

МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МОЭВМ

ОТЧЕТ
по лабораторной работе №3
по дисциплине «Программирование»
Тема: «Обход файловой системы».

Студент гр. 1304

Сенников К.Д.

Преподаватель

Чайка К.В.

Санкт-Петербург

2022

Цель работы

Научиться работать с директориями и файлами с помощью языка программирования Си. Изучить рекурсивный способ обхода по дереву.

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

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

Подключены заголовочные файлы *stdio.h*, *stdlib.h*, *dirent.h*, *sys/types.h*, *string.h*. Создана функция *makeOperations* возвращающая значение *long long int* и принимающая на вход две переменные *path* – путь до директории или файла, обрабатываемый функцией, и *last_dir*, предыдущая директория, типа

*char **. Внутри функции создаются две переменные типа *long long int* – *res*, которой присваивается значение 0, и *tmp*. Создается указатель *dir_copy* на тип *char*, который будет хранить в себе абсолютный путь до директории или файла. Затем указателю *dir* на *DIR* было присвоено значение функции *opendir*, в которую был передан *path*. Затем с помощью трех условных операторов проводится проверка на предыдущую папку.

Если предыдущей папкой была *add*, то переменной *res* присваивается значение 0 и объявляется указатель *cur* на структуру *dirent*, принимающий значение функции *readdir*, которой на вход был подан указатель *dir*. Затем с помощью цикла *while* проведены проверки на имя считанного файла или директории.

- Если имя директории *add*, то указателю *dir_copy* передается адрес памяти, выделенной динамически в куче, с помощью функции *calloc*, которой на вход были поданы сумма длины строки *path* и длины строки *"/add"* и размер типа *char*. С помощью двух функций *strcpy* в *dir_copy* скопированы значения строк *path* и *"/add"*. К значению переменной *res* прибавляется значение функции *makeOperations*, которой на вход были поданы *dir_copy* и строка *"/add"*. После память, выделенная для *dir_copy*, очищается.
- Если имя директории *mul*, то указателю *dir_copy* передается адрес памяти, выделенной динамически в куче, с помощью функции *calloc*, которой на вход были поданы сумма длины строки *path* и длины строки *"/mul"* и размер типа *char*. С помощью двух функций *strcpy* в *dir_copy* скопированы значения строк *path* и *"/mul"*. К значению переменной *res* прибавляется значение функции *makeOperations*, которой на вход были поданы *dir_copy* и строка *"/mul"*. После память, выделенная для *dir_copy*, очищается.
- Если в имени считанного элемента была подстрока *".txt"*, то указателю *dir_copy* передается адрес памяти, выделенной динамически в куче, с помощью функции *calloc*, которой на вход

были поданы сумма длины строки `path`, длины строки, хранящей в себе название файла, и числа 2 и размер типа `char`. С помощью двух функций `strcpy` в `dir_copу` скопированы значения строк `path` и `"/"`, а с помощью `strcat` совершена конкатенация строк `dir_copу` и название файла, который храниться в `cur->d_name`. С помощью функции `fopen` на чтение открывается файл `dir_copу` и указатель на тип `FILE` записывается в `file`. С помощью цикла `while` и функции `fscanf` считываются числа в файле и записываются в переменную `tmp`, которая прибавляется к значению переменной `res`. После окончания цикла файл закрывается, а память `dir_copу` очищается.

В переменную `cur` передается следующий считанный элемент.

Если предыдущей папкой была `mul`, то переменной `res` присваивается значение 1 и объявляется указатель `cur` на структуру `dirent`, принимающий значение функции `readdir`, которой на вход был подан указатель `dir`. Затем с помощью цикла `while` проведены проверки на имя считанного файла или директории.

