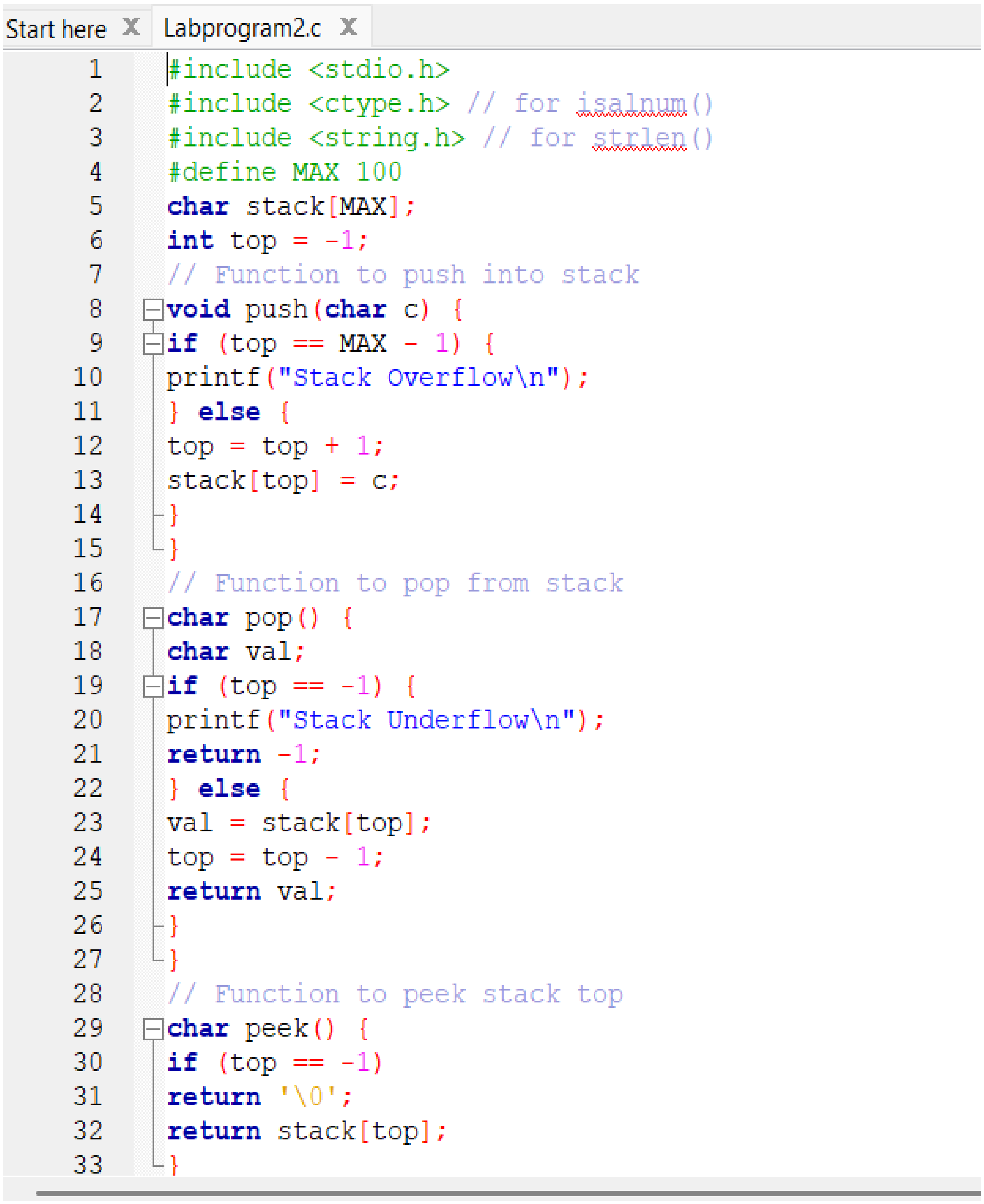
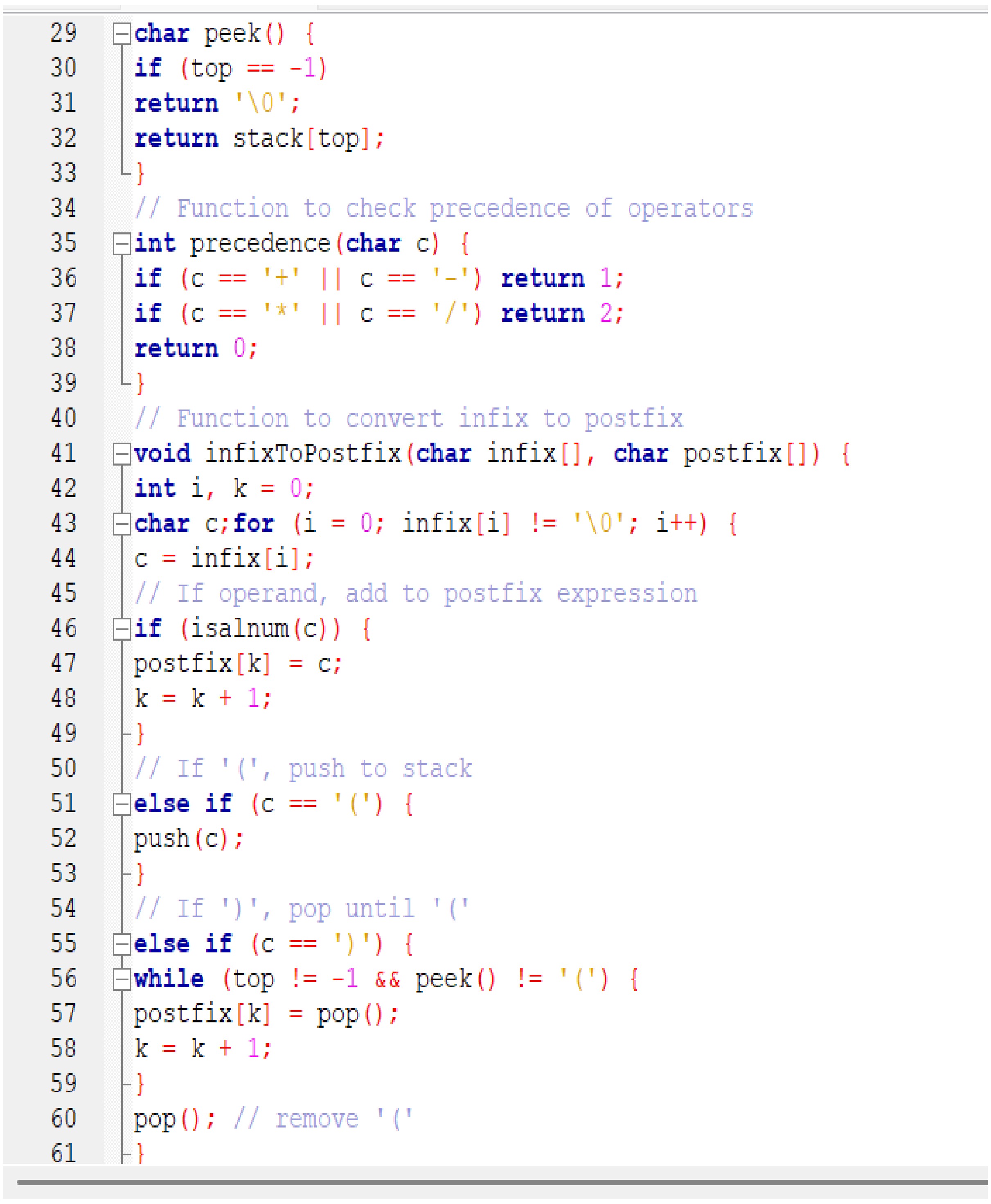
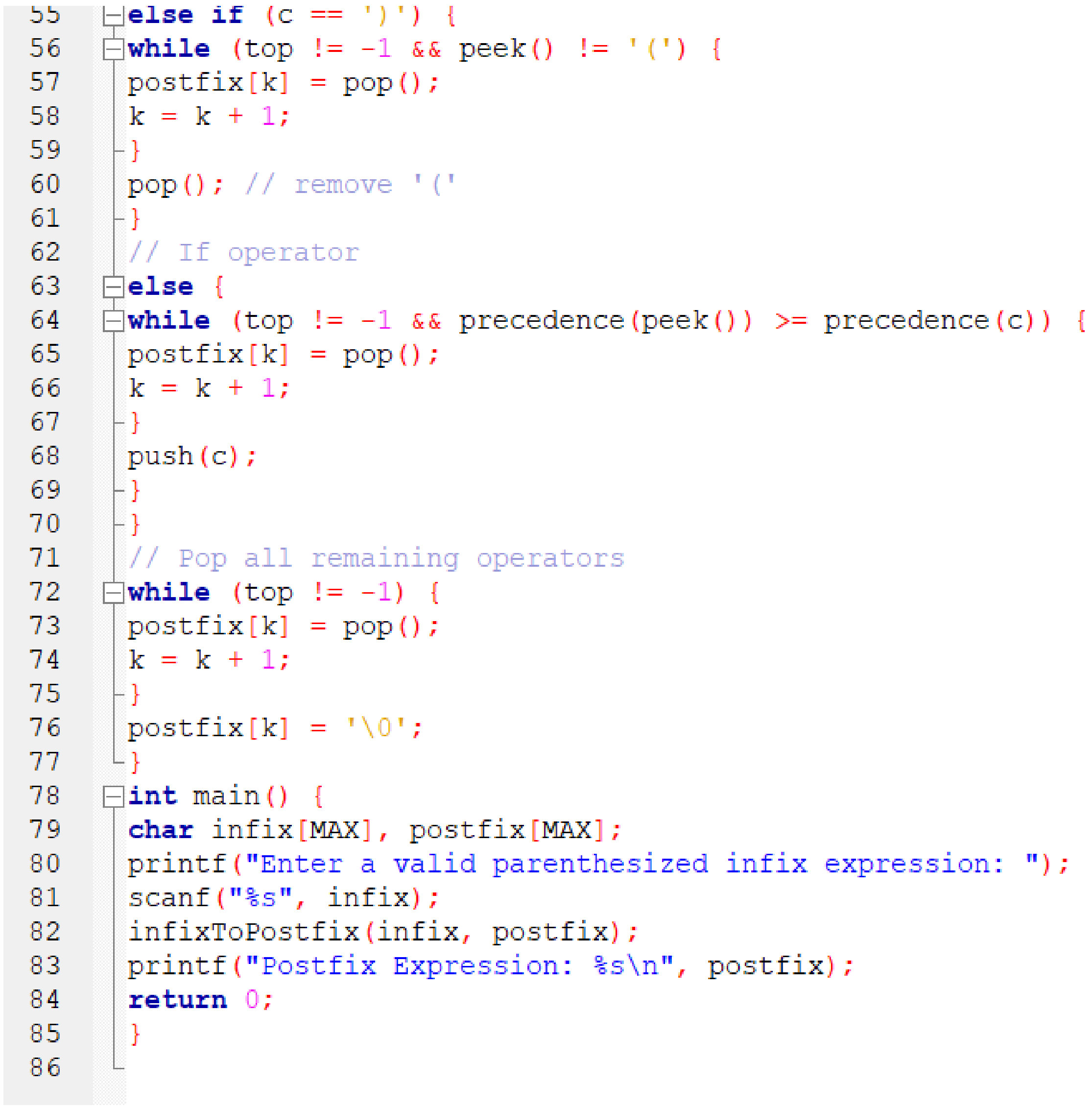
DS LAB PROGRAM 2



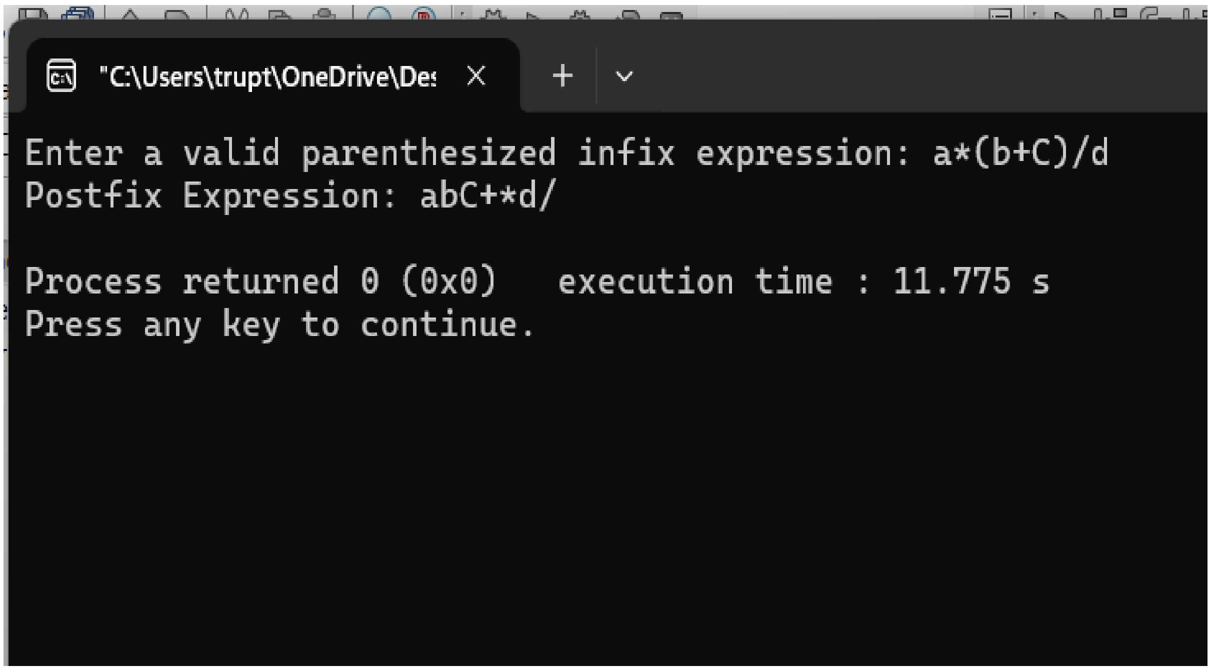
```c
#include <stdio.h>
#include <ctype.h> // for isalnum()
#include <string.h> // for strlen()
#define MAX 100
char stack[MAX];
int top = -1;
// Function to push into stack
void push(char c) {
if (top == MAX - 1) {
printf("Stack Overflow\n");
} else {
top = top + 1;
