**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

|  |  |  |
| --- | --- | --- |
| Студентка гр. 3343 |  | Лобова Е. И. |
| Преподаватель |  | Государкин Я.С. |

Санкт-Петербург

2024

## Цель работы

Целью работы является освоение работы с рекурсивными функциями и

файловой системой, а также ее рекурсивным обходом.

Для достижения поставленной цели требуется решить следующие задачи:

1) Ознакомиться с понятием рекурсии;

2) Освоить написание рекурсивных функций в языке Си;

3) Изучить работу с файловой системой в языке Си;

4) Написать программу для рекурсивного обхода всех файлов в папке в

том числе во вложенных папках.

## 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 <filename>.txt

В каждом текстовом файле хранится одна строка, начинающаяся с числа вида:

<число><пробел><латинские буквы, цифры, знаки препинания> ("124 string example!")

Требуется написать программу, которая, будучи запущенной в корневой директории, выведет строки из файлов всех поддиректорий в порядке возрастания числа, с которого строки начинаются

Пример:

root/file.txt: 4 Where am I?

root/Newfolder/Newfile.txt: 2 Simple text

root/Newfolder/Newfolder/Newfile.txt: 5 So much files!

root/Newfolder(1)/Newfile.txt: 3 Wow? Text?

root/Newfolder(1)/Newfile1.txt: 1 Small text

Решение:

1 Small text

2 Simple text

3 Wow? Text?

4 Where am I?

5 So much files!

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.

## Выполнение работы

Для решения задания лабораторной работы была создана структура data с полями char text[100], int n, которые обозначают число и последующую часть строки.

Функции, реализованные в программе:

* Функция *data\* read\_file(const char \*file\_name)* открывает файл на чтение, имя которого передано в функцию, считывает данные из него в созданную структуру, закрывает его и возвращает указатель на структуру.
* Функция *void write\_to\_file(const char \*file\_name, int\* count, data\*\* strings)* открывает файл на запись, имя которого передано в функцию, записывает отсортированные строки в него и очищает память.
* Функция *char \*pathcat(const char \*path1, const char \*path2)* принимает на вход две части пути к файлу или директории. Выделяется память под новую строку и с помощью функции sprintf() в строку соединяются две части пути, что и возвращается.
* Функция *data\*\* list\_dir(const char \*dir\_name, int\* count, data\*\* strings)* принимает на вход указатель на имя директории или файла, массив указателей на уже полученные структуры с информацией из файлов и их количество. Выполняется проверка - открыт файл или директория. Если файл, то получается его полное имя с помощью функции *char\* pathcat()* и передается в функцию *data\* read\_file(),* в массив записывается полученный указатель на структуру. Если директория, то так же получается полный путь и вызывается эта же функция, но подается путь этой директории(рекурсия).
* Функция *int compar(const void \*a, const void \*b)* - компаратор для qsort, которая сравнивает структуры по значению поля int n.

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 |
|  | root/file.txt: 4 Where am I?  root/Newfolder/Newfile.txt: 2 Simple text  root/Newfolder/Newfolder/Newfile.txt: 5 So much files!  root/Newfolder(1)/Newfile.txt: 3 Wow? Text?  root/Newfolder(1)/Newfile1.txt: 1 Small text | 1 Small text  2 Simple text  3 Wow? Text?  4 Where am I?  5 So much files! | Выходные данные корректны. |

## Выводы

Были изучены работа с рекурсивными функциями и их написание на языке Си, файловая система и основные функции библиотеки *<dirent.h>* для работы с файлами и директориями. Так же написана программа, реализующая рекурсивный обход по файловой системе.

# Приложение А Исходный код программы

Название файла: main.c

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <dirent.h>

#define BLOCK 100

typedef struct data{

char text[BLOCK];

int n;

}data;

int compar(const void \*a, const void \*b) {

const data\* data\_a = \*(data\*\*)a;

const data\* data\_b = \*(data\*\*)b;

if (data\_a->n < data\_b->n) return -1;

if (data\_a->n > data\_b->n) return 1;

return data\_a->n == data\_b->n;

}

data\* read\_file(const char \*file\_name){

char str\_file[BLOCK];

FILE \*fin = fopen(file\_name,"r");

data\* f = NULL;

if (fin){

f = (data\*)malloc(sizeof(data));

fscanf(fin, "%d", &(f->n));

fgets(str\_file, BLOCK, fin);

strncpy(f->text, str\_file, BLOCK);

fclose(fin);

}else{

printf("Failed to open %s file\n", file\_name);

}

return f;

}

void write\_to\_file(const char \*file\_name, int\* count, data\*\* strings){

FILE \*fout = fopen(file\_name, "w");

if (fout){

for (int i = 0; i<(\*count); i++){

fprintf(fout, "%d%s\n", strings[i]->n, strings[i]->text);

free(strings[i]);

}

free(strings);

fclose(fout);

} else{

printf("Failed to open %s file\n", file\_name);

}

}

char \*pathcat(const char \*path1, const char \*path2){

int res\_path\_len = strlen(path1) + strlen(path2) + 2;

char \*res\_path = malloc(res\_path\_len \* sizeof(char));

sprintf(res\_path, "%s/%s", path1, path2);

return res\_path;

}

data\*\* list\_dir(const char \*dir\_name, int\* count, data\*\* strings){

DIR \*dir = opendir(dir\_name);

if(dir){

struct dirent \*de = readdir(dir);

while (de){

if(de->d\_type == DT\_REG && strstr(de->d\_name, ".txt") != NULL && strstr(de->d\_name, "result.txt") == NULL){

char \*name\_file = pathcat(dir\_name, de->d\_name);

data\* buf = read\_file(name\_file);

if (buf){

strings[(\*count)++] = buf;

}

free(name\_file);

}else if (de->d\_type == DT\_DIR && strcmp(de->d\_name, ".") != 0 && strcmp(de->d\_name, "..") != 0){

char \*new\_dir = pathcat(dir\_name, de->d\_name);

list\_dir(new\_dir, count, strings);

free(new\_dir);

}

de = readdir(dir);

}

closedir(dir);

}else

printf("Failed to open %s directory\n", dir\_name);

}

int main(){

int count;

count = 0;

data\*\* strings = (data\*\*)malloc(50\*BLOCK\*sizeof(data\*));

list\_dir(".", &count, strings);

qsort(strings, count, sizeof(data\*), compar);

write\_to\_file("result.txt", &count, strings);

return 0;

}