**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"**

Інститут **ІКНІ**

Кафедра **ПЗ**

**ЗВІТ**

До лабораторної роботи № 4

**На тему:** *“символьні рядки в С .Ввід та вивід”*

**З дисципліни:** *“Основи програмування”*

**Лектор:**

ст.викл. каф. ПЗ

Муха Т.О.

**Виконав:**

ст. гр. ПЗ-15

Марущак А.С.

**Прийняла:**

асист. каф. ПЗ

Заводовська Н.О.

« \_\_\_\_ » \_\_\_\_\_\_\_\_ 2021 р.

∑= \_\_\_\_ .

Львів – 2021

**Тема роботи:** символьні рядки в С. Ввід та вивід.

**Мета роботи:** здобути практичні навики опрацювання текстової інформації з врахуванням особливостей організації символьних рядків у мові С. Вивчити основні засоби потокового вводу/виводу в С.

**Теоретичні відомості**

Мова С не має спеціального типу для оголошення символьних рядків, а розглядає символьний рядок як особливий вид масиву. Елементи масиву, який називають символьним рядком, мають тип char, його значеннями є коди символів, з яких складається цей рядок (ASCII-коди, якщо заданий компілятор застосовує ASCII-таблицю для кодування символів). Останнім символом рядка повинен бути т. зв. нуль-символ ('\0'), код якого дорівнює 0. 3 кожним символьним рядком пов'язується вказівник на початок даного рядка. У всьому іншому – символьні рядки повністю зберігають властивості масивів.

Рядкові константи (літерали) в мові С записуються як послідовність довільних символів взятих у подвійні лапки: "...". В оперативній пам'яті їм виділяється ділянка, обсяг якої на один байт більший за кількість символів у рядку. В цей додатковий байт автоматично записується нуль-символ (‘\0’), який надалі слугуватиме ознакою кінця рядка. Приклад:

char \*pst = "Hello world!";

Для збереження в пам’яті записаного рядка компілятор виділить 13 байтів, з них 12 байтів для символів і останній для '\0' . Адресу початку рядка отримає вказівник pst. Символьні рядки також можуть оголошуватися як звичайні масиви:

char імя\_символьного\_рядка [кількість\_символів];

Оголошений нижче масив str призначений для збереження символьного рядка:

char str[150];

Також існує бібліотека для роботи з рядками(string.h). Деякі з її функцій:

char\* strcpy(sr, s);

Копіює рядок s (з '\0' включно) за адресою, заданою параметром sr. Повертає значення sr - адресу скопійованого рядка.

char\* strcat(sr, s); Додає рядок s (з '\0' включно) у кінець рядка sr. Повертає значення sr – адресу доповненого рядка.

int strcmp(s1, s2); Послідовно порівнює символи рядків s1 і s2 як дані з типом unsigned char. Повертає ціле число, значення якого <0, якщо s1<s2; 0, якщо s1 == s2; > 0, якщо s1 > s2.

char\* strncpy(sr, s, n); Аналог strcpy(), але з s копіюється не більше, ніж n початкових символів; якщо скопійована група символів не закінчується '\0', то нуль-символ у sr не заноситься.

char\* strncat (sr, s, n); Аналог strcat(), але додає до sr тільки n початкових символів з s; у кінець об'єднаного рядка заноситься '\0'. char\*

strncmp(s1, s2, n); Аналог strcmp(), але порівнює тільки n початкових символів рядків s1 та s2. Якщо якийсь із рядків коротший за n, то порівняння припиняється з досягненням ' \0'.

unsigned strlen(s); Повертає довжину рядка s у символах (' \ 0' не враховується).

char\* strchr(s, sym); Перевіряє, чи символ sym входить у рядок s. Повертає вказівник на перше входження sym у s або NULL, якщо sym не зустрічається в рядку s.

char\* strrchr(s, sym); Аналог strchr(). Повертає вказівник на останнє входження символа sym у рядок s або NULL, якщо sym не зустрічається в рядку s.

char\* strstr(s1, s2); Перевіряє, чи рядок s2 входить як підрядок у s1. Повертає вказівник на перший символ рядка s2 у s1 або NULL, якщо рядок s2 не зустрічається у рядку s1.

char\* strtok(sr, s); Виділяє в рядку sr лексеми, обмежені символами з рядка s. Повертає вказівник на виділену лексему або NULL.

char\* strdup(s); Копіює рядок s (з ' \0 ' включно) в динамічну пам'ять, попередньо виділивши там ділянку потрібної довжини. Повертає адресу рядка в динамічній пам'яті.