stack[top] = c;
}
}
// Function to pop from stack
char pop() {
char val;
if (top == -1) {
printf("Stack Underflow\n");
return -1;
} else {
val = stack[top];
top = top - 1;
return val;
}
}
// Function to peek stack top
char peek() {
if (top == -1)
return '\0';
return stack[top];
}
```



```c
char peek() {
if (top == -1)
return '\0';
return stack[top];
}
// Function to check precedence of operators
int precedence(char c) {
if (c == '+' || c == '-') return 1;
if (c == '*' || c == '/') return 2;
return 0;
}
// Function to convert infix to postfix
void infixToPostfix(char infix[], char postfix[]) {
int i, k = 0;
char c;for (i = 0; infix[i] != '\0'; i++) {
c = infix[i];
// If operand, add to postfix expression
if (isalnum(c)) {
postfix[k] = c;
k = k + 1;
}
// If '(', push to stack
else if (c == '(') {
push(c);
}
// If ')', pop until '('
else if (c == ')') {
while (top != -1 && peek() != '(') {
postfix[k] = pop();
k = k + 1;
}
pop(); // remove '('
}
```

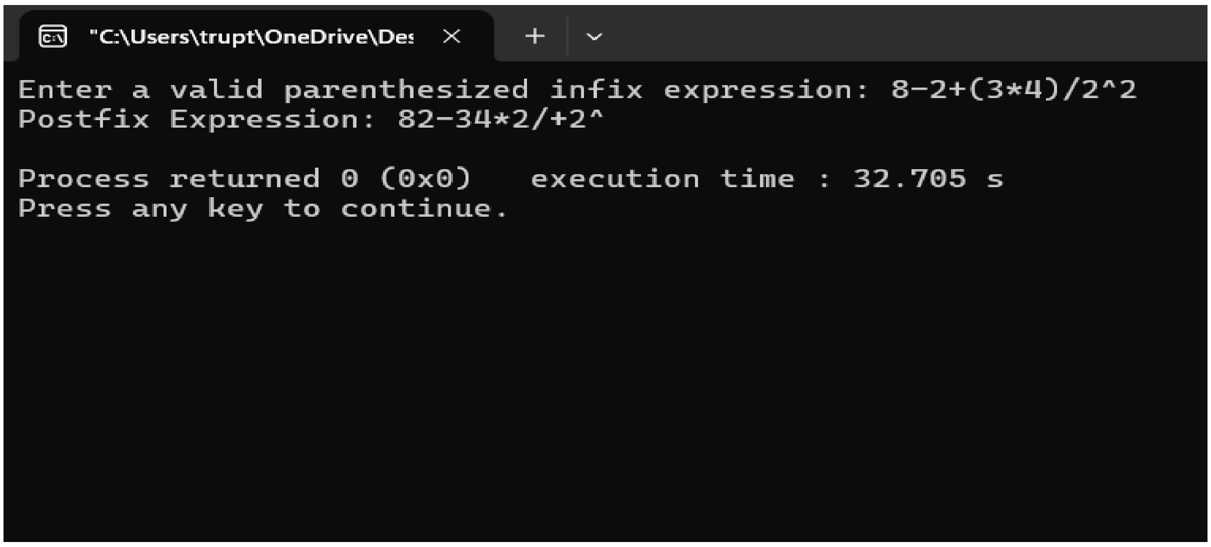


```c
else if (c == ')') {
while (top != -1 && peek() != '(') {
postfix[k] = pop();
k = k + 1;
}
pop(); // remove '('
}
// If operator
else {
while (top != -1 && precedence(peek()) >= precedence(c)) {
postfix[k] = pop();
k = k + 1;
}
push(c);
}
}
// Pop all remaining operators
while (top != -1) {
postfix[k] = pop();
k = k + 1;
}