- Если имя директории `add`, то указателю `dir_copу` передается адрес памяти, выделенной динамически в куче, с помощью функции `calloc`, которой на вход были поданы сумма длинны строки `path` и длинны строки `"/add"` и размер типа `char`. С помощью двух функций `strcpy` в `dir_copу` скопированы значения строк `path` и `"/add"`. Значение переменной `res` умножается на значение функции `makeOperations`, которой на вход были поданы `dir_copу` и строка `"/add"`. После памяти, выделенная для `dir_copу`, очищается.
- Если имя директории `mul`, то указателю `dir_copу` передается адрес памяти, выделенной динамически в куче, с помощью функции `calloc`, которой на вход были поданы сумма длинны строки `path` и длинны строки `"/mul"` и размер типа `char`. С помощью двух функций `strcpy` в `dir_copу` скопированы значения строк `path` и `"/mul"`. Значение переменной `res` умножается на значение функции

makeOperations, которой на вход были поданы dir_cору и строка “/add”. После память, выделенная для dir_cору, очищается.

- Если в имени считанного элемента была подстрока “.txt”, то указателю dir_cору передается адрес памяти, выделенной динамически в куче, с помощью функции calloc, которой на вход были поданы сумма длины строки path, длины строки, хранящей в себе название файла, и числа 2 и размер типа char. С помощью двух функций strcpy в dir_cору скопированы значения строк path и “/”, а с помощью strcat совершена конкатенация строк dir_cору и название файла, который храниться в cur->d_name. С помощью функции fopen на чтение открывается файл dir_cору и указатель на тип FILE записывается в file. С помощью цикла while и функции fscanf считываются числа в файле и записываются в переменную tmp. А значение res умножается на tmp и записывается в res. После окончания цикла файл закрывается, а память dir_cору очищается.

В переменную cur передается следующий считанный элемент.

Если предыдущей папкой была tmp, то переменной res присваивается значение 0 и объявляется указатель cur на структуру dirent, принимающий значение функции readdir, которой на вход был подан указатель dir. Затем с помощью цикла while проведены проверки на имя считанного файла или директории.

- Если имя директории add, то указателю dir_cору передается адрес памяти, выделенной динамически в куче, с помощью функции calloc, которой на вход были поданы сумма длинны строки path и длинны строки “/add” и размер типа char. С помощью двух функций strcpy в dir_cору скопированы значения строк path и “/add”. К значению переменной res прибавляется значение функции makeOperations, которой на вход были поданы dir_cору и строка “/add”. После память, выделенная для dir_cору, очищается.

- Если имя директории `mul`, то указателю `dir_copu` передается адрес памяти, выделенной динамически в куче, с помощью функции `calloc`, которой на вход были поданы сумма длинны строки `path` и длинны строки `“/mul”` и размер типа `char`. С помощью двух функций `strcpy` в `dir_copu` скопированы значения строк `path` и `“/mul”`. К значению переменной `res` прибавляется значение функции `makeOperations`, которой на вход были поданы `dir_copu` и строка `“/mul”`. После память, выделенная для `dir_copu`, очищается.

В переменную `cur` передается следующий считанный элемент.

Функция *`makeOperations`* возвращает значение *`res`*.

В функции `main` выделена память для указателя `path` на тип `char` для строки `“tmp”`. В переменную `res` типа `long long int` записан 0. В указатель `resultf` на `FILE` записывается адрес файла `result.txt`, который открыт на чтение. В переменную `path` копируется строка `tmp` с помощью функции `strcpy`. Переменной `res` присвоено значение функции `makeOperations`, которой на вход поданы две строки `path` и `path`. Затем в файл `resultf` записывается значение `res` и файл закрывается.

Тестирование

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования.

№ п/п	Входные данные	Выходные данные	Комментарии
1.	tmp/add/add/file1.txt - 1 tmp/add/add/file2.txt - 1 tmp/add/mul/file3.txt - 10 tmp/add/mul/file4.txt - 10 tmp/add/mul/add/file5.txt - 1 tmp/add/mul/add/file6.txt - 1	./result.txt - 202	Программа отработала верно

