Group D: Assignment No 12

Implement C++ program for expression conversion as infix to postfix 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.**

```
#include<iostream>
#include<string.h>
using namespace std;
class stackop
{  char  st[20],st1[20];  int top;
   public:
    void input();
    void push(char a);
    void pop();
    int pri(char b);

 };
 int stackop::pri(char b)
 {   if(b=='-')
       return 1;
        if(b=='+')
       return 2;
        if(b=='/')
       return 3;
        if(b=='*')
       return 4;
 }

void stackop::input()
{   char ch[20];   top=-1;   int f=1,l,i=0,j=0;
     cout<<"\n enter the expression\n";
     cin>>ch;
     l=strlen(ch);



     while(i<l)
     {     f=1;
```

```
        if(isalpha(ch[i]))
          {    cout<<ch[i]; st1[j]=ch[i]; j++;     }
          if(isdigit(ch[i]))
          {  cout<<ch[i];  st1[j]=ch[i]; j++;   }
        if(ch[i]=='(')
          {    push(ch[i]); }
         if(ch[i]==')')
          {
            while(st[top]!='(')
             {   cout<<st[top]; st1[j]=st[top]; j++; pop();}
                 pop();
          }
          if((ch[i]=='+')||(ch[i]=='-')||(ch[i]=='*')||(ch[i]=='/'))
          {    while(f==1)
               {
                if(top==-1)
                {   push(ch[i]); f=0;   }
                else
                {   if(st[top]=='(')
                    {   push(ch[i]);  f=0; }
                    else
                    {
                          if((pri(ch[i]))>(pri(st[top])))
                          {         push(ch[i]); f=0;       }
                          else
                          {      cout<<st[top]; st1[j]=st[top]; j++;  pop();       }

                    }
                }
               }
          }
     i++;
    }
   while(top!=-1)
    { cout<<st[top]; st1[j]=st[top]; j++; pop();   }  cout<<"\n";

}


void stackop::push(char a)
{    top++; st[top]=a;   }
void stackop::pop()
{  top--;         }
int main()
{  stackop s;
   s.input();
```

```
   return 0;
}
```

OUTPUT:

enter the expression:

```
((a+b)*(c-d))/e                    // Infix Expression
ab+cd-*e/                          //Postfix Expression
```