

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

Вердин К.К.

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>`, `<sys/types.h>` `<dirent.h>` для работы с файлами и директориями. Была создана рекурсивная функция `long 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 директории и название 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 сформируем к ней путь при помощи созданной функции `char* pathcat(char* path1, char* path2)`, в которой расширяется память под строку для нового пути и происходит конкатенация путей. Затем вызываем нашу функцию рекурсивно, возвращаемое значение сохраняем. Также происходит исключение путей `“.”` и `“..”`, которые обозначают текущую и родительскую директории, 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 `long int overall = count(current_file, 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 и подсчет данных с помощью `while(fscanf(fp, "%ld ", &c) == 1) overall += c`; . В `main` запускается функция для подсчета 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: solution.c

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

char* pathcat(char* path1, char* path2)
{
    int res_path_len = strlen(path1)+strlen(path2)+2;
    char* res_path = malloc(sizeof(char)*res_path_len);
    sprintf(res_path, "%s/%s", path1, path2);
    return res_path;
}

long int count(FILE* current_file, char* operation)
{
    if (current_file==NULL) return -1;

    long int count, c;
    if (strcmp(operation, "add") == 0)
    {
        count = 0;
        while(fscanf(current_file, "%ld ", &c)==1) count += c;
        return count;
    }
    else if(strcmp(operation, "mul") == 0)
    {
        count = 1;
        while(fscanf(current_file, "%ld ", &c)==1) count *= c;
        return count;
    }
    else return 0;
}

long int listdir(char* path, char* last_dir)
{
    DIR* dir = opendir(path);
    if (dir == NULL) exit(-1);
    struct dirent* entry;

    long int c;
    if (strcmp(last_dir, "add") == 0) c = 0;
    else if (strcmp(last_dir, "mul") == 0) c = 1;

    while((entry = readdir(dir)) != NULL)
    {
        if (entry->d_type == DT_DIR)
```

```

        {
            if (strcmp(entry->d_name, "." ) == 0 || strcmp(entry-
>d_name, "..") == 0) continue;
            char* current_path = pathcat(path, entry->d_name);
            long int count = listdir(current_path, entry->d_name);

            if (strcmp(last_dir, "add") == 0) c+=count;
            else if(strcmp(last_dir, "mul") == 0) c*=count;
            else return count;
        }
    else if (entry->d_type == DT_REG)
    {
        char* current_path = pathcat(path, entry->d_name);
        FILE* current_file = fopen(current_path, "r");
        long int overall = count(current_file, last_dir);
        fclose(current_file);
        if (strcmp(last_dir, "add") == 0) c += overall;
        else if (strcmp(last_dir, "mul") == 0) c *= overall;
    }
}
closedir(dir);
return c;
}

int main()
{
    long int answer = listdir("./tmp", ".");
    FILE* result = fopen("result.txt", "w");
    fprintf(result, "%ld", answer);
    fclose(result);
    return 0;
}

```