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: Обход файловой системы

Студенка гр. 0382 _____ Охотникова Г.С.

Преподаватель _____ Берленко Т.А.

Санкт-Петербург
2021

Цель работы.

Освоить рекурсивный обход файловой системы и основные функции по работе с файлами и директориями.

Задание.

Вариант 4:

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.

Основные теоретические положения.

Основные функции для работы с деревом файловой системы, объявления которых находятся в заголовочном файле *dirent.h* (также, может понадобиться включить заголовочный файл *sys/types.h*):

Для того, чтобы получить доступ к содержимому некоторой директории можно использовать функцию *DIR *opendir(const char *dirname),* которая возвращает указатель на объект типа *DIR,* с помощью которого можно из программы работать с заданной директорией. Тип *DIR* представляет собой поток содержимого директории.

Для того, что бы получить очередной элемент этого потока, используется функция *struct dirent *readdir(DIR *dirp)*. Она возвращает указатель на объект структуры *dirent,* в котором хранится информация о файле. Основной интерес представляют поля, хранящие имя и тип объекта в директории (это может быть не только "файл" и "папка").

После завершения работы с содержимым директории, необходимо вызвать функцию *int closedir(DIR *dirp),* Передав ей полученный функцией *readdir()* ранее дескриптор.

Выполнение работы.

В функции *main()* в переменную *path* записывается путь к нужному каталогу. В переменную *word*, память под которую выделена статически, считывается строка, которую должны образовывать пути файлов. Затем с помощью функции *fopen()* получаем указатель на файл, в который будет записан результат выполнения программы. В цикле, который ограничен длиной введенной строки, происходит вызов функции *print_path()*.

Функция *print_path*() в качестве аргументов получает путь к нужному каталогу, указатель на файл, в который будет записан результат, строку *word*, а также индекс элемента этой строки, с которым будет происходить сравнение. С помощью функции *opendir()* получаем доступ к первой записи каталога, в переменную *int len* сохраняем длину пути, а затем, если удалось получить доступ к каталогу, с помощью функции *readdir()* получаем очередной элемент директории. В цикле while, пока элементы директории не закончатся, проверяем: если имя файла *“.”*, то считываем следующий элемент и продолжаем. С помощью функции *strcat()* получаем путь к файлу, добавляя *“/”* и имя файла к переменной path. Затем, если тип элемента файл, то его путь записывается в *file*, если же нет и тип элемента директория, происходит рекурсивный вызов данной функции. Затем с помощью добавления *“\0”* путь “обрезается” до исходного, получаем следующий элемент директории. В конце директория закрывается с помощью функции *closedir()*.

После завершения цикла в функции *main()* с помощью функции *fclose()* закрываем файл, в который был записан результат.

Разработанный программный код см. в приложении А.

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии
1.	*HeLlO*	tmp/asdfgh/mkoipu/H.txt hello_world_test/qwerty /e.txt tmp/qwerty/qwert/L.txt tmp/asdfgh/l.txt tmp/asdfgh/O.txt	Программа работает верно.

Выводы.

Были изучены основные функции по работе с файлами и освоен обход файловой системы.

Разработана программа, выполняющая обход файловой системы и записывающая пути к нужным файлам.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: lb3.c

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

void print_path(char* path, FILE* file, char* word, int index) {
    DIR *dir = opendir(path);
    int len = (int)strlen(path);
    if (dir) {
        struct dirent *de = readdir(dir);
        while (de != NULL) {
            if(de->d_name[0] == '.'){
                de = readdir(dir);
                continue;
            }
            strcat(path,"/");
            strcat(path,de->d_name);
            if (de->d_type == 8) {
                if(de->d_name[0] == word[index] && de->d_name[1] ==
'.') {
                    fprintf(file, "%s\n", path);
                }
            }
            else if(de->d_type == 4){
                print_path(path, file, word, index);
            }
            path[len] = '\0';
            de = readdir(dir);
        }
    }
    closedir(dir);
}

int main() {
    char path[150] = "./tmp";
    int index;
    char word[100];
    fgets(word, 100, stdin);
    FILE *file = fopen("result.txt", "w");
    for (index = 0; index < strlen(word); index++) {
        print_path(path, file, word, index);
    }
    fclose(file);
    return 0;
}
```