17.

#include <stdio.h>

#include <ctype.h>

int main() {

FILE \*file = fopen("input.txt", "r");

char ch;

int lines = 0, words = 0, characters = 0;

int in\_word = 0;

while ((ch = fgetc(file)) != EOF) {

characters++;

if (ch == '\n') lines++;

if (isspace(ch)) {

if (in\_word) {

words++;

in\_word = 0;

}

} else {

in\_word = 1;

}

}

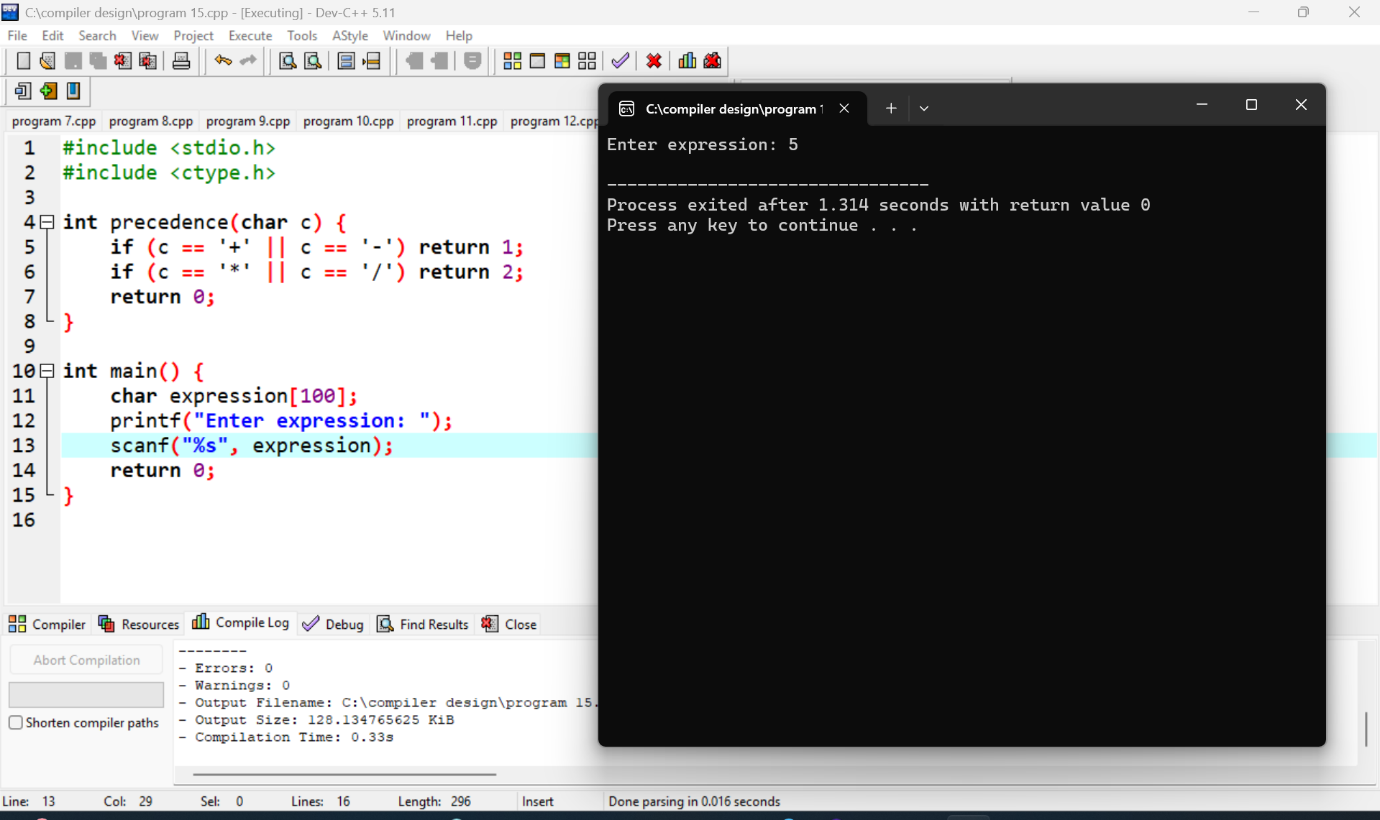
fclose(file);

printf("Lines: %d, Words: %d, Characters: %d\n", lines, words, characters);

return 0;

}

**Output:**

****

18.

