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 _____ Зубов К.А.

Преподаватель _____ Берленко Т.А.

Санкт-Петербург

2017

Оглавление

Цель

Написание программы, которая, будучи запущенной в корневой дирректории, выведет строки из файлов всех поддиректорий в порядке возрастания числа, с которого начинаются строки.

Задание

Дана некоторая корневая дирректория, 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!").

Содержание

В ходе выполнения данной работы были использованы следующие функции:

1. Функция открытия файла

Функция, которая открывает txt файл и возвращает указатель на первую его строку (если она есть).
В данной функции происходит открытие файла, определение размера файла для корректного выделения памяти, выделение памяти под строку. Первая строка txt файла считывается в выделенную память.Файл закрывается, указатель на строку возвращается.

```
char* read_file(char* file_name)
{
	FILE* file = fopen(file_name, "r"); //Открытие файла

	fseek(file,0, SEEK_END);  //Определение размера файла для корректного выделения памяти
	int file_size = ftell(file);
	fseek(file,0, SEEK_SET);

	char*str_in_file = (char*)malloc(sizeof(char)*(file_size));    //Выделяется память под строку.

	fgets(str_in_file, file_size*sizeof(char), file);       // Первая строка txt файла считывается в выделенную память.Файл закрывается, указатель на строку возвращается.
	fclose(file);
	return str_in_file;
}
```

2. Обход каталога

Данная функция обходит каталог и при наличии txt файла открывает его с помощью функции read_file. Полученный указатель помещается в массив из указателей на строки. В данной функции выделяется память под строку, в которую будет помещаться путь к файлам. Используется NAME_MAX, определяющий макс. длину имени файла. Открывается поток каталога и проверяется на корректное открытие. Далее с помощью функции readdir считывается структура с информацией о первом файле в каталоге.
Происходит цикл, который выполняется, пока в текущей директории будут объекты. К строке, содержащий путь, добавляется имя объекта, проверяется, является ли объект файлом или директорией, если объект - это непустой txt файл,то возвращается указатель на первую строку из него, если объект - это директория не родительская и не текущая, то рекурсивно вызывается функция read_catalog для прочтения данной директории, после проверки текущего объекта current_path "обрезается" до первоначального положения. Далее считывается следующий объект из данной директории. После обхода всех объектов в директории, она закрывается.

```
void read_catalog(const char* dir_name,char**strs_in_file, int* len)
```

```
{
	char* current_path = (char*)malloc(sizeof(char)*(strlen(dir_name)+NAME_MAX));//Выделяется память под строку, в которую будет помещаться путь к файлам. Используется NAME_MAX, определяющий макс. длину имени файла.
	strcpy(current_path,dir_name);

	DIR *current_dir = opendir(current_path);  //Открывается поток каталога и проверяется на корректное открытие. Далее с помощью функции readdir считывается структура с информацией о первом файле в каталоге.
	if(current_dir == NULL)
		return;
	dirent* file_in_current_dir = (dirent*)readdir(current_dir);

		while(file_in_current_dir)	//Цикл выполнятеется, пока в текущей директории будут объекты*/
	{
		int path_len = strlen(current_path);	//К строке, содержащий путь, добавляется имя объекта
			strcat(current_path, "/");
			strcat(current_path, file_in_current_dir->d_name);


			if		//Проверяется,является ли объект файлом или директорией
			(
				file_in_current_dir->d_type == DT_REG &&
				strstr(file_in_current_dir->d_name, ".txt")!=NULL
			)
			{
				if((strs_in_file[*len] = (char*)read_file(current_path))!=NULL ) //Если объект - это  непустой  txt файл,то возвращается указатель на первую строку из него
					(*len)++;
			}
		if
			(
				file_in_current_dir->d_type == DT_DIR &&	// Если объект - это директория не родительская и не текущая, то рекурсивно вызывается функция read_catalog для прочтения данной директории
			strcmp(".",file_in_current_dir->d_name)!=0 &&
				strcmp("..",file_in_current_dir->d_name)!=0
			)
			{
			read_catalog(current_path,strs_in_file, len);
			}
			current_path[path_len] = '\0';		//После проверки текущего объекта current_path "обрезается" до первоначального положения. Далее считывается следующий объект из данной директории
			file_in_current_dir = (dirent*)readdir(current_dir);
		}
  closedir(current_dir);  //После обхода всех объектов в директории, она закрывается
}
```

3. Компаратор для функции qsort

```
int compare(const void* a, const void* b)
{
```

```
    return atoi(*(char**)a) - atoi(*(char**)b);
}
```

Вывод

В ходе выполнения работы изучено использование рекурсивной функции для обхода дерева.

