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: Строки. Рекурсия, циклы, обход дерева

Студент гр. 3344 _____ Жаворонок Д.Н.

Преподаватель _____ Глазунов С.А.

Санкт-Петербург

2024

Цель работы

Освоение работы с рекурсией на языке Си на примере использующей ее программы.

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

Выполнение работы

*Были подключены <stdio.h>, <stdlib.h>, <string.h> и <dirent.h> для работы с файлами и директориями. Была создана рекурсивная функция int listdir(const char *name, char *last_dir) для обхода дерева файлов и подсчета значений в соответствии с условием int c;. На вход функция принимает путь к текущей и предыдущей директории. В функции создаются специальные структуры DIR *dir; struct dirent *entry; для работы с директориями. Также создается переменная счетчик, для подсчета значений в текущей директории и происходит проверка на успешное открытие директории. Далее запускается цикл while ((entry = readdir(dir)) != NULL) для прохода по всем файлам в текущей директории. Если текущий файл является директорией, то мы формируем к ней путь и вызываем нашу функцию рекурсивно, возвращаемое значение сохраняем. Также происходит исключение путей “.” и “..”, чтобы исключить бесконечную рекурсию. Возвращаемое значение прибавляется или умножается на переменную счетчик, чтобы просчитать всю директорию. Если текущий файл является файлом, то формируется путь к нему, открывается поток к этому файлу и вызывается функция int count = counter(fp, last_dir); для подсчета значений в файле с учетом директории, которой он находится. После поток закрывается. Возвращаемое значение прибавляется или умножается на переменную счетчик, чтобы просчитать всю директорию. После цикла закрывается поток на директорию и возвращается переменная счетчик. В функции для подсчета данных внутри файла происходит сканирование подсчет данных с помощью while(fscanf(fp, "%d ", &c) == 1) overall += c;. В main запускается функция для просчета всех подкаталогов и происходит запись результата в текстовый файл с помощью FILE* fp = fopen("result.txt", "w"); fprintf(fp, "%d", overall); fclose(fp);*

Разработанный программный код см. в приложении А.

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1.	file.txt: 1 file1.txt: 1 file2.txt: 2 2 file3.txt: 7 file4.txt: 1 2 3 file5.txt: 3 -1 root/add/add/file.txt root/add/add/file1.txt root/add/mul/file2.txt root/add/mul/file3.txt root/add/mul/add/file4.txt root/add/mul/add/file5.txt	236 result.txt	-

Выводы

Была освоена работа с рекурсивными функциями на языке Си на примере использующей их программы.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.c

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

typedef struct dirent dirent;

int calc(FILE *file, char *dirname)
{
    if (!file)
        return -1;

    int mode = strcmp(dirname, "add") == 0;

    int result = mode ? 0 : 1;
    int temp = 0;
    while (fscanf(file, "%d ", &temp) == 1)
    {
        if (mode)
            result += temp;
        else
            result *= temp;
    }
    return result;
}

int iterate_over_dir(const char *dirname, char *prev_dir)
{
    DIR *dir;
    dirent *entry;

    dir = opendir(dirname);
    if (!dir)
        exit(-1);

    int is_add = strcmp(prev_dir, "add") == 0;
    int is_mul = strcmp(prev_dir, "mul") == 0;
    int result = is_mul;

    while ((entry = readdir(dir)) != NULL)
    {
        if (entry->d_type == DT_DIR)
        {
             if (strcmp(entry->d_name, ".") == 0 || strcmp(entry->d_name,
"..") == 0)
                continue;
            char *path = malloc(strlen(dirname) + strlen(entry->d_name) + 2);
             snprintf(path, strlen(dirname) + strlen(entry->d_name) + 2,
"%s/%s", dirname, entry->d_name);
            int n = iterate_over_dir(path, entry->d_name);
```

```
            if (is_add)
                result += n;
            else if (is_mul)
                result *= n;
            else
            {
                return n;
            }
        }
        else
        {
            char *temp = malloc(strlen(dirname) + strlen(entry->d_name) + 2);
             snprintf(temp, strlen(dirname) + strlen(entry->d_name) + 2,
"%s/%s", dirname, entry->d_name);
            FILE *file = fopen(temp, "r");
            int n = calc(file, prev_dir);
            fclose(file);
            if (is_add)
                result += n;
            else if (is_mul)
                result *= n;
        }
    }
    closedir(dir);
    return result;
}

int main(void)
{
    int result = iterate_over_dir("./tmp", ".");
    FILE *file = fopen("result.txt", "w");
    if (!file)
        return -1;

    fprintf(file, "%d", result);
    fclose(file);
    return 0;
}
```