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

ЦЕЛЬ:

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

ЗАДАНИЕ:

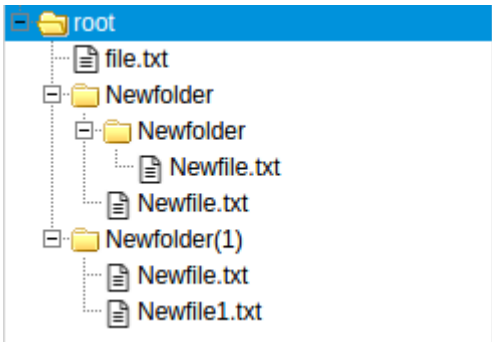
Дана некоторая корневая директория, в которой может находиться некоторое количество папок, в том числе вложенных. В этих папках хранятся некоторые текстовые файлы, имеющие имя вида <*filename*>.txt

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);
}
```