Також необхідні функції для роботи з файлами:

FILE\* fopen(char const\* file\_name, char const\* fmode);

Параметрами функції є вказівниками на символьні рядки. file\_name – ім’я файлу, fmode– режим обміну даними. За умови успішного відкриття потоку створюється спеціальна структура типу FILE і функція повертає її адресу. Якщо ж потік відкрити не вдалось, функція повертає NULL.

Для закриття потоків використовується функція int fclose(FILE\* fp); fp – вказівник на потік, який треба закрити. При успішному використанні функції вона повертає значення 0.

Для форматованого виводу даних у конкретний файл використовується функція:

int fprintf(FILE\* fp, char\* format, … ); Параметр fp є вказівником потоку виводу. Решта параметрів такі ж, як і в printf().

Для форматованого вводу даних з конкретного файлу використовується функція:

int fscanf(FILE\* fp, char\* format, … ); Параметр fp є вказівником потоку вводу. Решта параметрів такі ж, як і в scanf().

**ЛАБОРАТОРНЕ ЗАВДАННЯ**

1. Ознайомитися з теоретичним матеріалом викладеним вище в даній інструкції і виконати приклади програм.

2. Одержати індивідуальне завдання.

3. Розробити алгоритм розв’язання індивідуального завдання і подати його у вигляді блоксхеми.

4. Скласти програму на мові С у відповідності з розробленим алгоритмом.

5. Виконати обчислення по програмі.

6. Самостійно ознайомитися з функціями fread(), fwrite(), fseek(), ftell().

7. Скласти програму на мові С, яка обчислюватиме значення виразу з лабораторної роботи №1 Додаток 1 “Обчислення заданих арифметичних виразів”. Значення параметрів X, Y, Z прочитати із заданого бінарного файлу (вводиться користувачем з клавіатури). Результат записати у файл, теж вказаний користувачем. Формат файлу (бінарний чи текстовий) також задається користувачем. При цьому робота програми повинна бути запротокольована. Виконання основних подій, починаючи зі старту програми, має бути відображене у log-файлі з часовими мітками (розміщення вибрати самостійно). Старі записи у log-файлі мають зберігатися. Перелік подій для логування:  старт програми  відкриття файлу з параметрами  обчислення значення виразу  запис обчисленого значення у вихідний файл  завершення програми

Приклад log-файлу:

[2016.09.01 8:30.000] Program started.

[2016.09.01 8:30.011] Parameters file “d:\data\input.bin” opened. X=3, Y=1, Z=2.4.

[2016.09.01 8:30.012] Expression calculated. Result = 4.56.

[2016.09.01 8:30.014] Output file “d:\data\output.txt” saved.

[2016.09.01 8:30.015] Program ended. Для отримання поточного часу можна скористатися функцією time() із заголовного файлу time.h.

11.Підготувати та здати звіт про виконання лабораторної роботи.

**Індивідуальне завдання**

№1

Вважаючи, що введене речення з клавіатури складається з довільної кількості слів, між якими є довільна кількість пробілів, і закінчується речення крапкою, написати програму для розв’язання завдання:

22. Ввести речення і ключове слово. Визначити, чи є у введеному реченні слово, яке складається з тих самих літер, що й задане ключове слово.

Код програми:

#include <string.h>

#include <stdio.h>

#include <ctype.h>

#define MAX\_LIMIT 50

int main()

{

    char sentence[MAX\_LIMIT], keyword[MAX\_LIMIT];

    printf("Enter sentence: ");

    fgets(sentence, MAX\_LIMIT, stdin);

    sentence[strlen(sentence)-1] = '\0';

    sentence[0] = tolower(sentence[0]);

    char\* delptr = strchr(sentence, '.');

    if(delptr) \*delptr = '\0';

    printf("Enter keyword: ");

    fgets(keyword, MAX\_LIMIT, stdin);

    keyword[strlen(keyword)-1] = '\0';

    char key\_chars[MAX\_LIMIT] = { 0 };

    int key\_point = 0;

    for(char\* p = keyword; \*p != '\0'; ++p){

        if(strchr(key\_chars, \*p) == NULL){

            key\_chars[key\_point] = \*p;

            ++key\_point;

        }

    }

    char\* word = strtok(sentence, " ");

    int match = 1;

    while(word != NULL){

        char word\_chars[MAX\_LIMIT] = { 0 };

        int word\_point = 0;

        for(char\* p = word; \*p != '\0'; ++p){

            if(strchr(word\_chars, \*p) == NULL){

                word\_chars[word\_point] = \*p;

