Government College of Engineering, Karad Programming for Problem Solving Lab

Nanekar Saurabh Rajesh

20141212

I1

**Experiment No. 7**

**Title**: Implement stack as an ADT to perform expression conversion and evaluation for postfix to infix.

**Outcome:** Students can perform expression conversion and evaluation for postfix to infix and its related applications.

**Theory:**

**Convert Postfix to Infix Expression:**

**Example**:

Input: Postfix expression:  A B +

Output: Infix expression- (A + B)

Input: Postfix expression:  ABC/-AK/L-\*

Output: Infix expression: ((A-(B/C))\*((A/K)-L))

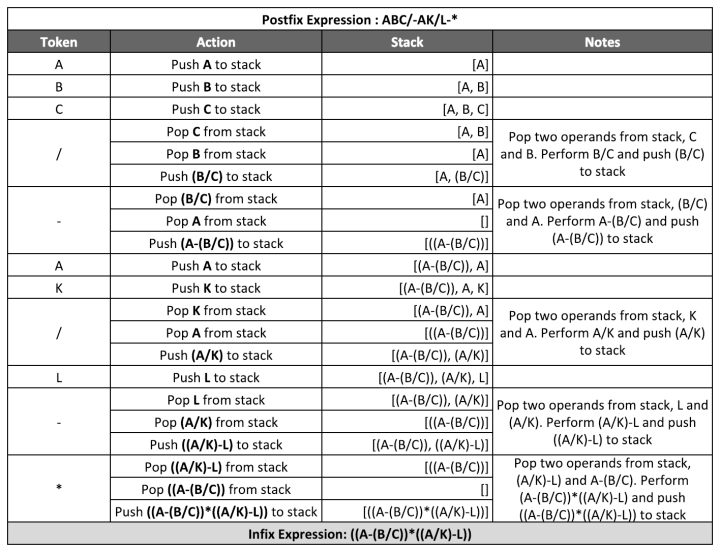
Input: Postfix expression:  A B C \* + D +

Output: Infix expression: A + B \* C + D

**Approach**: [Use Stack](https://algorithms.tutorialhorizon.com/stack-java-class-explained/)

**Algorithm**:

1. Iterate the given expression from left to right, one character at a time
2. If a character is operand, push it to stack.
3. If a character is an operator,
4. pop operand from the stack, say it’s s1.
5. pop operand from the stack, say it’s s2.
6. perform (s2 operator s1) and push it to stack.
7. Once the expression iteration is completed, initialize the result string and pop out from the stack and add it to the result.
8. Return the result.
9. Please walk through the example below for more understanding.



**Analysis:**



**List of similar programs: Solve any one.**

1. Write a C program to reverse a string using Stack.
2. Write a C program for conversion of decimal to binary using Stack.

**Title Program:** Implement stack as an ADT to perform expression conversion and evaluation for postfix to infix.

**Source code of Implemented Programs:**

//Nanekar Saurabh Rajesh

#include <stdio.h>

#include <stdlib.h>

int top = 10;

struct node

{

        char ch;

        struct node \*next;

        struct node \*prev;

}  \*stack[11];

typedef struct node node;

void push(node \*str)

{

        if (top <= 0)

        printf("Stack is Full ");

        else

        {

                stack[top] = str;

                top--;

        }

}

node \*pop()

{

        node \*exp;

        if (top >= 10)

                printf("Stack is Empty ");

        else

                exp = stack[++top];

        return exp;

}

void convert(char exp[])

{

        node \*op1,  \*op2;

        node \*temp;

        int i;

        for (i=0;exp[i]!='\0';i++)

        if (exp[i] >= 'a'&& exp[i] <= 'z'|| exp[i] >= 'A' && exp[i] <= 'Z')

        {

                temp = (node\*)malloc(sizeof(node));

                temp->ch = exp[i];

                temp->next = NULL;

                temp->prev = NULL;

                push(temp);

        }

        else if (exp[i] == '+' || exp[i] == '-' || exp[i] == '\*' || exp[i] == '/' ||

exp[i] == '^')

        {

                op1 = pop();

                op2 = pop();

                temp = (node\*)malloc(sizeof(node));

                temp->ch = exp[i];

                temp->next = op1;

                temp->prev = op2;

                push(temp);

        }

}

void display(node \*temp)

{

        if (temp != NULL)

        {

                display(temp->prev);

                printf("%c", temp->ch);

                display(temp->next);

        }

}

void main()

