НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Кафедра «ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ ТА ПРОГРАМУВАННЯ»

«Програмування ч.1»

*Звіт з лабораторної роботи №14*

*Тема: «Взаємодія з файлами»*

Виконав:

ст. гр. КІТ-120А

Старовойтов Н.А.

Перевірив:

Челак В.В.

Харків – 2021

***Мета:*** Отримати навички взаємодії з файлами та ії використанням при розв’язуванні задач.

***Індивідуальне завдання***

Робота на оцінку “відмінно”.

**1)** Програму, яка була розроблена у попередній лабораторній роботі (робота з строками), змінити так, щоб:

• початкові дані вводилися з файлу;

• видача результуючих даних проводилася і в консоль, і у файл.

• ім’я вхідного та вихідного фалу було отримано від користувача;

• при запиті даних, користувач отримав повідомлення, що від нього очікують.

Завдання №5: "Текст - це програма на мові С. Визначити, скільки в ньому операторів циклу".

**2)** Вивести структуру файлів та каталогів, як це робить утиліта Linux tree. Виклик функції system заборонено.

**3)** Визначити об’єм запитаного каталогу. Результат нормалізувати. Формат розміру:

• не більше ніж 3 знаки до коми;

• не більше ніж 2 знаки після коми;

• єдиний випадок, коли в чисельний частині числа може бути 0 - коли

розмір дорівнює 0 байт.

