# МИНОБРНАУКИ РОССИИ

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

# «ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)

**Кафедра МО ЭВМ**

# ОТЧЕТ

**по лабораторной работе №3**

# по дисциплине «Программирование» Тема: РАБОТА С ФАЙЛОВОЙ СИСТЕМОЙ

|  |  |  |
| --- | --- | --- |
| Студент гр. 0382 |  | Крючков А.М. |
| Преподаватель |  | Берленко Т.А. |

Санкт-Петербург 2021

# Цель работы.

Изучение функций для работы с файловой системой

# Задание.

**Вариант 2**

Задана иерархия папок и файлов по следующим правилам:

* название папок может быть только "add" или "mul"
* В папках могут находиться другие вложенные папки и/или текстовые файлы
* Текстовые файлы имеют произвольное имя с расширением .txt
* Содержимое текстовых файлов представляет собой строку, в

которой через пробел записано некоторое количество целых чисел Требуется написать программу, которая, запускается в корневой

директории, содержащей одну папку с именем "add" или "mul" и вычисляет и выводит на экран результат выражения состоящего из чисел в поддиректориях по следующим правилам:

* Если в папке находится один или несколько текстовых файлов, то

математическая операция определяемая названием папки (add = сложение, mul = умножение) применяется ко всем числам всех файлов в этой папке

* Если в папке находится еще одна или несколько папок, то сначала

вычисляются значения выражений, определяемые ими, а после используются уже эти значения

Ваше решение должно находиться в директории /home/box, файл с решением должен называться solution.c. Результат работы программы должен быть записан в файл result.txt. Ваша программа должна обрабатывать директорию, которая называется tmp.

# Основные теоретические положения.

Исходный код смотрите в приложении А.

Рассмотрим основные функции для работы с деревом файловой системы, объявления которых находятся в заголовочном файле dirent.h (также, может понадобиться включить заголовочный файл sys/types.h)

Для того, чтобы получить доступ к содержимому некоторой директории можно использовать функцию

DIR \*opendir(const char \*dirname);

Которая возвращает указатель на объект типа DIR с помощью которого можно из программы работать с заданной директорией.

Тип DIR представляет собой поток содержимого директории. Для того, что бы получить очередной элемент этого потока, используется функция

struct dirent \*readdir(DIR \*dirp);

Она возвращает указатель на объект структуры dirent, в котором хранится информация о файле. Основной интерес представляют поля, хранящие имя и тип объекта в директории (это может быть не только "файл" и "папка").

После завершения работы с содержимым директории, необходимо вызвать функцию

int closedir(DIR \*dirp);

Передав ей полученный функцией readdir() ранее дескриптор.

# Выполнение работы.

*Int main()*:

Открывается поток записи в файл *result.txt* и записывается туда результат работы функции *add()* над папкой *tmp.*

*Long long add(char\* curPath):*

Рекурсивная функция, проходится по всем файлам и директриям текущей папки*.* Дескриптор *readdir()* играет роль стека. Если найден файл, то в зависимости от названия вызывает функцию *add()* или *mul()*. Если найден файл, то с помощью функции *fscanf()*, числа этого файла добавляются в переменную res.

*Long long mul(char\* curPath):*

Работа аналогична функии *add()*, только действия сложения *res*, заменены на умножение.

# Тестирование.

Здесь результаты тестирования, которые помещаются на одну страницу. Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Входные данные | Выходные данные | Комментарии |
| 1. |  | 226 | Правильно! |

# Выводы.

Были изучены функции для работы с файловой системой.

# ПРИЛОЖЕНИЕ А ИСХОДНЫЙ КОД ПРОГРАММЫ

Сначала указываем имя файла, в котором код лежит в репозитории:

Название файла: src/main.c

#include <stdio.h> #include <dirent.h> #include <string.h>

long long mul(char \*curPath); long long add(char \*curPath) {

FILE \*fptr;

DIR \*dir = opendir(curPath); long long res = 0, n;

char c;

if (dir) {

struct dirent \*de = readdir(dir); while (de) {

if (de->d\_name[0] == '.') { de = readdir(dir); continue;

== 0) {

"mul") == 0) {

}

if (de->d\_type == DT\_DIR && strcmp(de->d\_name, "add")

strcat(curPath, "/add"); res = res + add(curPath);

curPath[strlen(curPath) - 4] = '\0';

} else if (de->d\_type == DT\_DIR && strcmp(de->d\_name,

strcat(curPath, "/mul"); res = res + mul(curPath);

curPath[strlen(curPath) - 4] = '\0';

} else { // text file strcat(curPath, "/"); strcat(curPath, de->d\_name); fptr = fopen(curPath, "r"); fscanf(fptr, "%lld", &n);

c = fgetc(fptr); res = res + n;

while (c != EOF && c != '\n' && fscanf(fptr,

"%lld", &n) != EOF) {

}

c = fgetc(fptr); res = res + n;

= '\0';

fclose(fptr);

curPath[strlen(curPath) - strlen(de->d\_name) - 1]

}

de = readdir(dir);

}

}

closedir(dir);

return res;

}

long long mul(char \*curPath) { FILE \*fptr;

DIR \*dir = opendir(curPath); long long res = 0, n, flag = 0; char c, c1;

if (dir) {

struct dirent \*de = readdir(dir); while (de) {

if (de->d\_name[0] == '.') { de = readdir(dir); continue;

== 0) {

"mul") == 0) {

}

if (de->d\_type == DT\_DIR && strcmp(de->d\_name, "add")

strcat(curPath, "/add"); res = res \* add(curPath);

curPath[strlen(curPath) - 4] = '\0';

} else if (de->d\_type == DT\_DIR && strcmp(de->d\_name,

strcat(curPath, "/mul"); res = res \* mul(curPath);

curPath[strlen(curPath) - 4] = '\0';

} else { // text file if (flag == 0) {

flag = 1;

res = 1;

"%lld", &n) != EOF) {

}

strcat(curPath, "/"); strcat(curPath, de->d\_name); fptr = fopen(curPath, "r"); fscanf(fptr, "%lld", &n);

c = fgetc(fptr); res = res \* n;

while (c != EOF && c != '\n' && fscanf(fptr,

c = fgetc(fptr); res = res \* n;

= '\0';

}

fclose(fptr);

curPath[strlen(curPath) - strlen(de->d\_name) - 1]

}

de = readdir(dir);

}

}

closedir(dir); return res;

}

int main() {

char cwd[10000];

// if (getcwd(cwd, sizeof(cwd)) != NULL) {

// } else {

// perror("getcwd() error");

// return 1;

// }

strcpy(cwd, "."); strcat(cwd, "/result.txt");

FILE \*fres = fopen(cwd, "w"); cwd[strlen(cwd) - 11] = '\0';

strcat(cwd, "/tmp");

// printf("%d\n", add(cwd));

// printf("%d\n", mul(cwd)); fprintf(fres, "%lld", add(cwd)); return 0;

}