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: Строки. Рекурсия, циклы, обход дерева.

Студент гр. 2384 _____ Кузьминых Е.М

Преподаватель _____ Гаврилов А.В.

Санкт-Петербург
2023

Цель работы.

Написать программу для обхода директории на языке программирования Си, изучить работу с библиотекой dirent.h.

Задачи.

Вариант 3

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.

Выполнение работы.

Программа начинается с структуры *file_array,* которая хранит в себе массив строк *text,* а также количество строк в массиве *count* и размер буфера *buffer*.

Функция *check_directory* совершает рекурсивный обход директории и считывает строки из текстового файла, сохраняя их в массив *text*. На вход принимается строка с путем к директории и структуру *file_array*. С помощью функции *opendir()* открывается директория, если это удалось, то с помощью *readdir()* выбираем элемент из директории. Если это удалось, то проводится проверка на то, что выбранный элемент не является файлом системы создается строка buf, в которую сохраняется путь до файла с помощью *sprintf()*. Если в имени файла содержится расширение txt, то файл открывается с помощью *fopen(),* строка из файла считывается с помощью *fgets()* и сохраняется в массив text. После идет проверка размера буфера *buffer*. Если количество предложений

становится равно *buffer-y*, то он увеличивается и происходит перевыделение памяти с использованием *realloc(),* после файл закрывается.

Если выбранный элемент не содержит в своем названии расширения *txt,* то функция рекурсивно вызывает себя, предварительно добавив / к концу строки с путем к файлу.

После с помощью *readdir()* меняется элемент для прочтения в директории, а после выхода из цикла директория закрывается функцией *closedir().*

Функция comp сравнивает 2 строки по первому числу в строках. Функция *atoi()* считывает первое число в каждой строке, если два числа равны функция возвращает 0, если первое число больше второго, возвращает 1, иначе - -1.

Функция *sort()* вызывает *qsort()* для сортировки массива text по первому числу в строке. На вход функции передается массив строк из структуры file_array, количество строк. Строки сравниваются функцией comp.

Функция *print_in_file()* создает файл result.txt и функцией *fprintf()* отсортированные строки печатаются в созданный файл.

В функции *main()* создается структура *file_array* и выделяется под него память, вызываются функции *check_directory, sort, print_in_file*.

Тестирование.

№	Входные данные	Выходные данные	Комментарий
1	Директория с 2000 файлами расширения txt	Файл result.txt, хранящий в себе отсортированные строки из файлов директории	Ответ верный

Вывод.

В ходе выполнения лабораторной работы была написана программа, которая обходит директорию, считывает и сортирует строки из файлов в директории и сохраняет их в файл.

Приложение А.

Исходный код программы.

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


typedef struct file_array{
    char** text;
    int count;
    int buffer;
} file_array;

void check_directory(file_array* array, char* path){
    DIR *dir = opendir(path);
    if(dir){
        struct dirent * de = readdir(dir);
        while(de){
            if(strcmp(de->d_name,".")!=0        &&        strcmp(de-
>d_name,"..")!=0){
                char path_to_file[200];
                snprintf(path_to_file,200,"%s%s",path,de->d_name);
                if(strstr(de->d_name,".txt")!=NULL){
                    FILE *file = fopen(path_to_file,"r");
                    if(file){
                        char info[30];
                        fgets(info,30,file);
                        char * pos;
                        if((pos= strchr(info,'\n'))!=NULL){
                            *pos = '\0';
                        }
                        array->text[array->count]=     malloc(sizeof
(char)*(strlen(info)+1));
                        strncpy(array->text[array->count],     info,
strlen(info)+1);
```

```
                            array->count++;
                            if(array->count>=array->buffer){
                                array->buffer+=10;
                                array->text=  realloc(array->text,array-
>buffer*sizeof (char*));
                            }
                            fclose(file);
                        }
                    }
                    else{
                        strcat(path_to_file,"/");
                        check_directory(array,path_to_file);
                    }
                }
                de = readdir(dir);
            }

        }
        closedir(dir);
    }
    int comp(const void* s1, const void* s2){
        s1 = *(char** )s1;
        s2 = *(char **)s2;

        int num1 = atoi(s1);
        int num2 = atoi(s2);
        if(num1>num2){
            return 1;
        }
        else if(num1<num2){
            return -1;
        }
        else{
            return 0;
        }
    }
    void sort(file_array *array){
        qsort(array->text, array->count,sizeof(char*), comp);
```

```
}
void print_in_file(file_array * array){
    FILE * fl = fopen("result.txt","w");
    for (int i = 0; i < array->count; i++) {
        fprintf(fl,"%s\n",array->text[i]);
        free(array->text[i]);
    }
    fclose(fl);
}
int main(){
    int count =0;
    int buffer = 1;
    file_array *array = malloc(sizeof(file_array*));
    array->count=count;
    array->buffer=buffer;
    array->text = malloc(array->buffer*sizeof (char*));
    check_directory(array,"./");
    sort(array);
    print_in_file(array);
    return 0;
}
```