**Опис програми**

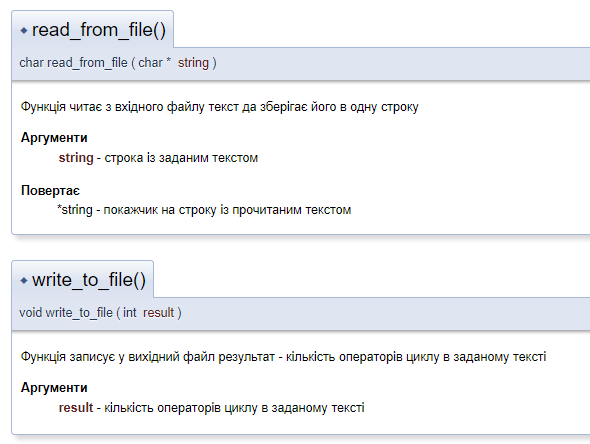


Рисунок 1 — Функції read\_from\_file та write\_to\_file

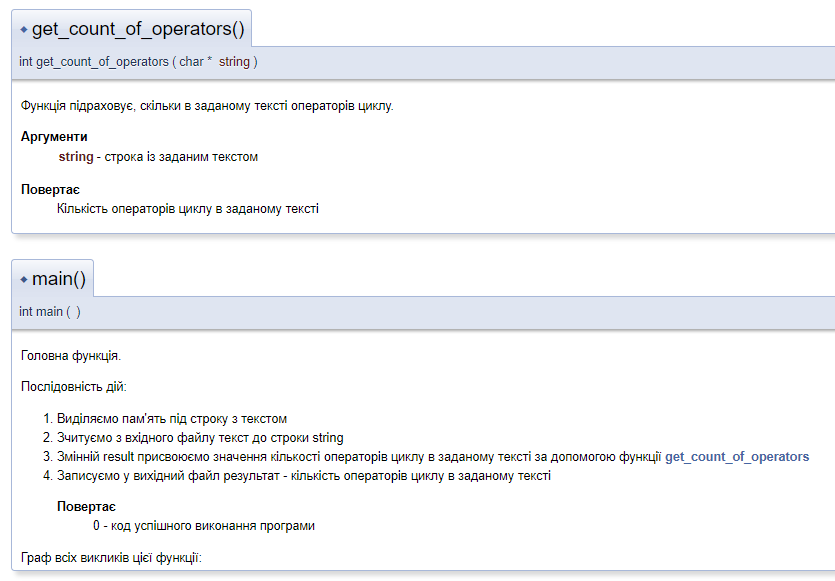


Рисунок 2 — Функції get\_count\_of\_operators та main

**Схеми алгоритмів функцій**

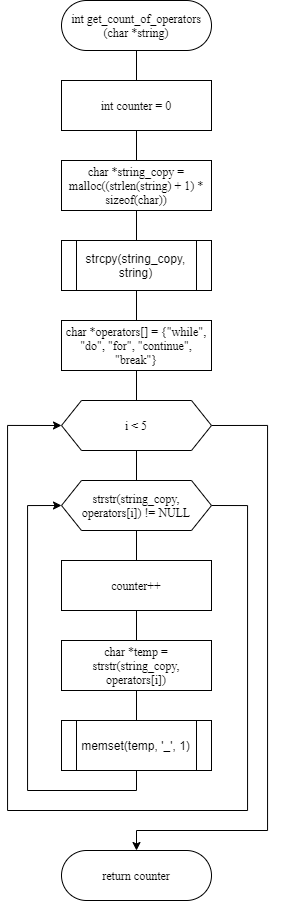
****

Рисунок 3 — Блок-схема функції get\_count\_of\_operator

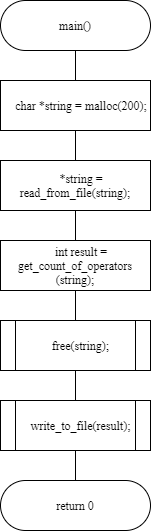


Рисунок 4 — Блок-схема функції main завдання 1

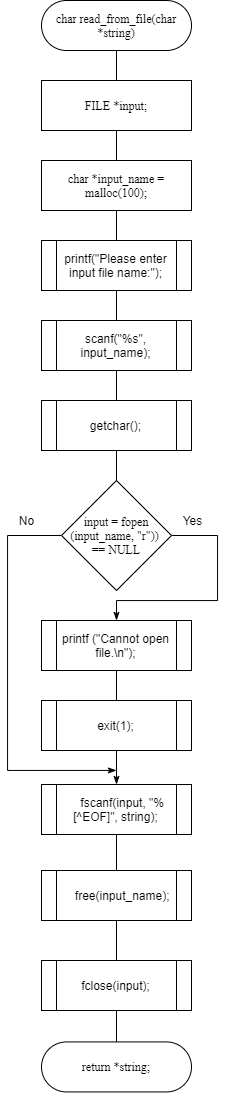


Рисунок 5 — Блок-схема функції read\_from\_file

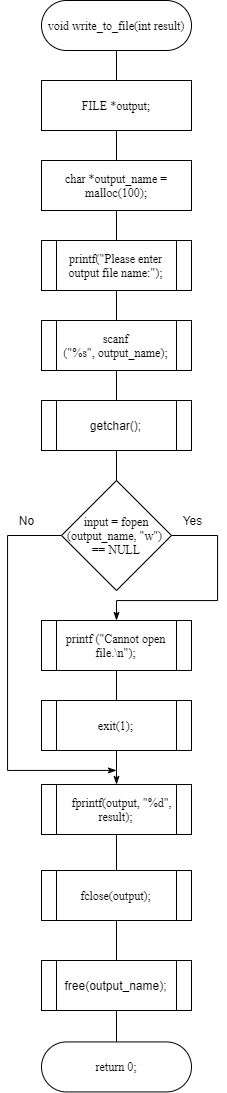


Рисунок 6 — Блок-схема функції write\_to\_file

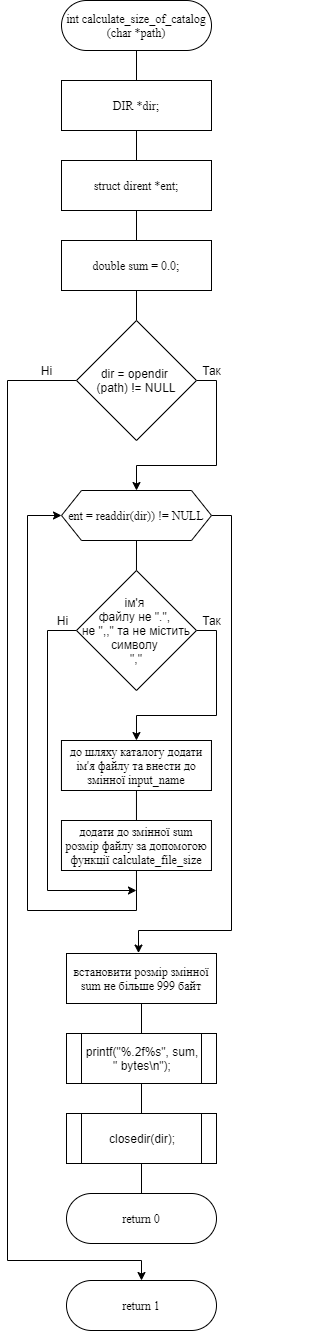
****

Рисунок 7 — Блок-схема функції calculate\_size\_of\_catalog

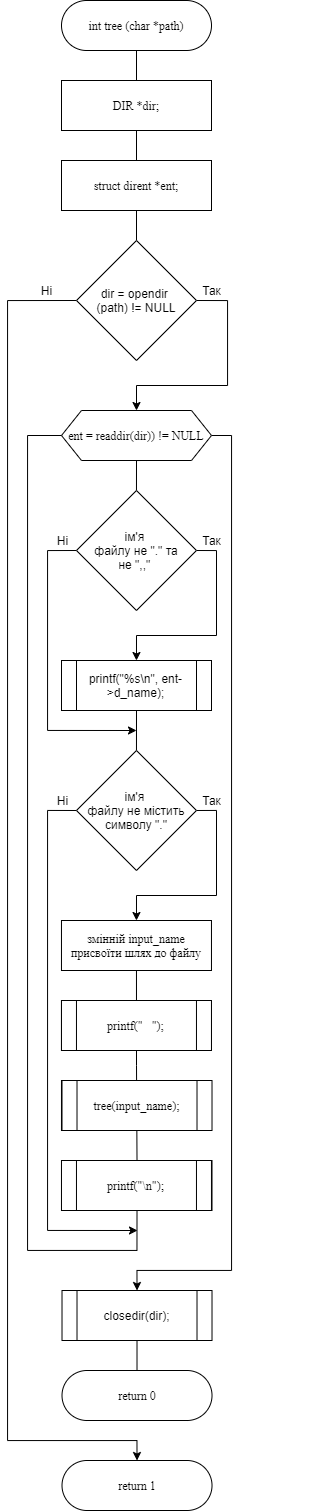


Рисунок 8 — Блок-схема функції tree