{

        char exp[50];

        printf("Enter the postfix expression :");

        scanf("%s", exp);

        convert(exp);

        printf("\nThe Equivalant Infix expression is:");

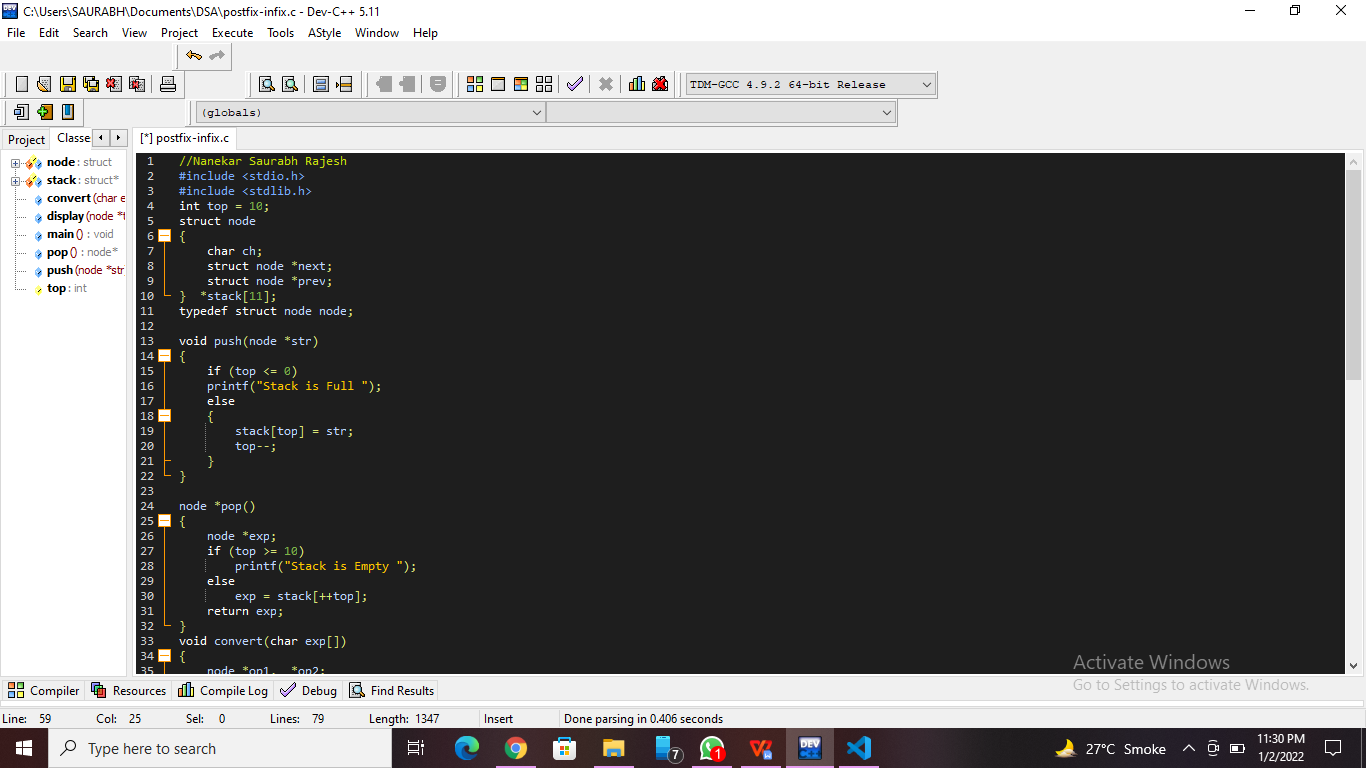
        display(pop());