#include <stdio.h>

#include <string.h>

int temp\_count = 1;

void generate\_three\_address\_code(char \*op, char \*op1, char \*op2) {

printf("t%d = %s %s %s\n", temp\_count++, op1, op, op2);

}

void optimize\_code() {

// Simple optimization (e.g., eliminating unnecessary temporaries or redundant operations)

printf("Optimization step: Eliminate common subexpressions (if applicable).\n");

}

void generate\_target\_code() {

printf("Target machine code generation: Converting 3-address code to assembly.\n");

}

int main() {

// Example: a simple expression `a + b \* c` turned into intermediate code

generate\_three\_address\_code("\*", "b", "c");

generate\_three\_address\_code("+", "a", "t1");

// Optimization step

optimize\_code();

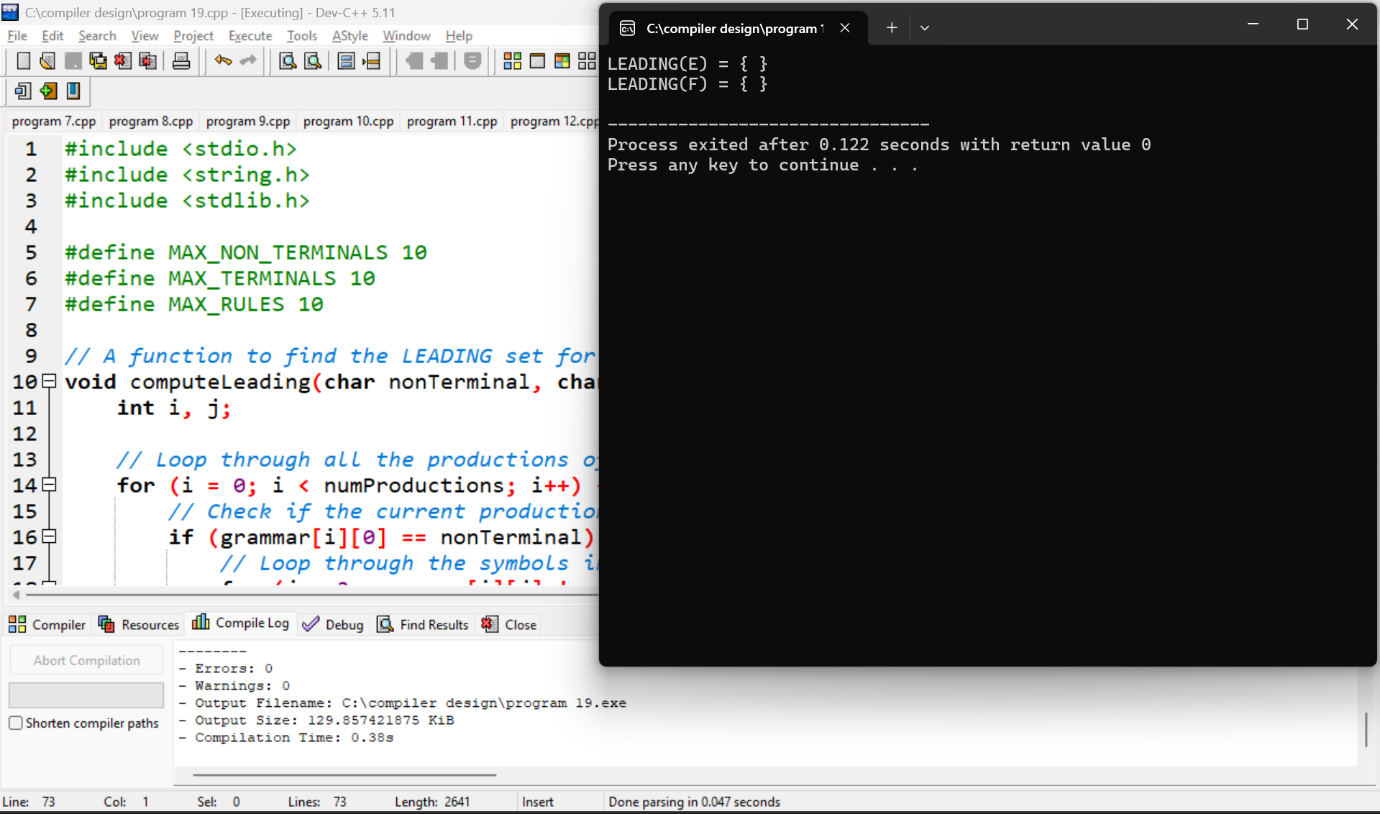
// Final code generation (machine code or assembly)

generate\_target\_code();

return 0;

}

**Output:**

****