                ++word\_point;

            }

        }

        if(strlen(word\_chars) == strlen(key\_chars)){

            match = 1;

            for(char \*c = key\_chars; \*c != '\0'; c++){

                if(strchr(word\_chars, \*c) == NULL){

                    match = 0;

                    break;

                }

            }

            if(match){

                printf("Match! Word: %s", word);

                break;

            }

        }

        word = strtok(NULL, " ");

    }

    if(!match){

        printf("No match");

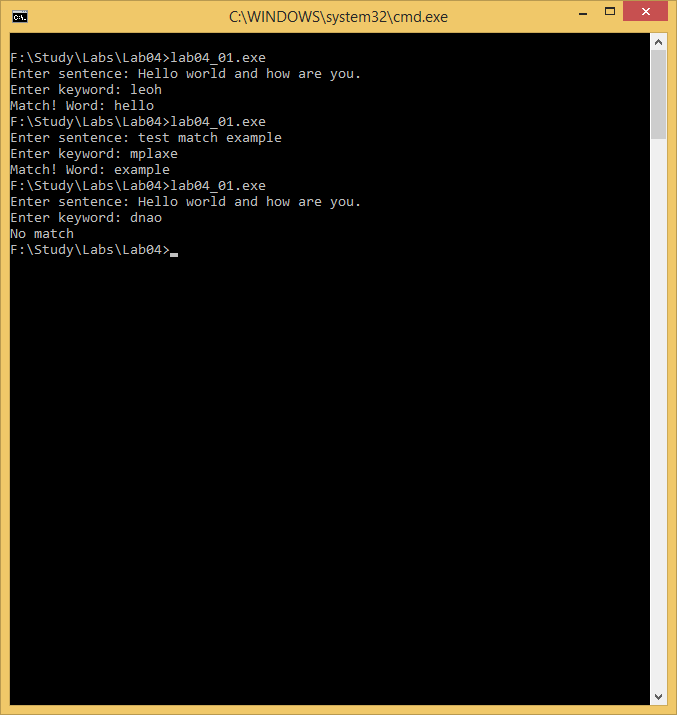
    }

    return 0;

}

**Протокол роботи:**

Программа отримує 2 стрічки, намагається знайти слово в реченні, що складається з тих самих літер, що і ключове слово, і по можливості виводить його:



В перших двох випадках слова були знайдені, а в останньому ні.

**Висновок:** ми написали програму, яка приймає 2 стрічки і шукає слово, що складається з тих самих літер, що і ключове слово. Це допомогло нам навчитись зчитувати стрічки і проводити над ними операції за допомогою функцій стандартної бібліотеки.

№2

Код програми:

#include <stdio.h>

#include <math.h>

#include <string.h>

#include <time.h>

#define MAX\_FILE\_NAME\_LENGTH 50

#define LOG\_FILE "log.txt"

int main(){

    time\_t curr\_time;

    FILE \*logFile = fopen(LOG\_FILE, "a");

    char \*errorMessage = "";

    time(&curr\_time);

    char\* time\_str = ctime(&curr\_time);

    \*(strchr(time\_str, '\n')) = '\0';

    fprintf(logFile, "[%s] Program started.\n", time\_str);

    double X, Y, Z;

    char inputFileName[MAX\_FILE\_NAME\_LENGTH];

    printf("Input file: ");

    scanf("%s", inputFileName);

    getchar();

    FILE \*input = fopen(inputFileName, "rb");

    if(input == NULL){

        errorMessage = "The input file doesn't exist.";

        printf("%s\n",errorMessage);

        time(&curr\_time);

        time\_str = ctime(&curr\_time);

        \*(strchr(time\_str, '\n')) = '\0';

        fprintf(logFile, "[%s] An error ocurred: %s. Aborting program...\n", time\_str, errorMessage);

        return 0;

    }

    time(&curr\_time);

    time\_str = ctime(&curr\_time);

    \*(strchr(time\_str, '\n')) = '\0';

    fprintf(logFile, "[%s] Parameters file \"%s\" opened.\n", time\_str, inputFileName);

    fread(&X, sizeof(double), 1, input);

    fread(&Y, sizeof(double), 1, input);

    fread(&Z, sizeof(double), 1, input);

    fclose(input);

    time(&curr\_time);

    time\_str = ctime(&curr\_time);

