**Міністерство Освіти І НАУКИ України**

**Національний університет "Львівська політехніка"**

Інститут **КНІТ**

Кафедра **ПЗ**

### ЗВІТ

До лабораторної роботи № 4

**На тему:** *“Робота з символьними рядками в С”*

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

**Лектор:**

ст.викл. каф. ПЗ

Муха Т.О.

**Виконав:**

ст. гр. ПЗ-14

Губик А. С.

**Прийняв:**

ст.викл. каф ПЗ

Кутельмах Р. К.

« \_\_\_\_ » \_\_\_\_\_\_\_\_ 2022 р.

∑= \_\_\_\_ .

Львів – 2022

**Тема роботи:** робота з символьними рядками в С.

**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.

**Індивідуальне завдання**

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

Додаток 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:

У введеному реченні визначити середню довжину слова.

**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. 2.1 Оголошення та ініціалізація символьних рядків 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]; У str можна записати довільний символьний рядок, довжина якого не перевищує 149 символів, оскільки останнім записується нуль-символ – для нього треба обов'язково зарезервувати один байт. Слід також пам'ятати, що перевищення встановленої в оголошенні кількості символів не контролюється компілятором і може призвести до небезпечних помилок у роботі програми. В оголошеннях символьні рядки, як і масиви символів, можна ініціалізувати. Розглянемо декілька характерних прикладів:

char ml[20] = {'a', 'b', 'с', 'd', 'e', 'f'};

char m2[20] = {'a', 'b', 'с', 'd', 'e', ' f', '\0'};

char m3[20] = "abcdef";

char m4[ ] = "abcdef";

Символьні рядки m1, m2 і m3 оголошено однаково – як масиви з 20 елементів, що мають тип char, але ініціалізацію їх виконано різними способами. Початкові шість елементів масиву m1 заповнено послідовністю літер, проте без ' \0' у кінці, тому цей масив не буде повноправним символьним рядком, з ним можна буде працювати тільки як із звичайним масивом символів. У масив m2 записано таку ж послідовність літер, а після неї – нуль-символ. Фактично в m2 занесено рядок символів "abcdef" . Такий же рядок записано в масив m3 , тобто результати ініціалізації m2 та m3 збігаються (очевидно, що ініціалізація m3 є простішою у записі). Незаповнені елементи масивів ml, m2 та m3 містять"сміття" (за умови, що масиви оголошено як локальні, а в разі глобальних чи статичних масивів усі вільні елементи заповнюються нулями). Надлишкові елементи можна використовувати надалі для доповнення і розширення відповідних рядків. В оголошенні масиву m4 не вказано граничну кількість символів, тому розмірність цього масиву (символьного рядка) встановлюється за кількістю елементівініціалізаторів. Для наведеного прикладу розмірність m4 становитиме 7 символів: 6 перших байтів масиву заповнюються кодами літер, а сьомий – кінцевий нуль-символом. Хоча масиви m3 та m4 проініціалізовані константними рядками, елементи цих масивів можна змінювати так само, як елементи масивів m1 та m2. 2.2. Звертання до елементів символьних рядків Процеси опрацювання символьних рядків базуються на двох основних властивостях рядків: 1) ім'я символьного рядка є константним вказівником на його перший символ; 2) кінець рядка задається нуль-символом ' \0'. Для звертання до символів рядка застосовують як індексну, так і вказівникову систему доступу до елементів масиву

Файлом вважається іменована сукупність даних, розташованих на зовнішньому носії, а також термінальні пристрої (клавіатура, принтер …). Для уніфікації процесів файлового обміну даними у функціях високого рівня використовують поняття потоку. Потік (stream) – послідовність байтів, що надходять від певного логічного пристрою (файлу) або передаються у цей файл (пристрій). Для узагальнення обміну даними у процесах вводу/виводу здійснюється проміжна буферизація даних. Для кожного відкритого файлу в оперативній пам’яті створюється буфер обміну заданої ємності. Перед введенням даних з конкретного файлу або записом даних у файл, необхідно створити потік, пов’язаний з цим фізичним файлом. Створення потоку реалізує функція відкриття файлу, прототип якої оголошено так:

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. Основні режими відкриття файла (другий параметр fmode функції fopen() ):

"r" – (від англ. read). Файл відкривається тільки для читання (повинен існувати);

"w" - (від англ. write). Файл відкривається тільки для запису (якщо існує – його вміст знищується);

"a" – (від англ. append). Файл відкривається тільки для доповнення;

"r+" - файл відкривається для читання з можливістю запису в нього;

"w+" – файл відкривається для запису з можливістю читання з нього;

"a+" - файл відкривається для доповнення з можливістю читання.

