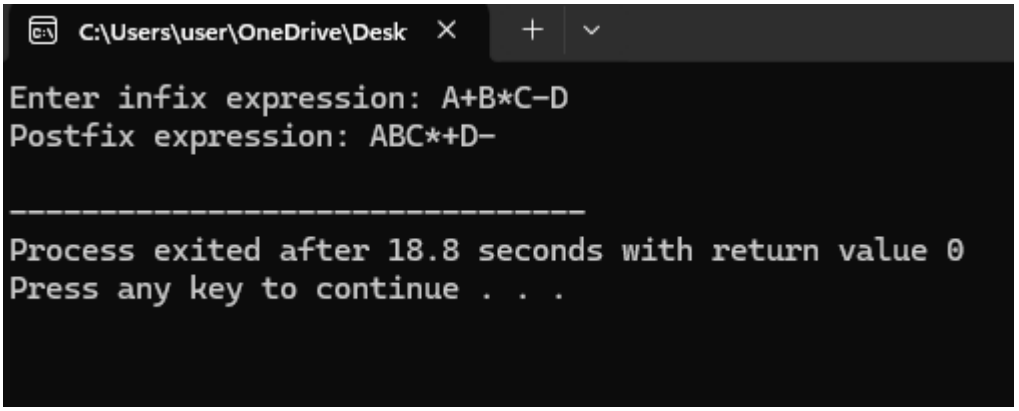
```c
#include <stdio.h>
#include <ctype.h>
#include <string.h>

#define MAX 100

char stack[MAX];
int top = -1;

// Push to stack
void push(char ch) {
    if (top < MAX - 1)
        stack[++top] = ch;
}

// Pop from stack
char pop() {
    if (top >= 0)
        return stack[top--];
    return -1;
}

// Peek top of stack
char peek() {
    if (top >= 0)
        return stack[top];
    return -1;
}

// Check precedence
int precedence(char op) {
    switch (op) {
        case '^': return 3;
        case '*':
        case '/': return 2;
        case '+':
        case '-': return 1;
        default:  return 0;
    }
}

// Convert infix to postfix
void infixToPostfix(char infix[], char postfix[]) {
    int i, k = 0;
    char ch;

    for (i = 0; infix[i] != '\0'; i++) {
        ch = infix[i];
```

```
        if (isalnum(ch)) {
            postfix[k++] = ch; // Operand directly to postfix
        } else if (ch == '(') {
            push(ch);
        } else if (ch == ')') {
            while (peek() != '(') {
                postfix[k++] = pop();
            }
            pop(); // Pop '('
        } else { // Operator
            while (precedence(peek()) >= precedence(ch)) {
                postfix[k++] = pop();
            }
            push(ch);
        }
    }

    // Pop remaining operators
    while (top != -1) {
        postfix[k++] = pop();
    }

    postfix[k] = '\0'; // Null-terminate 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;
}
```


C:\Users\user\OneDrive\Desk
Enter infix expression: A+B*C-D
Postfix expression: ABC*+D-

Process exited after 18.8 seconds with return value 0
Press any key to continue . . .