    \*(strchr(time\_str, '\n')) = '\0';

    fprintf(logFile, "[%s] Got values of variables: X = %lf; Y = %lf; Z = %lf;\n", time\_str, X, Y, Z);

    if(fabs(Z) > 1){

        errorMessage = "Incorrect values in input file!";

        printf("%s\n",errorMessage);

        time(&curr\_time);

        time\_str = ctime(&curr\_time);

        \*(strchr(time\_str, '\n')) = '\0';

        fprintf(logFile, "[%s] An error ocurred: %s. Aborting program...\n", time\_str, errorMessage);

        fclose(logFile);

        return 0;

    }

    double b = pow(asin(Z), 2) + fabs(X + Y);

    time(&curr\_time);

    time\_str = ctime(&curr\_time);

    \*(strchr(time\_str, '\n')) = '\0';

    fprintf(logFile, "[%s] Expression calculated: b = %lf;\n", time\_str, b);

    char outputFileName[MAX\_FILE\_NAME\_LENGTH];

    printf("Output file: ");

    scanf("%s", outputFileName);

    getchar();

    printf("Output format (t|b): ");

    char choice;

    scanf("%c", &choice);

    if(choice == 't' || choice == 'T'){

        FILE \*output = fopen(outputFileName, "w");

        time(&curr\_time);

        time\_str = ctime(&curr\_time);

        \*(strchr(time\_str, '\n')) = '\0';

        fprintf(logFile, "[%s] Text output file \"%s\" opened.\n", time\_str, outputFileName);

        fprintf(output, "%lf", b);

        fclose(output);

        time(&curr\_time);

        time\_str = ctime(&curr\_time);

        \*(strchr(time\_str, '\n')) = '\0';

        fprintf(logFile, "[%s] Text output file \"%s\" saved.\n", time\_str, outputFileName);

    }

    else if(choice == 'b' || choice == 'B'){

        FILE \*output = fopen(outputFileName, "wb");

        time(&curr\_time);

        time\_str = ctime(&curr\_time);

        \*(strchr(time\_str, '\n')) = '\0';

        fprintf(logFile, "[%s] Binary output file \"%s\" opened.\n", time\_str, outputFileName);

        fwrite(&b, sizeof(double), 1, output);

        fclose(output);

        time(&curr\_time);

        time\_str = ctime(&curr\_time);

        \*(strchr(time\_str, '\n')) = '\0';

        fprintf(logFile, "[%s] binary output file \"%s\" saved.\n", time\_str, outputFileName);

    }

    else{

        errorMessage = "Incorrect choice!";

        printf("%s\n",errorMessage);

        time(&curr\_time);

        time\_str = ctime(&curr\_time);

        \*(strchr(time\_str, '\n')) = '\0';

        fprintf(logFile, "[%s] An error ocurred: %s. Aborting program...\n", time\_str, errorMessage);

        fclose(logFile);

        return 0;

    }

    time(&curr\_time);

    time\_str = ctime(&curr\_time);

    \*(strchr(time\_str, '\n')) = '\0';

    fprintf(logFile, "[%s] Program finished.\n", time\_str);

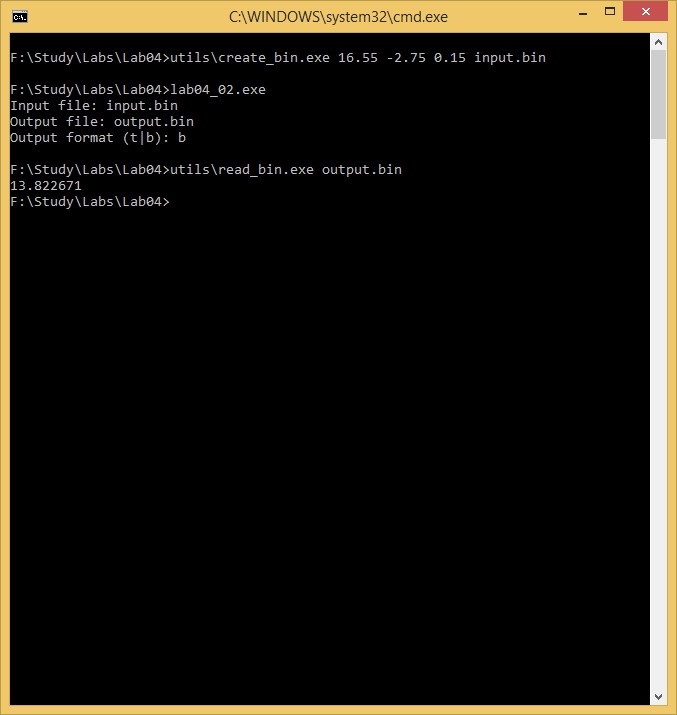
    fclose(logFile);

    return 0;