Додатково в параметрі fmode можна задавати текстовий (t) чи бінарний (b) режим відкриття потоку, за замовчуванням встановлюється текстовий режим. Для закриття потоків використовується функція int fclose(FILE\* fp); fp – вказівник на потік, який треба закрити. При успішному використанні функції вона повертає значення 0. Стандартні потоки На початку виконання кожної С-програми відкриваються стандартні потоки:

stdin – потік введення, який пов'язується з клавіатурою;

stdout – потік виведення даних на екран (командний рядок);

stderr – потік повідомлень про помилки.

Стандартні потоки можна перескеровувати (перепризначати), пов’язувати їх із заданим файлом чи пристроєм. Перескерування потоків виконує функція FILE \*freopen (char \*fname, char \*fmode, FILE \*fp); Функція пов’язує потік fp з файлом fname. Режим доступу до даних задає параметр fmode . Значення та форми завдання параметрів fname і fmode такі ж, як для функції fopen(). За умови успішного виконання функція повертає вказівник на створений потік, а в разі помилки – NULL.

**Код програми**

Назва файлу: **eq2.0.c**

#include <stdio.h>

#include <time.h>

#include <string.h>

#define DATE\_LENGTH 25

#define PATH\_LENGTH 100

int main(){

time\_t CurrentTime;

char \*date;

char InputFile[PATH\_LENGTH], OutputFile[PATH\_LENGTH];

FILE \*io, \*log;

float Ti, x;

log = fopen("log.txt", "w");

CurrentTime = time(0);

date = ctime(&CurrentTime);

date[DATE\_LENGTH - 1] = '\0';

fprintf(log, "[ %s %lf ] ", date, (double)clock() / CLOCKS\_PER\_SEC);

fprintf(log, "Program started \n");

printf("Enter the name of input file: ");

fgets(InputFile, PATH\_LENGTH, stdin);

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

CurrentTime = time(0);

date = ctime(&CurrentTime);

date[DATE\_LENGTH - 1] = '\0';

fprintf(log, "[ %s %lf ] ", date, (double)clock() / CLOCKS\_PER\_SEC);

fprintf(log, "Name of input file was entered \n");

io = fopen(InputFile, "r");

if (!io){

CurrentTime = time(0);

date = ctime(&CurrentTime);

date[DATE\_LENGTH - 1] = '\0';

fprintf(log, "[ %s %lf ] ", date, (double)clock() / CLOCKS\_PER\_SEC);

fprintf(log, "Can\'t open \"%s\"\n", InputFile);

return 1;

}

CurrentTime = time(0);

date = ctime(&CurrentTime);

date[DATE\_LENGTH - 1] = '\0';

fprintf(log, "[ %s %lf ] ", date, (double)clock() / CLOCKS\_PER\_SEC);

fprintf(log, "\"%s\" was opened \n", InputFile);

fscanf(io , "%f", &x);

CurrentTime = time(0);

date = ctime(&CurrentTime);

date[DATE\_LENGTH - 1] = '\0';

fprintf(log, "[ %s %lf ] ", date, (double)clock() / CLOCKS\_PER\_SEC);

fprintf(log, "X is now = %f \n", x);

fclose(io);

CurrentTime = time(0);

date = ctime(&CurrentTime);

date[DATE\_LENGTH - 1] = '\0';

fprintf(log, "[ %s %lf ] ", date, (double)clock() / CLOCKS\_PER\_SEC);

fprintf(log, "\"%s\" was closed \n", InputFile);

printf("Enter the of a output file: ");

fgets(OutputFile, PATH\_LENGTH, stdin);

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

CurrentTime = time(0);

date = ctime(&CurrentTime);

date[DATE\_LENGTH - 1] = '\0';

fprintf(log, "[ %s %lf ] ", date, (double)clock() / CLOCKS\_PER\_SEC);

fprintf(log, "Name of output file was entered \n");

io = fopen(OutputFile, "w");

CurrentTime = time(0);

date = ctime(&CurrentTime);

date[DATE\_LENGTH - 1] = '\0';

fprintf(log, "[ %s %lf ] ", date, (double)clock() / CLOCKS\_PER\_SEC);

fprintf(log, "\"%s\" was opened \n", OutputFile);

Ti = x / (1 - x\*x /(3 - x\*x / (5 - x\*x / 7)));

CurrentTime = time(0);

date = ctime(&CurrentTime);

date[DATE\_LENGTH - 1] = '\0';

fprintf(log, "[ %s %lf ] ", date, (double)clock() / CLOCKS\_PER\_SEC);

fprintf(log, "Ti is now = %f \n", Ti);

fprintf(io, "Result is %f \n", Ti);

CurrentTime = time(0);

date = ctime(&CurrentTime);

date[DATE\_LENGTH - 1] = '\0';

fprintf(log, "[ %s %lf ] ", date, (double)clock() / CLOCKS\_PER\_SEC);

fprintf(log, "\"%s\" was written \n", OutputFile);

fclose(io);