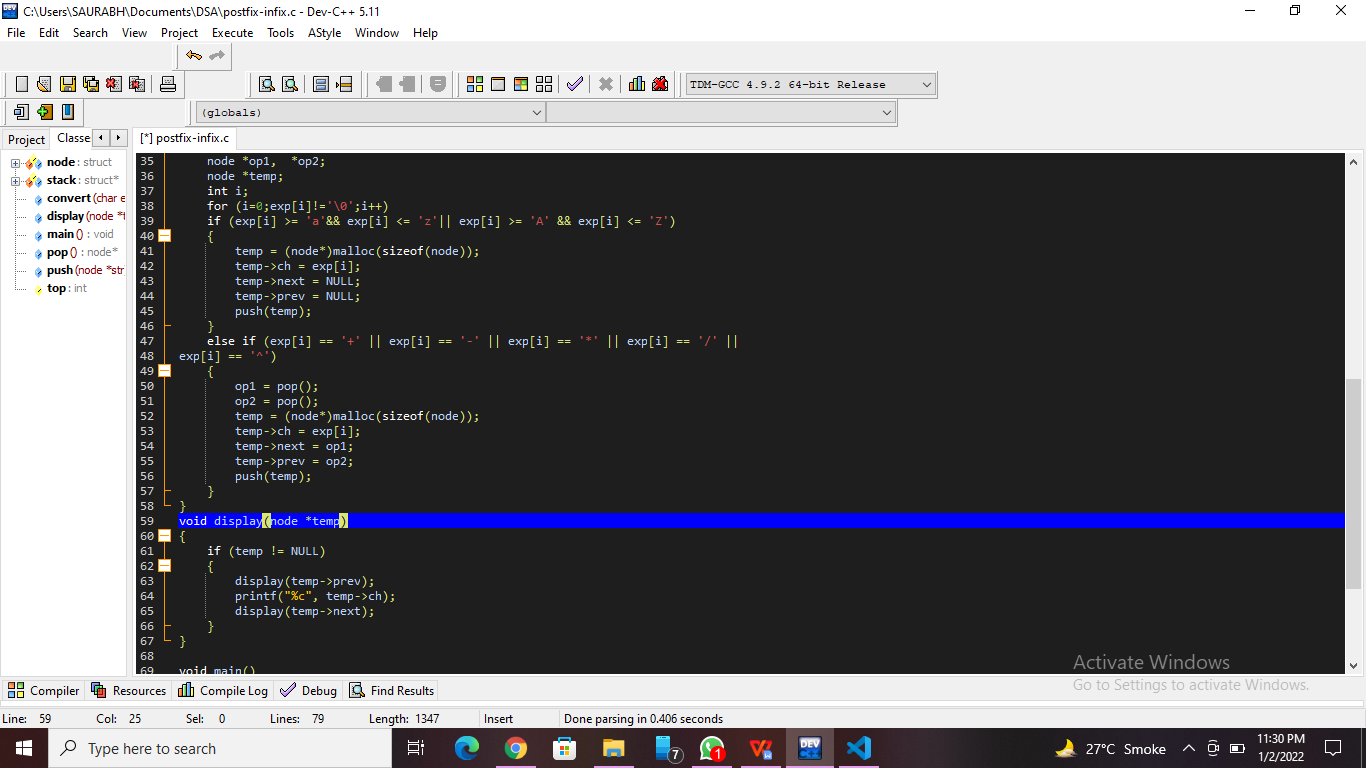
        printf("\n\n");

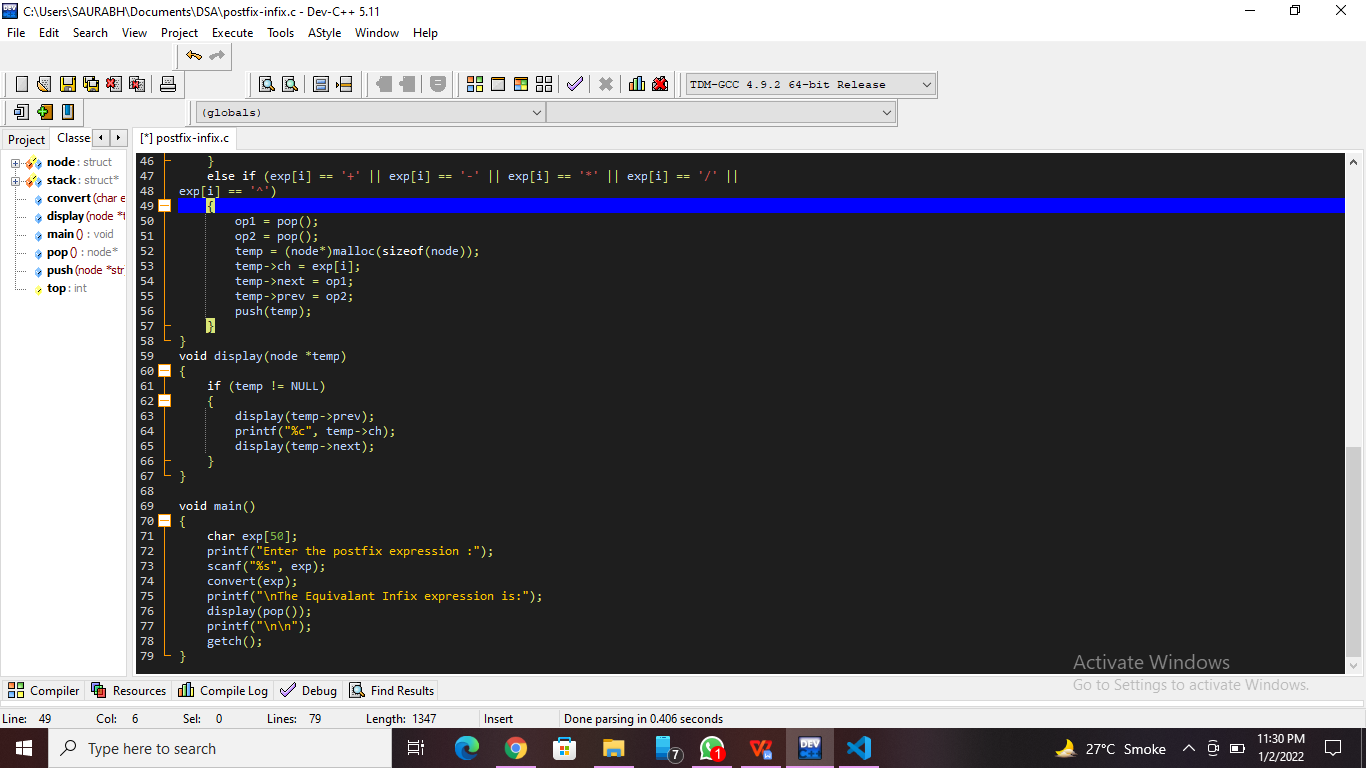
        getch();