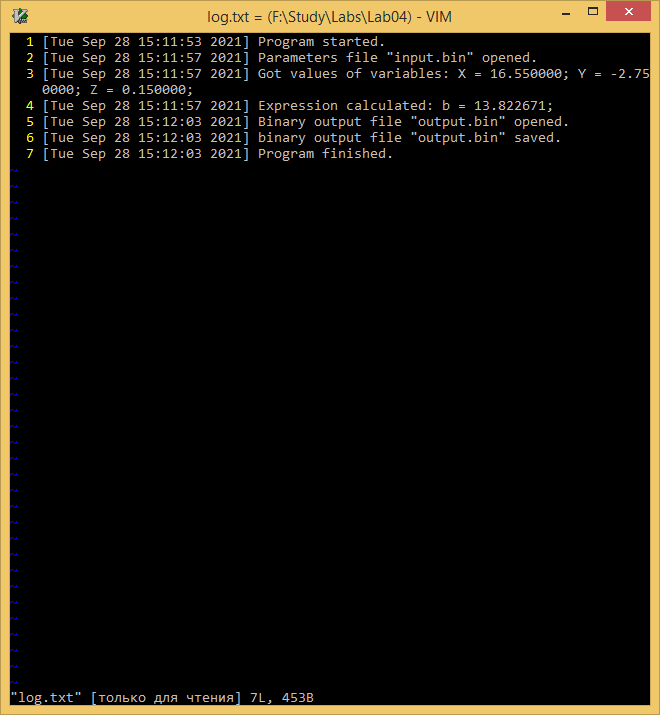
}

**Протокол роботи:**

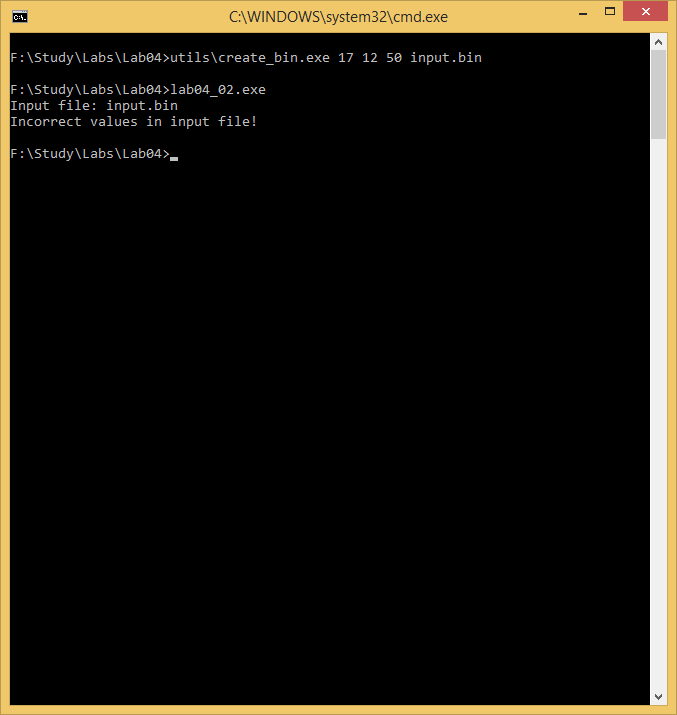
Програма зчитує назву вхідного файлу, відкриває його, зчитує дані, записує відповідь у названий файл у вибраному форматі:



