**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**](https://e.moevm.info/mod/lesson/view.php?id=905)

|  |  |  |
| --- | --- | --- |
| Студент гр. 3343 |  | Пухов А.Д. |
| Преподаватель |  | Государкин Я.С. |

Санкт-Петербург

2024

## Цель работы.

Изучение работы с файлами и директориями. Написание программы на языке Си для рекурсивного поиска нужных файлов среди директорий.

## Задание.

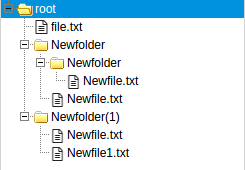
Дана некоторая корневая директория, в которой может находиться некоторое количество папок, в том числе вложенных. В этих папках хранятся некоторые текстовые файлы, имеющие имя вида *<filename>*.txt

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.

Структура **Text**:

* **char text[200]** — хранит строку
* **int number** — хранит число с которого начинается строка

Функции:

* **Text \*readfile(const char \*file\_name)** — открывает и считывает содержимое файла.
* **Text \*\*list\_dir(const char \*dir\_name, int \*size, Text \*\*strings)** — рекурсивно обходит директории и ищет файлы в нужном формате(txt).
* **char \*pathcat(const char \*path1, const char \*path2)** — склеивает две части пути к директории или к файлу, и возвращает полный путь.
* **write(const char \*name\_file, int size, Text \*\*strings)** — записывет результат программ в файл.
* **int compar(const void \*a, const void \*b)** — компаратор для qsort.

Написанный программный код находится 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! | ОК |

## Выводы.

В данной лабораторной работе была изучена работа с файлами и с директориями спомощи библиотеки dirent.h в языке Си.

# Приложение А Исходный код программы

Название файла: lb3.c

#include <stdio.h>

#include <dirent.h>

#include <stdlib.h>

#include <string.h>

typedef struct Text

{

char text[200];

int number;

} Text;

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;

}

Text \*readfile(const char \*file\_name)

{

char str[200];

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

Text \*file = NULL;

if (new\_file)

{

file = (Text \*)malloc(sizeof(Text));

fscanf(new\_file, "%d", &(file->number));

fgets(str, 200, new\_file);

strncpy(file->text, str, 200);

fclose(new\_file);

}

return file;

}

Text \*\*list\_dir(const char \*dir\_name, int \*size, Text \*\*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 \*file\_name = pathcat(dir\_name, de->d\_name);

Text \*s = readfile(file\_name);

if (s)

{

strings[(\*size)++] = s;

}

free(file\_name);

}

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, size, strings);

free(new\_dir);

}

de = readdir(dir);

}

closedir(dir);

}

else

{

printf("error open dir %s\n", dir\_name);

}

}

void write(const char \*name\_file, int size, Text \*\*strings)

{

FILE \*f = fopen(name\_file, "w");

if (f)

{

for (int i = 0; i < size; i++)

{

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

free(strings[i]);

}

free(strings);

fclose(f);

}

else

{

printf("error open result.txt");

}

}

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

{

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

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

if (Text\_a->number < Text\_b->number)

return -1;

if (Text\_a->number > Text\_b->number)

return 1;

return Text\_a->number == Text\_b->number;

}

int main()

{

int size;

size = 0;

Text \*\*strings = (Text \*\*)malloc(200 \* 200 \* sizeof(Text));

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

qsort(strings, size, sizeof(Text \*), compar);

write("result.txt",size, strings);

return 0;

}