postfix[k] = '\0';
}
int main() {
char infix[MAX], postfix[MAX];
printf("Enter a valid parenthesized infix expression: ");
scanf("%s", infix);
infixToPostfix(infix, postfix);
printf("Postfix Expression: %s\n", postfix);
return 0;
}
```



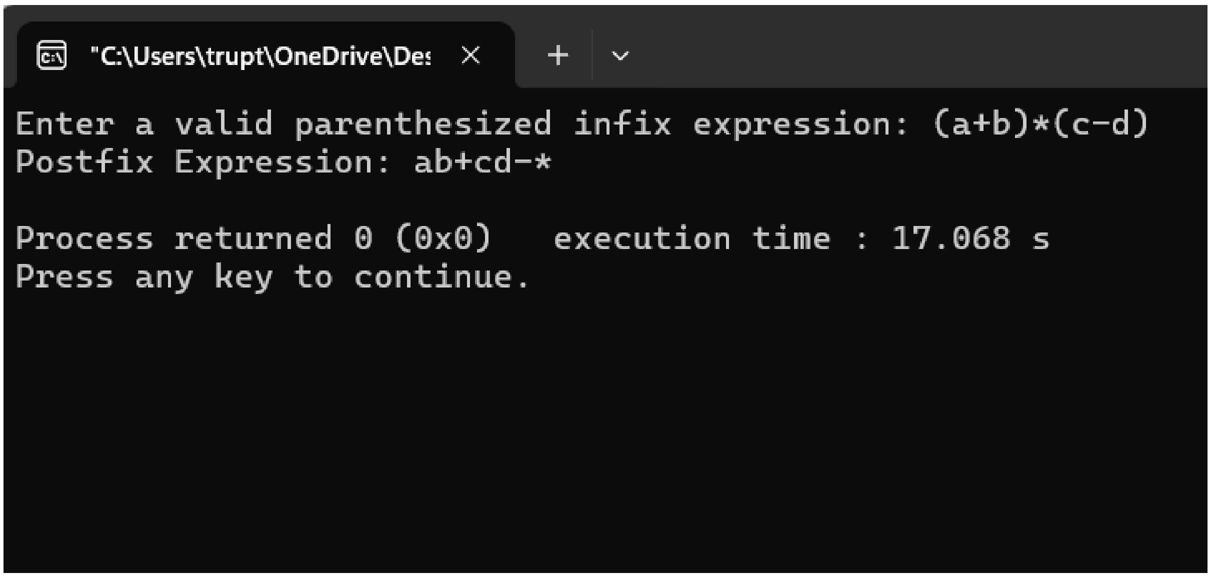
```
Enter a valid parenthesized infix expression: a*(b+C)/d
Postfix Expression: abC+*d/

Process returned 0 (0x0)   execution time : 11.775 s
Press any key to continue.