Приложение

```
#include <stdio.h>
#include <sys/types.h>
#include <dirent.h>
#include <string.h>
#include <stdlib.h>
```

```
typedef struct dirent dirent;

int compare(const void* a, const void* b)
{
	return atoi(*(char**)a) - atoi(*(char**)b);
}

char* read_file(char* file_name)    //     Функция, которая открывает txt файл и возвращает указатель на первую его строку (если она есть)
{
	FILE* file = fopen(file_name, "r"); //Открытие файла

	fseek(file,0, SEEK_END);  //Определение размера файла для корректного выделения памяти
	int file_size = ftell(file);
	fseek(file,0, SEEK_SET);

	char*str_in_file = (char*)malloc(sizeof(char)*(file_size));    //Выделяется память под строку.

	fgets(str_in_file, file_size*sizeof(char), file);      // Первая строка txt файла считывается в выделенную память.Файл закрывается, указатель на строку возвращается.
	fclose(file);
	return str_in_file;
}

void read_catalog(const char* dir_name,char**strs_in_file, int* len) //Данная функция обходит каталог и при наличии txt файла открывает его с помощью функции read_file. Полученный указатель помещается в массив
	из указателей на строки
{
	char* current_path = (char*)malloc(sizeof(char)*(strlen(dir_name)+NAME_MAX));//Выделяется память под строку, в которую будет помещаться путь к файлам.Использвуется NAME_MAX, определяющий макс. длину имени файлаа
	strcpy(current_path,dir_name);

	DIR *current_dir = opendir(current_path);  //Открывается поток каталога и проверяется на корректное открытие. Далее с помощью функции readdir считывается структура с информацией о первом файле в каталоге.
	if(current_dir == NULL)
		return;
	dirent* file_in_current_dir = (dirent*)readdir(current_dir);

		while(file_in_current_dir)         //Цикл выполнятеется, пока в текущей директории будут объекты*/
	{
		int path_len = strlen(current_path);      //К строке, содержащий путь, добавляется имя объекта
			strcat(current_path, "/");
			strcat(current_path, file_in_current_dir->d_name);


			if                  //Проверяется,является ли объект файлом или директорией
			(
				file_in_current_dir->d_type == DT_REG &&
				strstr(file_in_current_dir->d_name, ".txt")!=NULL
			)
			{
				if((strs_in_file[*len] = (char*)read_file(current_path))!=NULL ) //Если объект - это непустой  txt файл,то возвращается указатель на первую строку из него
					(*len)++;
			}
		if
			(
```

```
                        file_in_current_dir->d_type == DT_DIR &&      // Если объект - это директория не родительская и не текущая, то рекурсивно вызывается функция read_catalog для прочтения данной директории
                        strcmp(".",file_in_current_dir->d_name)!=0 &&
                                strcmp("..",file_in_current_dir->d_name)!=0
                        )
                        {
                        read_catalog(current_path,strs_in_file, len);
                        }
                        current_path[path_len] = '\0';          //После проверки текущего объекта current_path "обрезается" до первоначального положения. Далее считывается следующий объект из данной директории
                        file_in_current_dir = (dirent*)readdir(current_dir);
                }
  closedir(current_dir);   //После обхода всех объектов в директории, она закрывается
}


int main()
{
    //Выделяется память под массив указателей на строки
        char** strs = (char**)malloc(sizeof(char*)*50);
        int len = 0;
        read_catalog(".",strs, &len); //Специальная функция обходит директории и заполняет массив из строк строками из файлов
           int i = 0;
                for (i; i < len; i++)
                printf("%s\n", strs[i]);
        qsort(strs, len, sizeof(char*),compare);  //Строки сортируются библиотечной функцией,компаратор для которой сравнивает числа в начале строк.Выводятся уже отсортированные строки.
           int j = 0;
                for (j; j < len; j++)
                printf("\n%s\n", strs[j]);
   int k = 0;
        for(k; k < len; k++)
        free(strs[k]);      //Освобождение памяти выделенной под каждую строку и под массив строк
        free(strs);
        return 0;
}
```