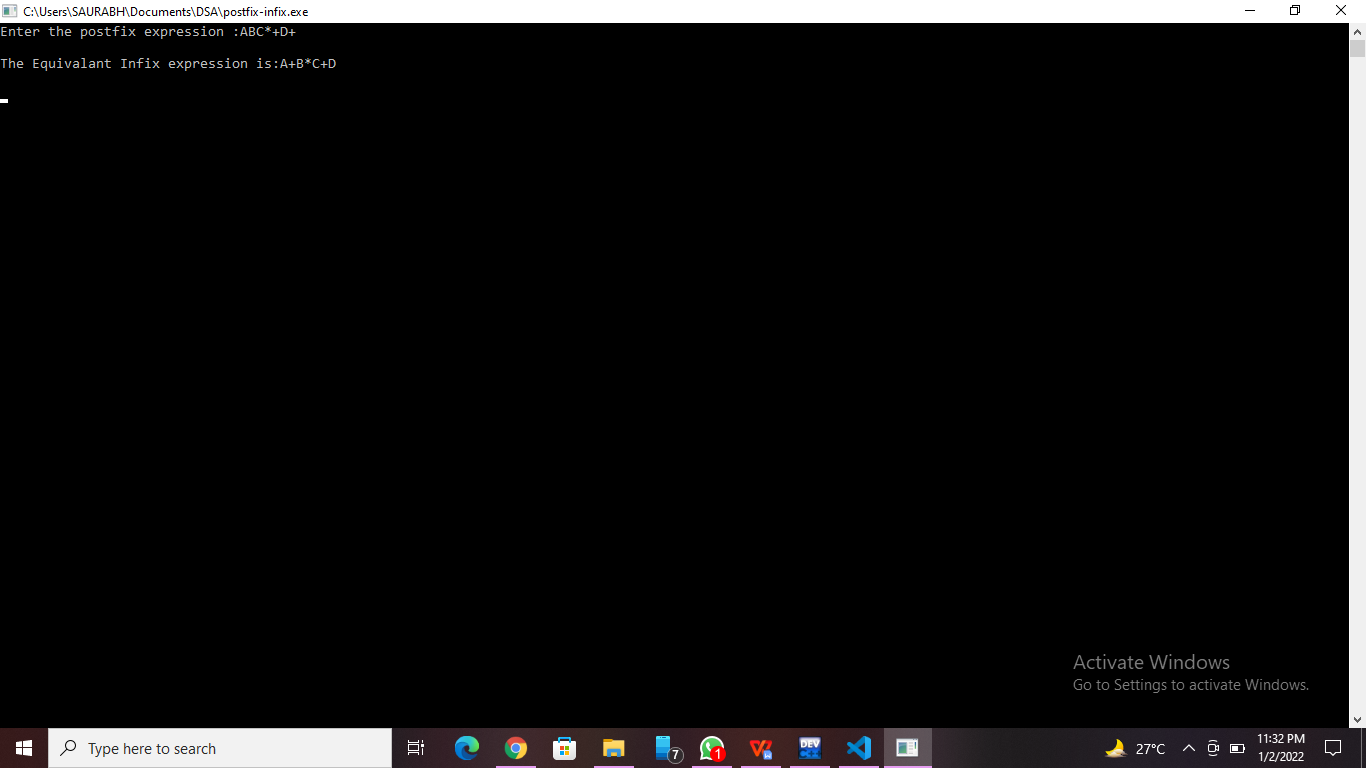
}

**Screenshots of Output:**

****

****

****

****

**Practice Program:**Write a C program to reverse a string using Stack.

//Nanekar Saurabh Rajesh\_20141212\_I1

#include <stdio.h>

#include <string.h>

#define max 100

int top,stack[max];

void push(char x)

{

      if(top == max-1){

          printf("overflow");

      }  else {

          stack[++top]=x;

      }

}

void pop()

{

      printf("%c",stack[top--]);

}

int main()

{

   char str[]="SaurabhRajeshNanekar";

   int len = strlen(str);

   int i;

   for(i=0;i<len;i++)

