DBMS LAB2:INFIX TO POSTFIX:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <ctype.h>

#define MAX 100

// Stack implementation

char stack[MAX];

int top = -1;

// Function to push an element to the stack

void push(char x) {

    if (top == MAX - 1) {

        printf("Stack overflow\n");

        return;

    }

    stack[++top] = x;

}

// Function to pop an element from the stack

char pop() {

    if (top == -1) {

        printf("Stack underflow\n");

        return -1;

    }

    return stack[top--];

}

// Function to get precedence of operators

int precedence(char op) {

    if (op == '+' || op == '-')

        return 1;

    if (op == '\*' || op == '/')

        return 2;

    return 0;

}

// Function to convert infix expression to postfix expression

void infixToPostfix(char\* infix, char\* postfix) {

    int i = 0, j = 0;

    char ch;

    while (infix[i] != '\0') {

        ch = infix[i];

        // If the character is an operand, add it to the postfix expression

        if (isalnum(ch)) {

            postfix[j++] = ch;

        }

        // If the character is '(', push it to the stack

        else if (ch == '(') {

            push(ch);

        }

        // If the character is ')', pop and add to postfix until '(' is found

        else if (ch == ')') {

            while (top != -1 && stack[top] != '(') {

                postfix[j++] = pop();

            }

            pop(); // Remove '(' from stack

        }

        // If an operator is encountered

        else {

            while (top != -1 && precedence(stack[top]) >= precedence(ch)) {

                postfix[j++] = pop();

            }

            push(ch);

        }

        i++;

    }

    // Pop all remaining operators from the stack

    while (top != -1) {

        postfix[j++] = pop();

    }

    postfix[j] = '\0'; // Null-terminate the postfix expression

}

int main() {

    char infix[MAX], postfix[MAX];

    printf("Enter infix expression: ");

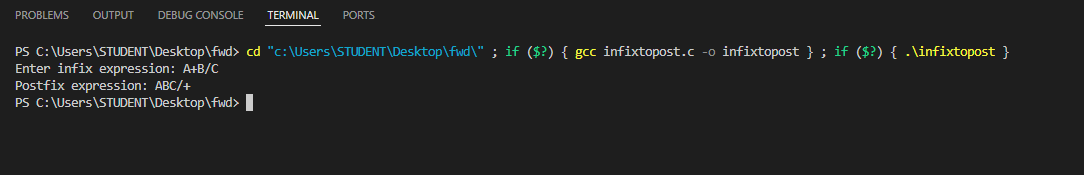
    scanf("%s", infix);

    infixToPostfix(infix, postfix);

    printf("Postfix expression: %s\n", postfix);

    return 0;

}

 OUTPUT: