<class T>
class Stack //Creating Stack
{
   public:
      T arr[max];
      int top;
      Stack();
      void push(T val);
      void pop();
      T peep();
      bool isempty();
      void display();
      int pr(T op);
};

template<class T>
Stack<T>::Stack()
{
   top=-1;
}

template<class T>
void Stack<T>::push(T val)
{
   if (top==max-1)
      cout<<"\nStack is Full\n";
```

```
    arr[++top]=val;
}

template<class T>
void Stack<T>::pop()
{
    if(top!=-1)
        top--;
}

template<class T>
T Stack<T>::peep()
{
    if(top!=-1)
        return arr[top];
}

template<class T>
bool Stack<T>::isempty()
{
    if(top==-1)
        return true;
    else
        return false;
}

template<class T>
void Stack<T>::display()
{
    if (top!=-1)
    {
        cout<<"\nElements of Stack are:\n";
        int x=top;
        while(x!=-1)
        {
            cout<<arr[x]<<" ";
            x--;
        }
    }
}

template<class T>
int Stack<T>::pr(T op)
{
    switch(op)
    {
    case '+':
    case '-':return 1;
    case '*':
    case '/':return 2;
    case '^':return 3;
    }
    return 0;
}

int main()
{
```

```
   Stack<int>s;

   string pstx,a;
   char op;
   int flag=0;
   cout<<"Enter Infix Expression:"<<endl;
   cin>>a;
   int i,ans,oper1,oper2;
   ans=0;
    //bool ans=s.isempty();
    for(i=0;i<a.length();i++)
    {
       if(s.isempty() && (a[i]=='(' || a[i]=='{' || a[i]=='[' || a[i]=='<'))
          s.push(a[i]);
       else if((s.peep()=='(' && a[i]==')') || (s.peep()=='{' && a[i]=='}') || (s.peep()=='[' && a[i]==']')
|| (s.peep()=='<' && a[i]=='>'))
          s.pop();
       else
          continue;
    }
    if(s.isempty())
       flag=1;
    else{
       flag=0;
       cout<<"Please Enter a Valid Expression";
    }
    //Conversion:
    if(flag==1)
    {
       for(i=0;i<a.length();i++)
       {
          op=a[i];
          if( op>='0' && op<='9' )
             pstx+=op;
          else if(op=='+' || op=='-' || op=='/' || op=='*' || op=='^')
          {
             while( s.pr(s.arr[s.top])>=s.pr(op) )
             {
                pstx+=s.peep();
                s.pop();
             }
              s.push(op);
          }
          else if(op=='(')
             s.push(op);
          else if(op==')')
          {
             while(s.arr[s.top]!='(')
             {
                pstx+=s.peep();
                s.pop();
             }
             s.pop();
          }
       }
       while(!s.isempty()){
          pstx+=s.peep();
```

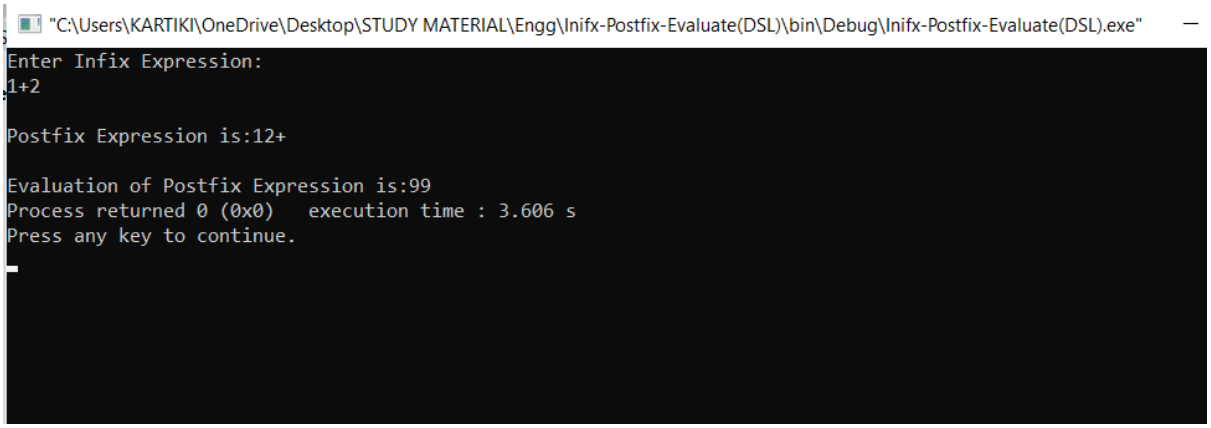
```
            s.pop();
        }
        cout<<"\nPostfix Expression is:"<<pstx<<"\n\n";

    //Evaluation of Postfix

    for(i=0;i<pstx.length();i++)
    {
        op=pstx[i];
        if(op>='0' && op<='9')
            s.push(op);
        else if(op=='+' || op=='-' || op=='/' || op=='*')
        {
            oper1=s.peep();
            s.pop();
            oper2=s.peep();
            s.pop();
            switch(op)
            {
                case '+': ans=oper2+oper1;
                          s.push(ans);
                          break;
                case '-': ans=oper2-oper1;
                          s.push(ans);
                          break;
                case '*': ans=oper2*oper1;
                          s.push(ans);
                          break;
                case '/': ans=oper2/oper1;
                          s.push(ans);
                          break;
            }
        }
    }
    cout<<"Evaluation of Postfix Expression is:"<<s.peep();

   }
    return 0;
}
```

- **Output:**


"C:\Users\KARTIKI\OneDrive\Desktop\STUDY MATERIAL\Engg\Inifx-Postfix-Evaluate(DSL)\bin\Debug\Inifx-Postfix-Evaluate(DSL).exe"
Enter Infix Expression:
1+2
Postfix Expression is:12+
Evaluation of Postfix Expression is:99
Process returned 0 (0x0)   execution time : 3.606 s
Press any key to continue.