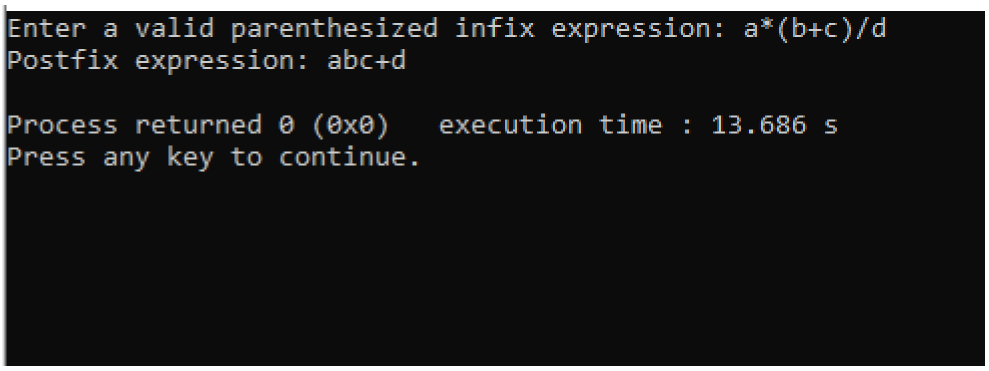
2. WAP to convert a given valid parenthesized infix arithmetic expression to postfix expression. The expression consists of single character operands and the binary operators + (plus), - (minus), * (multiply) and / (divide)

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

#define MAX 50

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

// Push into stack
void push(char ch) {
    stack[++top] = ch;
}

// Pop from stack
char pop() {
    return stack[top--];
}

int main() {
    char infix[MAX], postfix[MAX];
    int i = 0, k = 0;
    char ch;

    printf("Enter a valid parenthesized infix expression: ");
    scanf("%s", infix);

    while ((ch = infix[i++]) != '\0') {

        // If operand, add to postfix
        if (isalnum(ch)) {
            postfix[k++] = ch;
        }
        // If operator, push to stack
        else if (ch == '+' || ch == '-' || ch == '*' || ch == '/') {
            push(ch);
        }
        // If closing parenthesis, pop operator
        else if (ch == ')') {
            postfix[k++] = pop();
        }
        // Ignore opening parenthesis '('
    }

    postfix[k] = '\0';

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

    return 0;
}
```

OUTPUT:



```
Enter a valid parenthesized infix expression: a*(b+c)/d
Postfix expression: abc+d

Process returned 0 (0x0)   execution time : 13.686 s
Press any key to continue.
```