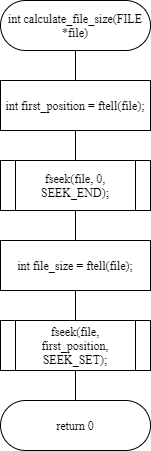
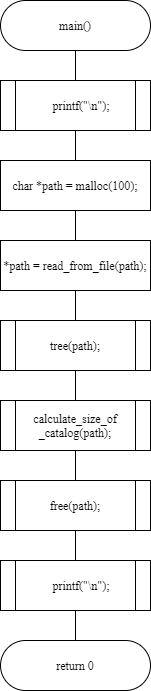
 ****

Рисунок 9, 10 — Блок-схема функції calculate\_file\_size та функції main другого та третього завдання

**Текст програми**

#include <stdlib.h>

#include <stdio.h>

#include <dirent.h>

#include <string.h>

char read\_from\_file(char \*string) {

FILE \*input;

char \*input\_name = malloc(100);

printf("Please enter input file name:");

scanf("%s", input\_name);

getchar();

if ((input = fopen(input\_name, "r")) == NULL) {

printf("Cannot open file.\n");

exit(1);

}

fscanf(input, "%[^EOF]", string);

free(input\_name);

fclose(input);

return \*string;

}

int calculate\_file\_size(FILE \*file)

{

int first\_position = ftell(file);

fseek(file, 0, SEEK\_END);

int file\_size = ftell(file);

fseek(file, first\_position, SEEK\_SET);

return file\_size;