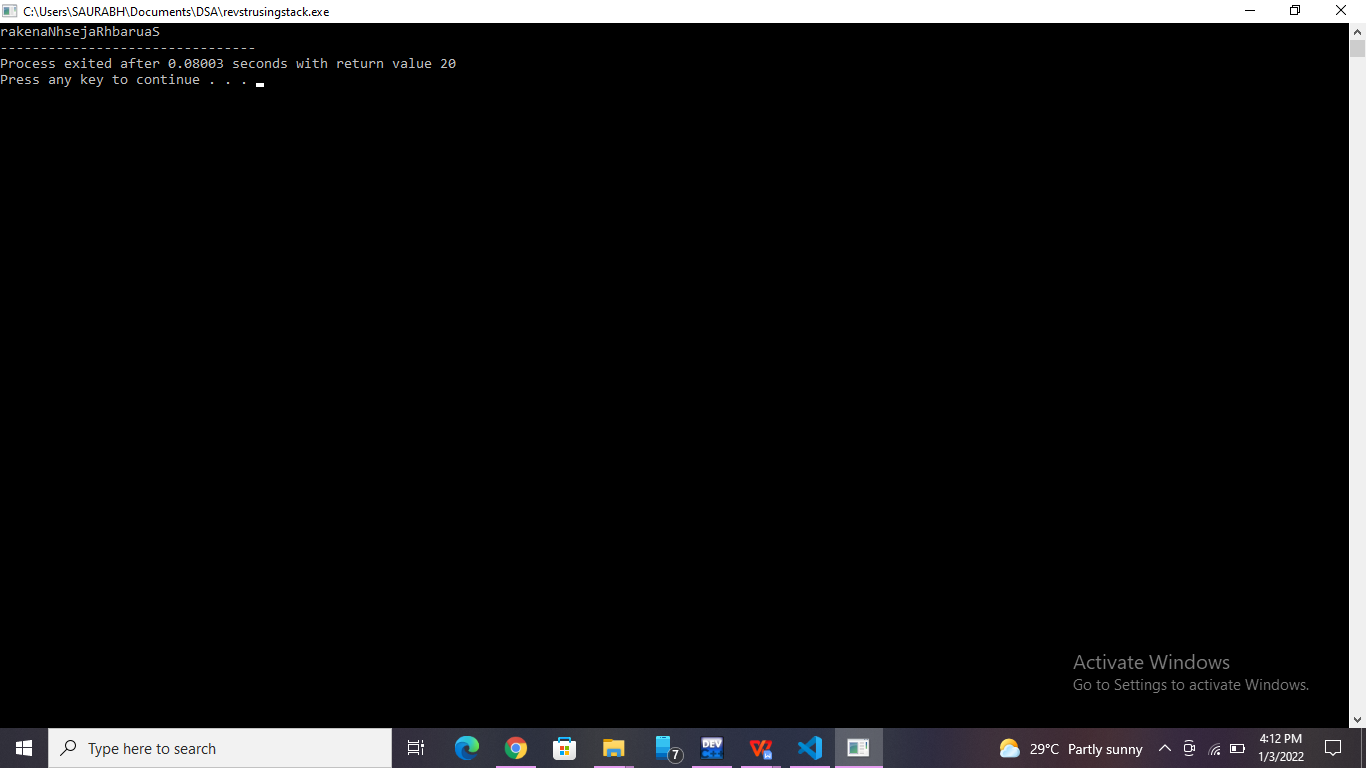
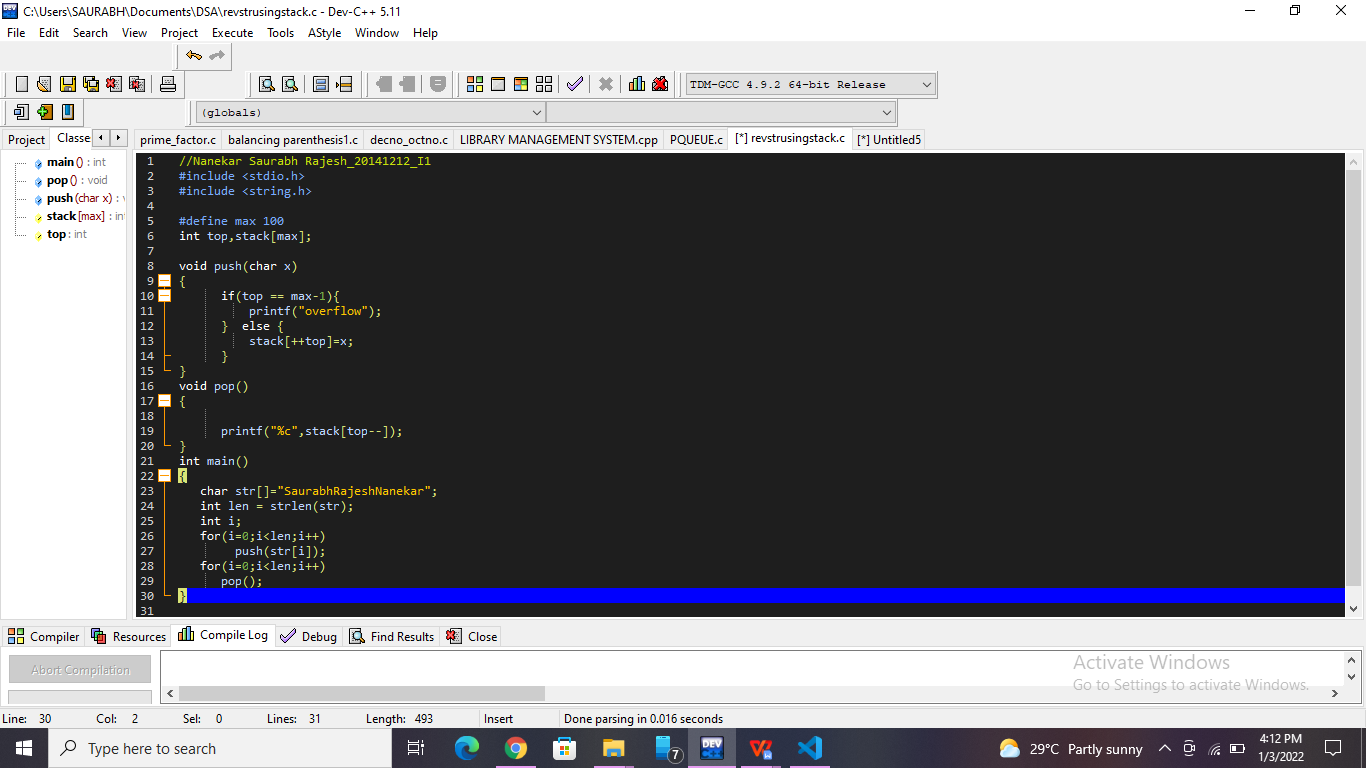
        push(str[i]);

   for(i=0;i<len;i++)

      pop();

}

**Screenshots of Practice Program:-**

****

**List of sample questions for oral examination:**

1. What are the 6 applications of stack?
2. What are the rules of infix to postfix?
3. [Which of the following data structure is used to convert the infix expression into prefix or postfix?](https://www.google.co.in/search?sxsrf=AOaemvKDoJyjPt_iVQGa0mYznrP0kIFRSQ:1637980542181&q=Which+of+the+following+data+structure+is+used+to+convert+the+infix+expression+into+prefix+or+postfix%3F&sa=X&ved=2ahUKEwiXyO2Cwbf0AhVb7nMBHSGqDL8Qzmd6BAgIEAU)
4. Which infix expression is equivalent to a postfix expression ABC ++?
5. How do you solve an infix expression?

**Conclusion:**

We know the various method to convert mathematical expressions for simplicity of program.