

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 файловой системы

Студент гр. 1304

Павлов Д.Р.

Преподаватель

Чайка К.В.

Санкт-Петербург

2022

Цель работы.

Исследование рекурсивного обхода файловой системы.

Задание.

Дана некоторая корневая директория, в которой может находиться некоторое количество папок, в том числе вложенных. В этих папках хранятся некоторые текстовые файлы, имеющие имя вида .txt. В качестве имени файла используется символ латинского алфавита.

На вход программе подается строка. Требуется найти и вывести последовательность полных путей файлов, имена которых образуют эту строку.

! Регистрозависимость

! Могут встречаться файлы, в имени которых есть несколько букв и эти файлы использовать нельзя.

! Одна буква может встречаться один раз.

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.

Вариант 4.

Основные теоретические положения.

В данной работе использовались такие библиотеки, как `<string.h>`, `<dirent.h>`, `<stdlib.h>`, `<stdio.h>`. Использование библиотек `<stdlib.h>` и `<stdio.h>` достаточно очевидно, в то время как использование библиотеки `<string.h>` оправдано записью пути к файлам. Библиотека `<dirent.h>` нужна для работы с файлами и директориями.

Выполнение работы.

Сначала пишется функция, которая будет отвечать за считывание строки с динамическим выделением памяти. Она возвращает искомую строку. Далее мы напишем функцию, которая будет проверять является ли файл формата «.txt», если это так, то возвращается true. В противном случае — false. Далее идет функция, которая будет проходить все папки и подпапки

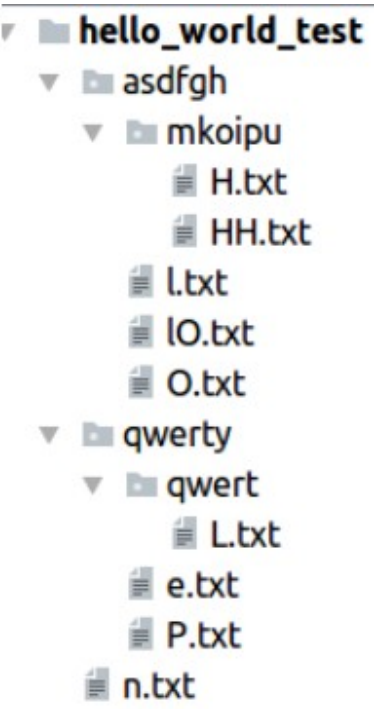
искомой директории. Она имеет 4 аргумента: 1 — путь директории, 2 — символ, имя файла которого ему равно; 3 — результирующий файл, в который будет записываться конечный ответ; 4 — флажок (так как функция рекурсивная, он нам нужен для выхода из цикла). Функция начинается с условия — если флажок равен true, то тогда выполняется цикл, в противном случае мы выходим из него. Под условием true мы открываем текущую директорию, если он не null, то переходим дальше: далее мы осуществляем сам процесс обхода. Для этого мы проверяем каждую папку на наличие файла, который нам нужен, и если этот файл не нашелся, то рассматриваем другую директорию. Поскольку мы не можем «откатываться назад», данное действие будет выполняться пока мы не найдем искомый файл или же не достигнем null'a. Следом идет функция main. В ней мы создаем файл куда будем записываться ответ и строку, буквы которой будем находить в директориях. Далее мы считываем строку и создаем цикл, который будет пробегаться по каждому символу введенной строки. В цикле мы пробегаемся по директориям при помощи ранее написанной функции list_dir.

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
-------	----------------	-----------------	-------------

1.		hello_world_test/asdfgh/ mkoipu/H.txt hello_world_test/qwerty/e.txt hello_world_test/qwerty/qw- ert/L.txt hello_world_test/asdfgh/l.txt hello_world_test/asdfgh/ O.txt	Ответ правильный
----	---	---	------------------

Выводы.

Была изучена рекурсия и применена для обхода файловой системы.

Разработана программа, которая ищет файлы с определённым названием и записывает в текстовый файл их полные пути.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.c

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

char* get_input_string(){
    char* result = NULL;
    result = malloc(sizeof(char));

    int i = 0;
    do{
        result = realloc(result, sizeof(char)*(i+1));
        scanf("%c", &result[i]);
        if (result[i] == '\n'){
            result[i] = '\0';
            break;
        }
        i++;
    }while (1);

    return result;
}

_Bool is_filename_correct(const char* filename){
    if (strlen(filename) == 5){
        if ((filename[1] == '.') && (filename[2] == 't') && (filename[3]
== 'x') && (filename[4] == 't')){
            return true;
        }
    }
    return false;
}

void list_dir(const char *dirPath, char symbol, FILE* res, _Bool flag){
    if (flag == true) {
        DIR *dir = opendir(dirPath);
```

```

        if (dir) {
            struct dirent *de = readdir(dir);

            while (de) {
                if (strcmp(de->d_name, ".") && strcmp(de->d_name, ".."))
{
                    char path[200];

                    path[0] = '\0';
                    strcat(path, dirPath);
                    strcat(path, de->d_name);

                                if (strstr(de->d_name, ".txt") != NULL &&
is_filename_correct(de->d_name) == true &&
                                (de->d_name)[0] == symbol) {
                                    flag = false;
                                    fprintf(res, "%s\n", path);
                                    return;
                                } else {
                                    //printf("%s\n", s);
                                }
                                if (opendir(path) == NULL) {

                                    } else {
                                        strcat(path, "/");
                                        //strcat(path, "\\");
                                        list_dir(path, symbol, res, flag);
                                    }
                                }
                                de = readdir(dir);
                            }
                        }
                    closedir(dir);
                }else if(flag == false){
                    return;
                }
            }

int main() {
    char* string = NULL;
    FILE* result;
    result = fopen("result.txt", "w");

```

```

//char* head_path = "/Users/pavlov/Desktop/hello_world_test/";
//char* head_path = "\\tmp\\";
char* head_path = "./tmp/";
string = get_input_string();
int i;
_Bool flag = true;
for (i = 0; i < strlen(string); i++) {
    list_dir(head_path,string[i], result, flag);
    flag = true;
}
fclose(result);
free(string);
return 0;
}

```