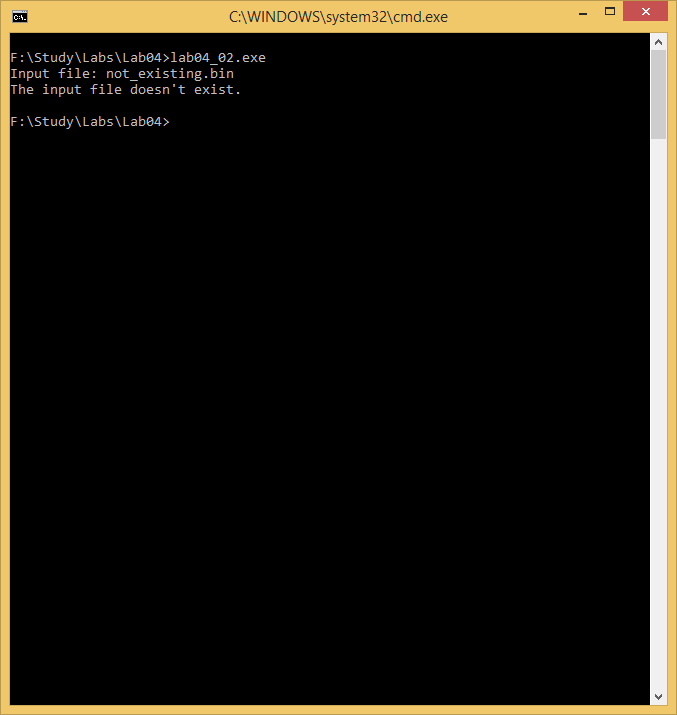
Файл логу:



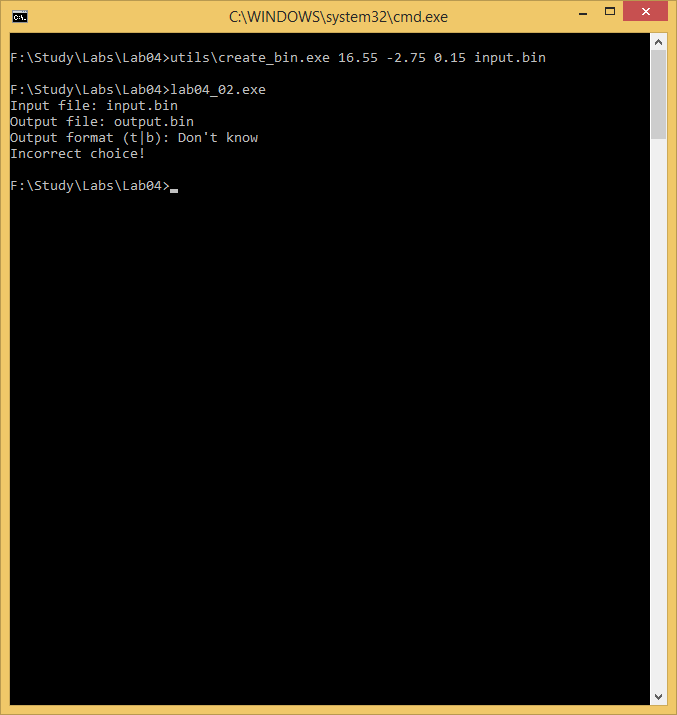
Також програма має змогу обробляти різні помилки:



1. Порушення області визначення функції.

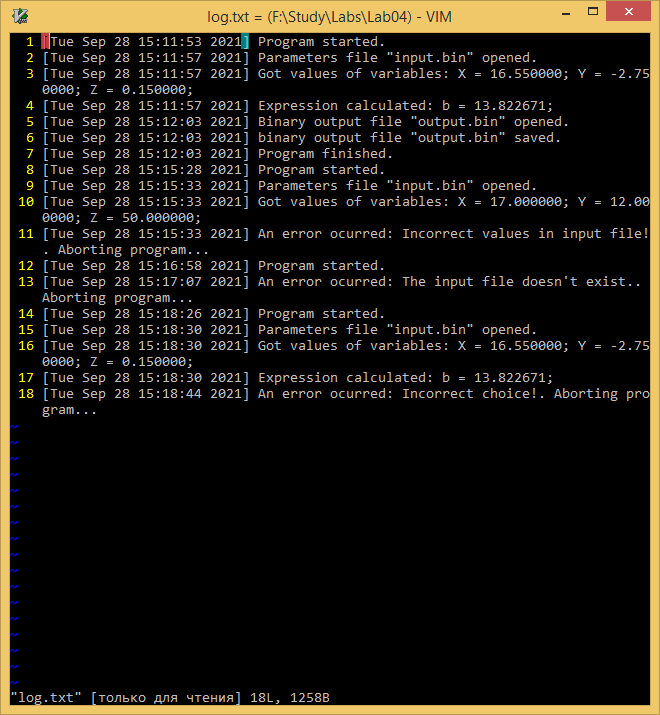


1. Коли вхідний файл не існує.



1. Невідомий вибір формату.

Файл логу:



**Висновок:** ми написали програму, що зчитує дані з файлу і записує результат в файл, а також проводить додаткове протоколювання. Це допомогло нам навчитися працювати з файловою системою за допомогою функцій мови С.