Выводы

Изучен принцип работы с файлами и директориями с помощью рекурсивного обхода дерева в глубину. Написана программа, которая перебирает все файлы и считывает значения складывая или перемножая их в зависимости от названия директории, а затем записывает конечное значение в файл.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: solution.c

```
#include <stdio.h>
#include <stdlib.h>
#include <dirent.h>
#include <sys/types.h>
#include <string.h>

long long int makeOperations(char *path, char *last_dir){
    // printf("%s\n", path);
    long long int tmp, res = 0;
    char *dir_copy;
    DIR *dir = opendir(path);
    if(!strcmp(last_dir, "add")){
        res = 0;
        struct dirent *cur = readdir(dir);
        while(cur){
            if(strcmp(cur->d_name, "add") == 0){
                dir_copy = (char *)calloc(strlen(path) + 5,
sizeof(char));
                strcpy(dir_copy, path);
                strcat(dir_copy, "/add");
                res += makeOperations(dir_copy, "add");
                free(dir_copy);
            }
            else if(strcmp(cur->d_name, "mul") == 0){
                dir_copy = (char *)calloc(strlen(path) + 5,
sizeof(char));
                strcpy(dir_copy, path);
                strcat(dir_copy, "/mul");
                res += makeOperations(dir_copy, "mul");
                free(dir_copy);
            }
            else if(strstr(cur->d_name, ".txt") != NULL){
                dir_copy = (char *)calloc(strlen(path) + strlen(cur-
>d_name) + 2, sizeof(char));
                strcpy(dir_copy, path);
                strcat(dir_copy, "/");
                strcat(dir_copy, cur->d_name);
                FILE *file = fopen(dir_copy, "r");
                while(fscanf(file, "%lld", &tmp) != EOF){
                    res += tmp;
                }
                fclose(file);
                free(dir_copy);
            }
            cur = readdir(dir);
        }
    }
    else if(!strcmp(last_dir, "mul")){
        res = 1;
        struct dirent *cur = readdir(dir);
```



```

        while(cur){
            if(strcmp(cur->d_name, "add") == 0){
                dir_copy = (char *)calloc(strlen(path) + 5,
sizeof(char));
                strcpy(dir_copy, path);
                strcat(dir_copy, "/add");
                res += makeOperations(dir_copy, "add");
                free(dir_copy);
            }
            else if(strcmp(cur->d_name, "mul") == 0){
                dir_copy = (char *)calloc(strlen(path) + 5,
sizeof(char));
                strcpy(dir_copy, path);
                strcat(dir_copy, "/mul");
                res += makeOperations(dir_copy, "mul");
                free(dir_copy);
            }
            else if(strstr(cur->d_name, ".txt") != NULL){
                dir_copy = (char *)calloc(strlen(path) + strlen(cur-
>d_name) + 2, sizeof(char));
                strcpy(dir_copy, path);
                strcat(dir_copy, "/");
                strcat(dir_copy, cur->d_name);
                FILE *file = fopen(dir_copy, "r");
                while(fscanf(file, "%lld", &tmp) != EOF){
                    res += tmp;
                }
                fclose(file);
                free(dir_copy);
            }
            cur = readdir(dir);
        }
    }
    else if(!strcmp(last_dir, "tmp")){
        res = 0;
        struct dirent *cur = readdir(dir);
        while(cur){
            if(strcmp(cur->d_name, "add") == 0){
                dir_copy = (char *)calloc(strlen(path) + 5,
sizeof(char));
                strcpy(dir_copy, path);
                strcat(dir_copy, "/add");
                res += makeOperations(dir_copy, "add");
                free(dir_copy);
            }
            else if(strcmp(cur->d_name, "mul") == 0){
                dir_copy = (char *)calloc(strlen(path) + 5,
sizeof(char));
                strcpy(dir_copy, path);
                strcat(dir_copy, "/mul");
                res += makeOperations(dir_copy, "mul");
                free(dir_copy);
            }
            cur = readdir(dir);
        }
    }
    return res;
}

```

```
}

int main(int argc, char **argv){
    char *path = (char *)calloc(4, sizeof(char));
    long long int res = 0;
    FILE *resultf = fopen("result.txt", "w");
    strcpy(path, "tmp");
    res = makeOperations(path, path);
    fprintf(resultf, "%lld", res);
    fclose(resultf);
    return 0;
}
```