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: Обход дерева файловой системы

Студент гр. 1304 _____ Дешура Д.В.

Преподаватель _____ Чайка К.В.

Санкт-Петербург
2022

Цель работы.

Создать программу, работающую с директориями и файлами. Изучить рекурсию, как метод обхода дерева в глубину.

Задание.

Задана иерархия папок и файлов по следующим правилам:

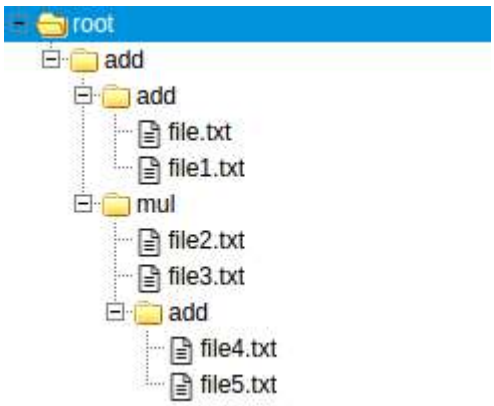
- название папок может быть только "add" или "mul"
- В папках могут находиться другие вложенные папки и/или текстовые файлы
- Текстовые файлы имеют произвольное имя с расширением .txt
- Содержимое текстовых файлов представляет собой строку, в которой через пробел записано некоторое количество целых чисел

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

Пример

(Программа в момент запуска находится в директории root)



file.txt: 1
file1.txt: 1
file2.txt: 2 2
file3.txt: 7
file4.txt: 1 2 3
file5.txt: 3 -1

Решение:

226

*Выражение в данном случае имеет вид: (((1+1))+((1+2+3+3+-1)*7*2*2))*

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**.

Выполнение работы.

В программе используются функции стандартной библиотеки из заголовочных файлов <stdio.h>, <stdlib.h>, <dirent.h>, <string.h>, <ctype.h>. Для решения задачи были реализованы 2 функции: sumator() и multiplicator(), проходящие по файлам текущей директории и соответственно складывая (перемножая) их содержимое, встречая вложенную папку эти функции рекурентно вызывают функции sumator() для add и multiplicator() для mul. Возвращают соответственно сумму или произведение обработанных чисел.

Результат работы программы для всей директории tmp выводится в файл result.txt.

Разработанный программный код см. в приложении А.

Тестирование.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные
1.	tmp add file.txt {1 3} mul file1.txt {2 4} file2.txt {-10} add file3.txt {-4 6} file4.txt {12}	-62

Выводы.

Выполнив лабораторную работу мы создали программу, работающую с директориями и файлами. Изучили рекурсивный метод обхода дерева в глубину.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: solution.c

```
#include <stdio.h>
#include "dirent.h"
#include "string.h"
#include "ctype.h"
#include "stdlib.h"


int multiplicator (int start, char* path);


int sumator(int start, char* path) {
    int res = start;
    DIR *dir = opendir(path);
    if (dir) {
        struct dirent *curdir = readdir(dir);
        while (curdir) {
            if (!strcmp(curdir -> d_name, "add")) {
                strcat(path, "/add");
                res += sumator(0, path);
                path[strlen(path) - 4] = '\0';
            } else {
                if (!strcmp(curdir -> d_name, "mul")) {
                    strcat(path, "/mul");
                    res += multiplicator(1, path);
                    path[strlen(path) - 4] = '\0';
                } else {
                    if(curdir -> d_type == DT_REG){
                        char* buf, str[100];
                        strcat(strcat(path, "/"), curdir -> d_name);
                        FILE *file = fopen(path, "r");
                        fgets(str, 100, file);

                        buf = strtok(str, " ");
                        while (buf) {
                            res += atoi(buf);
                            buf = strtok(NULL, " ");
                        }
                        fclose(file);
                        path[strlen(path) - strlen(curdir -> d_name)
- 1] = '\0';
                    }
                }
            }

            curdir = readdir(dir);
        }
        closedir (dir);
    }

    return res;
}
```

```
int multiplicator (int start, char* path) {
    int res = start;
    DIR *dir = opendir(path);
    if (dir) {
        struct dirent *curdir = readdir(dir);
        while (curdir) {
            if (!strcmp(curdir -> d_name, "add")) {
                strcat(path, "/add");
                res *= sumator(0, path);
                path[strlen(path) - 4] = '\0';
            } else {
                if (!strcmp(curdir -> d_name, "mul")) {
                    strcat(path, "/mul");
                    res *= multiplicator(1, path);
                    path[strlen(path) - 4] = '\0';
                } else {
                    if (curdir -> d_type == DT_REG) {
                        char* buf, str[100];
                        strcat(strcat(path, "/"), curdir -> d_name);
                        FILE *file = fopen(path, "r");
                        fgets(str, 100, file);

                        buf = strtok(str, " ");
                        while (buf) {
                            res *= atoi(buf);
                            buf = strtok(NULL, " ");
                        }
                        fclose(file);
                        path[strlen(path) - strlen(curdir -> d_name)
- 1] = '\0';
                    }
                }
            }

            curdir = readdir(dir);
        }
        closedir (dir);
    }

    return res;
}


int main() {
    char path[256] = "tmp";
    long long int res;
    DIR *dir = opendir(path);
    struct dirent *directory = readdir(dir);
    while(directory){
        if(!strcmp(directory -> d_name, "add")){
            strcat(path, "/add");
            res = sumator(0, path);
            break;
        }
        if(!strcmp(directory -> d_name, "mul")){
            strcat(path, "/mul");
            res = multiplicator(1, path);
```

```
            break;
        }
        directory = readdir(dir);
    }
    closedir (dir);

    FILE *file = fopen("result.txt", "w");
    fprintf(file, "%lld\n", res);
    fclose(file);
    return 0;
}
```