}

int calculate\_size\_of\_catalog (char \*path) {

DIR \*dir;

struct dirent \*ent;

printf("\n");

double sum = 0.0;

if ((dir = opendir(path)) != NULL) {

/\* print all the files and directories within directory \*/

while ((ent = readdir(dir)) != NULL) {

if (strchr(ent->d\_name, '.') != NULL && strcmp(ent->d\_name, ".") != 0 && strcmp(ent->d\_name, "..") != 0) {

char \*input\_name = malloc(100);

strcpy(input\_name, path);

strncat(input\_name, "\\", 2);

strncat(input\_name, ent->d\_name, sizeof(ent->d\_name));

FILE \*file;

if ((file = fopen(input\_name, "r")) == NULL) {

printf("Cannot open file.\n");

exit(1);

}

sum += (double) calculate\_file\_size(file);

free(input\_name);

fclose(file);

}

}

if (sum > 999.0) sum = 999.0;

printf("%.2f%s", sum, " bytes\n");

closedir(dir);

return 0;

} else {

perror("");

return 1;

}

}

int tree (char \*path){

DIR \*dir;

struct dirent \*ent;

if ((dir = opendir (path)) != NULL) {

/\* print all the files and directories within directory \*/

while ((ent = readdir (dir)) != NULL) {

if (strcmp(ent->d\_name, ".") != 0 && strcmp(ent->d\_name, "..") != 0) {

printf("%s\n", ent->d\_name);

}

if (strchr(ent->d\_name, '.') == NULL) {

char \*input\_name = malloc(100);

strcpy(input\_name, path);

strncat(input\_name, "\\", 2);

strncat(input\_name, ent->d\_name, sizeof(ent->d\_name));

printf(" ");

tree(input\_name);

printf("\n");

}

}

closedir (dir);

return 0;

} else {

perror ("");

return 1;

}

}

int main()

{

printf("\n");

char \*path = malloc(100);

\*path = read\_from\_file(path);

tree(path);

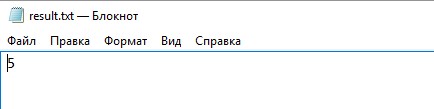
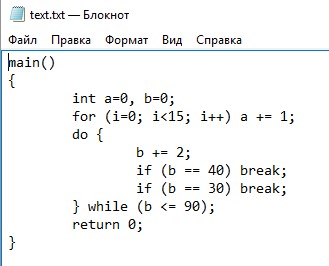
calculate\_size\_of\_catalog(path);

printf("\n");

return 0;

}

**Результати роботи програми**

****

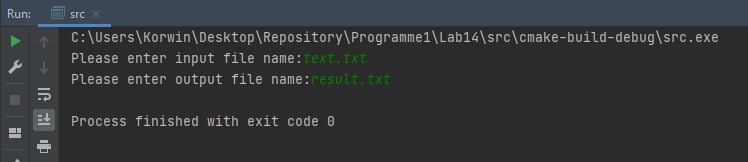


Рисунок 11 — Результат успішного виконання програми першого завдання

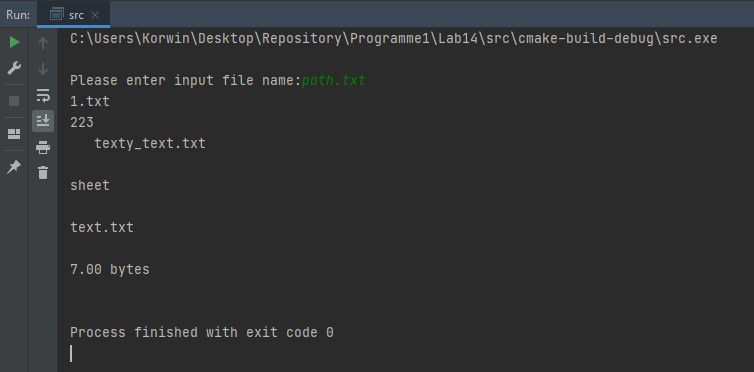


Рисунок 12 — Результат успішного виконання програми другого та третього завдання

**Висновки**

Під час виконання даної лабораторної роботи було отримано навички взаємодії з файлами та ії використанням при розв’язуванні задач.