```



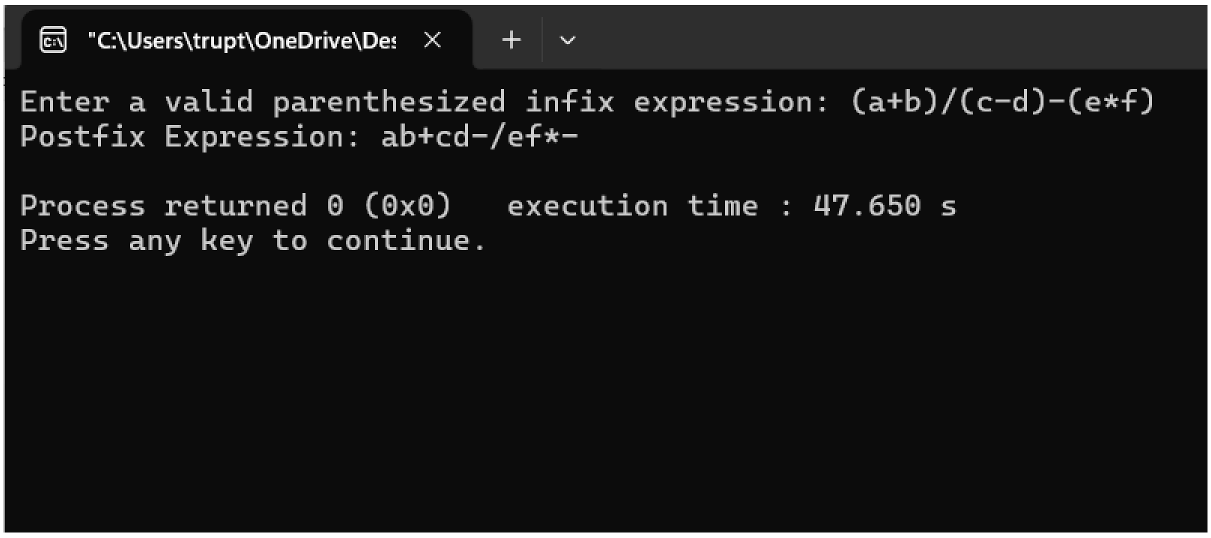
```
Enter a valid parenthesized infix expression: 8-2+(3*4)/2^2
Postfix Expression: 82-34*2/+2^

Process returned 0 (0x0)   execution time : 32.705 s
Press any key to continue.
```



```
Enter a valid parenthesized infix expression: (a+b)*(c-d)
Postfix Expression: ab+cd-*

Process returned 0 (0x0)   execution time : 17.068 s
Press any key to continue.
```



```
Enter a valid parenthesized infix expression: (a+b)/(c-d)-(e*f)
Postfix Expression: ab+cd-/ef*-

Process returned 0 (0x0)   execution time : 47.650 s
Press any key to continue.
```