CurrentTime = time(0);

date = ctime(&CurrentTime);

date[DATE\_LENGTH - 1] = '\0';

fprintf(log, "[ %s %lf ] ", date, (double)clock() / CLOCKS\_PER\_SEC);

fprintf(log, "\"%s\" was closed \n", OutputFile);

fclose(log);

return 0;

}

**Додаток 1.**

Назва файлу: **awgwrd.c**

#include <stdio.h>

int main(){

char str[256];

int WordLength = 0, WordNumber = 0, isWord = 0;

float AverageLength = 0;

printf("Enter a sentence: ");

scanf("%[^\n]s", str);

for(int i = 0; str[i]; ++i){

if ((str[i] >= 'a' && str[i] <= 'z') ||

(str[i] >= 'A' && str[i] <= 'Z') ||

(str[i] >= '0' && str[i] <= '9') ||

str[i] == '\''){

if (str[i] != '\'')

WordLength++;

isWord = 1;

}else{

AverageLength += WordLength;

WordLength = 0;

if (isWord){

WordNumber++;

isWord = 0;

}

}

}

AverageLength += WordLength;

WordLength = 0;

if (isWord)

WordNumber++;

AverageLength /= WordNumber;

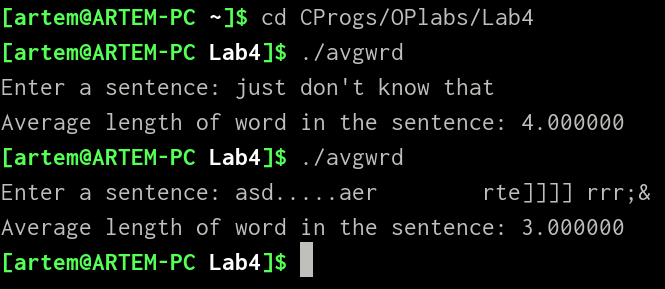
printf("Average length of word in the sentence: %f \n", AverageLength);

return 0;

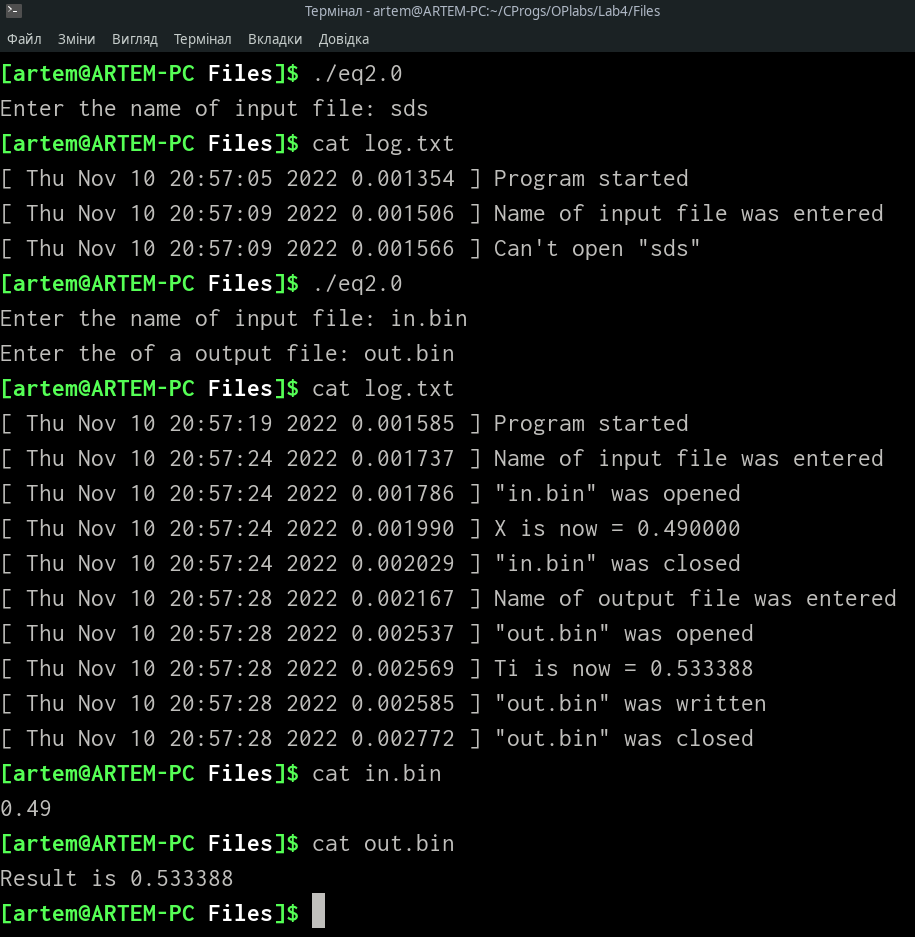
}

**Протокол роботи**

**Додаток 1: avgwrd.c**



**eq2.0.c :**



**Висновки**

Опрацювання стрічок в С дуже схоже на опрацювання масивів. Робота з файлами дає нам змогу зробити програму гнучкішою в плані вводу/виводу.