**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: *Рекурсивный обход деревьев*

|  |  |  |
| --- | --- | --- |
| Студент гр. 6303 |  | Фокин К.С. |
| Преподаватель |  | Берленко Т.А. |

Санкт-Петербург

2017

**Цель работы:** требуется написать программу, которая выводит последовательность полных путей файлов, имена которых образуют заданную строку.

**Ход работы:**

В ветке **fokin\_sem2\_lr3** репозитория **pr-2016-6303** создана папка **fokin\_sem2\_lr3** с проектом, состоящим из 1 файла ***main\_lr3.c***

**Формулировка задания:**

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 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.

**Ниже приведен код программы:**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <sys/types.h>

#include <dirent.h>

#define DIR\_SIZE 100

#define STR\_SIZE 100

#define PATH\_SIZE 1000

typedef struct letter

{

char symbol;

char path[PATH\_SIZE + 1];

int flag;

} letter;

int fullName (letter \*letterPath, int length)

{

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

if (!((letterPath + i)->flag))

return 0;

return 1;

}

void checkLetter (letter \*letterPath, char \*currDir, char \*fileName, int length)

{

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

{

if (((letterPath + i)->symbol == \*fileName) && !((letterPath + i)->flag))

{

\*(fileName + strlen(fileName)) = '.';

int currDir\_len = strlen(currDir);

strcat(currDir, "/");

strcpy((letterPath + i)->path, strcat(currDir, fileName));

(letterPath + i)->flag = 1;

\*(currDir + currDir\_len) = '\0';

break;

}

}

}

int viewDir (char \*dirName, letter \*letterPath, int length)

{

char currDir[DIR\_SIZE];

strcpy(currDir, dirName);

DIR \*dir = opendir(currDir);

struct dirent \*content = readdir(dir);

if(dir)

while(content)

{

if (content->d\_type == 8)

{

char \*fileName = strtok(content->d\_name, ".");

checkLetter(letterPath, currDir, fileName, length);

if (fullName(letterPath, length))

return 0;

}

if ((content->d\_type == 4) && strcmp(".", content->d\_name) && strcmp("..", content->d\_name))

{

int currDir\_len = strlen(currDir);

strcat(currDir, "/");

strcat(currDir, content->d\_name);

viewDir(currDir, letterPath, length);

\*(currDir + currDir\_len) = '\0';

}

content = readdir(dir);

}

closedir(dir);

return 0;

}

int main ()

{

char dirName[DIR\_SIZE] = {'.', '\0'};

char str[STR\_SIZE];

fgets(str, (STR\_SIZE + 2), stdin);

int length = (strlen(str) - 1);

letter \*letterPath = (letter\*)malloc(length \* sizeof(letter));

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

{

(letterPath + i)->symbol = \*(str + i);

(letterPath + i)->flag = 0;

}

viewDir(dirName, letterPath, length);

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

printf("%s\n", (letterPath + i)->path);

return 0;

}

Для обхода каталогов, поиска и сохранения полных путей нужных нам файлов реализован набор функций:

1. **viewDir** – функция получает на вход имя директории, в которой она должна проверить все файлы на соответствие и вложенные директории;
2. **checkLetter** – функция получает на вход имя файла и проверяет, есть ли в заданном слове буква имени файлаж
3. **fullName** – функция проверяет, собрано ли уже из имен файлов заданное слово.

Созданная папка с файлом загружена в репозиторий на **Github** с помощью следующих команд:

* git add fokin\_sem2\_lr3
* git commit –m “Fokin LR №3 done”
* git push origin fokin\_sem2\_lr3

**Вывод:** в ходе лабораторной работы изучены возможности применения рекурсивного обхода папок и каталогов директории в языке Си. Полученные знания закреплены в ходе разработки программы **main\_lr3.c** для поиска файлов с определенными характеристиками.