Code Instruction:

For both of the following problems, an operand is assumed to be a single digit. And an operator is limited to '**+**', '**-**', '*****', '**/**' (these 4 types). Also, for usage of parentheses, use only '(' for opening and ')' for closing.

In light of these remarks, an algebraic expression for example can be written like below:

2 * 4 + (6 – 3) / 3 **or** **a * b + (D – Q) / z**

1. Write a code to convert an infix algebraic expression to both postfix and prefix using the help of Stack or Queue.

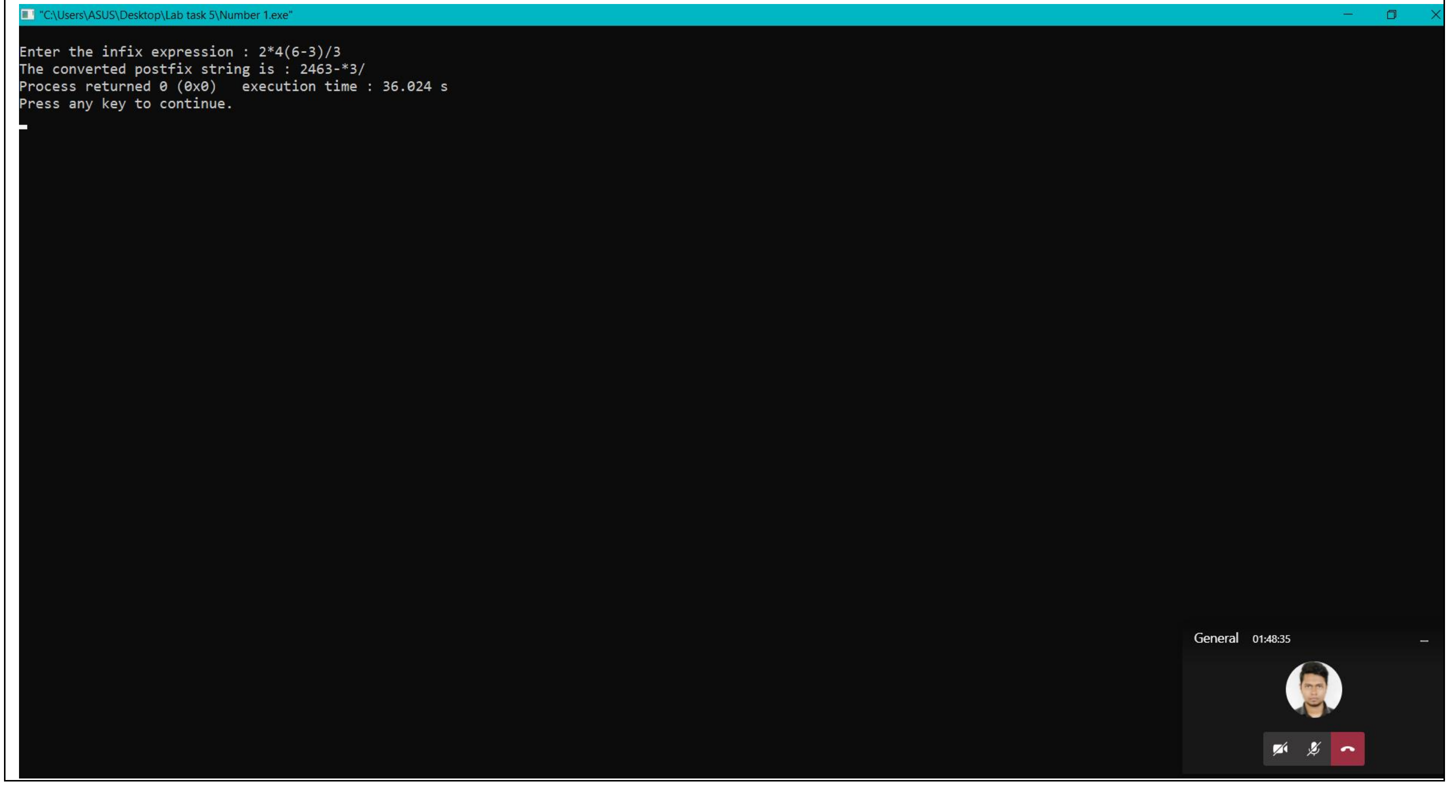
Your code here:

```
#include<iostream>
#include<string>
#define MAX 20
using namespace std;

char stk[20];
int top=-1;
// Push function here, inserts value in stack and increments stack top by 1
void push(char oper)
{
  if(top==MAX-1)
  {
    cout<<"stackfull!";
  }

  else
  {
    top++;
    stk[top]=oper;
  }
}
// Function to remove an item from stack.  It decreases top by 1
char pop()
{
  char ch;
  if(top==-1)
  {
    cout<<"stackempty!";
  }
  else
  {
    ch=stk[top];
    stk[top]='\0';
    top--;
    return(ch);
  }
  return 0;
```

```
}
int priority ( char sign )
{
   if(sign == '+' || sign =='-')
   {
      return(1);
   }

   if(sign == '*' || sign =='/')
   {
      return(2);
   }

   if(sign == '$')
   {
      return(3);
   }

   return 0;
}
string convert(string infix)
{
   int i=0;
   string postfix = "";
   while(infix[i]!='\0')
   {
      if(infix[i]>='a' && infix[i]<='z'|| infix[i]>='A'&& infix[i]<='Z'|| infix[i]>='0' && infix[i]<='9')

      {
         postfix.insert(postfix.end(),infix[i]);
         i++;
      }
      else if(infix[i]=='(' || infix[i]=='{'  || infix[i]=='[')
      {
         push(infix[i]);
         i++;
      }
      else if(infix[i]==')' || infix[i]=='}'  || infix[i]==']')
      {
         if(infix[i]==')')
         {
            while(stk[top]!='(')
            {              postfix.insert(postfix.end(),pop());
            }
            pop();
            i++;
         }
         if(infix[i]==']')
         {
            while(stk[top]!='[')
            {
               postfix.insert(postfix.end(),pop());
            }
            pop();
            i++;
         }

         if(infix[i]=='}')
         {
            while(stk[top]!='{')
            {
               postfix.insert(postfix.end(),pop());
            }
            pop();
            i++;
         }
      }
      else
      {
         if(top==-1)
         {
            push(infix[i]);
            i++;
         }

         else if( priority(infix[i]) <= priority(stk[top])) {
            postfix.insert(postfix.end(),pop());

            while(priority(stk[top]) == priority(infix[i])){
               postfix.insert(postfix.end(),pop());
               if(top < 0) {
                  break;
               }
            }
            push(infix[i]);
            i++;
         }
         else if(priority(infix[i]) > priority(stk[top])) {
            push(infix[i]);
            i++;
         }
      }
   }
   while(top!=-1)
   {
      postfix.insert(postfix.end(),pop());
   }
   cout<<"The converted postfix string is : "<<postfix;
   return postfix;
}

int main()
{
   string infix, postfix;
   cout<<"\nEnter the infix expression : ";
   cin>>infix;
```

```
    postfix = convert(infix);
    return 0;
}
```

Your whole Screenshot here: (Console Output):



"C:\Users\ASUS\Desktop\Lab task 5\Number 1.exe"

```
Enter the infix expression : 2*4(6-3)/3
The converted postfix string is : 2463-*3/
Process returned 0 (0x0)   execution time : 36.024 s
Press any key to continue.
```