**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МОЭВМ**

отчет

**по лабораторной работе №3**

**по дисциплине «Программирование»**

Тема: **Строки. Рекурсия, циклы, обход дерева.**

|  |  |  |
| --- | --- | --- |
| Студент гр. 6304 |  | Рыбин А.С. |
| Преподаватель |  | Берленко Т.А. |

Санкт-Петербург

2017

Оглавление

[**Оглавление** 2](#_Toc482736150)

[Цель: 3](#_Toc482736151)

[Задание: 3](#_Toc482736152)

[**Содержание:** 3](#_Toc482736153)

[Структура для итерации по файлам 3](#_Toc482736154)

[Функция открытия файла и его чтения в объект для последующей итерации 3](#_Toc482736155)

[Функция рекурсивного просмотра директорий (открытия и чтения файлов внутри) 4](#_Toc482736156)

[Функция итерации по списку файлов и его сортировки, вывода 4](#_Toc482736157)

[**Вывод** 5](#_Toc482736158)

[Приложение 6](#_Toc482736159)

Цель:

Освоение приёмов применения рекурсии и самостоятельной её реализации. Замена циклов рекурсией, рекурсивный обход дерева.

Задание:

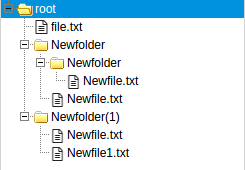
Дана некоторая корневая директория, в которой может находиться некоторое количество папок, в том числе вложенных. В этих папках хранятся некоторые текстовые файлы, имеющие имя вида *<filename>*.txt

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

Содержание:

**Структура для итерации по файлам**

struct files

{

char\* info;

struct files\* next;

};

Функция открытия файла и его чтения в объект для последующей итерации

void readfile(char\* name,struct files\* element)

{

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

fseek(file,0,SEEK\_END);

long size = ftell(file);

fseek(file,0,SEEK\_SET);

element->info = (char\*)malloc(sizeof(char)\*size);

fgets(element->info,size,file);

element->next = NULL;

fclose(file);

}

**Функция рекурсивного просмотра директорий (открытия и чтения файлов внутри)**

void listdir(const char\* startdir)

{

char current\_path[1000];

strcpy(current\_path,startdir);

DIR\* dir = opendir(current\_path);

struct dirent\* de = readdir(dir);

if(dir)

{

while(de)

{

if(de->d\_type == DT\_REG)

{

if(strstr(de->d\_name,".txt") != NULL)

{

int len = strlen(current\_path);

strcat(current\_path,"/");

strcat(current\_path,de->d\_name);

struct files\* newfile = (struct files\*)malloc(sizeof(struct files));

readfile(current\_path,newfile);

push(Head,newfile);

current\_path[len] = '\0';

}

}

else if(de->d\_type == DT\_DIR &&

0 != strcmp(".",de->d\_name) &&

0 != strcmp ("..",de->d\_name))

{

int len = strlen(current\_path);

strcat(current\_path,"/");

strcat(current\_path,de->d\_name);

listdir((const char\*)current\_path);

current\_path[len] = '\0';

}

de = readdir(dir);

}

closedir(dir);

}

}

**Функция итерации по списку файлов и его сортировки, вывода**

void sortandprint(struct files\* head)

{

int number = counter(head);

struct files\*\* ptr = (struct files\*\*)malloc(sizeof(struct files\*) \* number);

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

{

ptr[i] = head;

head = head->next;

}

qsort(ptr,number,sizeof(struct files\*),compare);

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

{

printf("%s\n",ptr[i]->info);

free(ptr[i]->info);

free(ptr[i]);

}

free(ptr);

}

# Вывод

В ходе выполнения лабораторной работы, используя стандартные средства языка С, был смоделирован связный однонаправленный список, который использовался для итерации по прочитанным файлам, изучен интерфейс работы с файловой системой и реализован рекурсивный проход по каталогам, подкаталогам и чтение файлов в них.

Приложение

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <dirent.h>

#include <sys/types.h>

struct files

{

char\* info;

struct files\* next;

};

void push(struct files\* head,struct files\* element);

int counter(struct files\* head);

void readfile(char\* name,struct files\* element);

void listdir(const char\* startdir);

void sortandprint(struct files\* head);

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

{

return atoi(((struct files\*)a)->info) - atoi(((struct files\*)b)->info);

}

struct files\* Head = NULL;

int main ()

{

listdir(".");

sortandprint(Head);

return 0;

}

void push(struct files\* head,struct files\* element)

{

if(Head == NULL)

Head = element;

else

{

while(head->next)

head = head->next;

head->next = element;

}

}

int counter(struct files\* head)

{

int count = 0;

while(head)

{

count++;

head = head->next;

}

return count;

}

void readfile(char\* name,struct files\* element)

{

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

fseek(file,0,SEEK\_END);

long size = ftell(file);

fseek(file,0,SEEK\_SET);

element->info = (char\*)malloc(sizeof(char)\*size);

fgets(element->info,size,file);

element->next = NULL;

fclose(file);

}

void listdir(const char\* startdir)

{

char current\_path[1000];

strcpy(current\_path,startdir);

DIR\* dir = opendir(current\_path);

struct dirent\* de = readdir(dir);

if(dir)

{

while(de)

{

if(de->d\_type == DT\_REG)

{

if(strstr(de->d\_name,".txt") != NULL)

{

int len = strlen(current\_path);

strcat(current\_path,"/");

strcat(current\_path,de->d\_name);

struct files\* newfile = (struct files\*)malloc(sizeof(struct files));

readfile(current\_path,newfile);

push(Head,newfile);

current\_path[len] = '\0';

}

}

else if(de->d\_type == DT\_DIR &&

0 != strcmp(".",de->d\_name) &&

0 != strcmp ("..",de->d\_name))

{

int len = strlen(current\_path);

strcat(current\_path,"/");

strcat(current\_path,de->d\_name);

listdir((const char\*)current\_path);

current\_path[len] = '\0';

}

de = readdir(dir);

}

closedir(dir);

}

}

void sortandprint(struct files\* head)

{

int number = counter(head);

struct files\*\* ptr = (struct files\*\*)malloc(sizeof(struct files\*) \* number);

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

{

ptr[i] = head;

head = head->next;

}

qsort(ptr,number,sizeof(struct files\*),compare);

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

{

printf("%s\n",ptr[i]->info);

free(ptr[i]->info);

free(ptr[i]);

}

free(ptr);

}