**Assignment 3**

**Name :-** Dhruv Thakkar

Branch :- CS

Division :- A

Roll no. :- 69

GR Number :- 12111512

**Problem Statement**

Write a program to convert a given Postfix expression into its

equivalent Prefix expression and evaluate it using stack

**Solution:**

#include<stdio.h>

#include<string.h>

#include<math.h>

#include<stdlib.h>

#include <stdbool.h>

char \*pop();

char postfix[50];

char stack[50][50];

void push(char \*str);

int isempty();

void postfix\_to\_prefix();

int top;

bool isOperator(char a);

int main()

{

        top = -1;

        printf("Enter postfix expression : ");

        gets(postfix);

        postfix\_to\_prefix();

        return 0;

}

void postfix\_to\_prefix()

{

        unsigned int i;

        char operand1[50], operand2[50];

        char symbol;

        char temp[2];

        char strin[50];

        for(i=0;i<strlen(postfix);i++)

        {

                symbol=postfix[i];

                temp[0]=symbol;

                temp[1]='\0';

                if(isOperator(symbol)){

                        if(isempty()){

                                printf("invalid prefix statement!!");

                        }

                        else{

                                strcpy(operand1,pop());

                                strcpy(operand2,pop());

                                strcpy(strin,temp);

                                strcat(strin,operand2);

                                strcat(strin,operand1);

                                push(strin);

                        }

                }

                else push(temp);

        }

        printf("\nPrefix Expression :: ");

        puts(stack[0]);

}

void push(char \*str)

{

        if(top > 50)

        {

                printf("\nStack overflow\n");

                exit(1);

        }

        else

        {

                top=top+1;

                strcpy(stack[top], str);

        }

}

char \*pop()

{

        if(top == -1 )

        {

            printf("\nStack already empty \n");

            exit(2);

        }

        else

            return (stack[top--]);

}

int isempty()

{

    if(top==-1)

        return 1;

    else

        return 0;

}

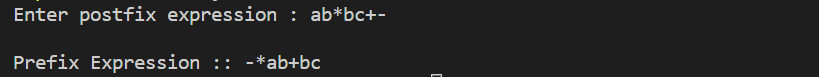
bool isOperator(char a){

    if(a=='+'|| a=='\*'|| a=='-'|| a=='/'|| a=='(') return true;

    else return false;

}

**Output:**

****