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

Студент гр. 0382 Шангичев В. А.

Преподаватель Берленко Т. А.

Санкт-Петербург
2021

Цель работы.

Применить знания рекурсии и функций языка Си для обхода файловой системы.

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 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.

- Заголовочный файл `dirent.h` – файл, содержащий описание типов и функций для работы с файлами и директориями.
- Заголовочный файл `sys/types.h` – файл, определяющий типы языка Си для различных целей.
- `FILE` – тип, использующийся для работы с файлами.
- `fopen` – функция, использующаяся для открытия файла. На вход принимает путь к файлу и режим, в котором следует его открыть (“w”-запись, “r” – чтение и др.) Возвращает указатель на FILE.
- `fclose` – функция, использующаяся для закрытия файла. На вход принимает указатель, возвращенный функцией `fopen()`.
- `DIR` – тип, предназначенный для работы с директориями.
- `opendir` – функция, использующаяся для открытия директории. Возвращает указатель на `DIR`.
- `struct dirent` – структура, характеризующая файл/директорию. Основными ее полями являются `d_type` и `d_name` – первое поле определяет, файл ли это, а второе задает название.

- `readdir` – функция, возвращающая следующий файл из переданной ей директории. Возвращает указатель на `DIR`.

- `closedir` – функция, закрывающая директорию.

Выполнение работы.

В первых строках производится включение необходимых заголовочных файлов. В главной функции программы сначала считывается слово в переменную `word`, затем создается динамический двумерный массив. Длина первой оси данного массива определяется длиной переданного слова, которая вычисляется с помощью функции `strlen`. Данный двумерный массив `answer` будет использоваться для записи путей нужных файлов. При этом каждая строка в данном массиве будет соответствовать символу переданной строки на той же позиции. После этого вызывается функция `listDir`.

Функция `listDir` принимает на вход путь к директории, слово и массив, куда необходимо записывать нужные пути. В теле данной функции сначала создается массив `next`, туда с помощью функции `strcpy` переданный путь, и в конец добавляется символ “/”. Эта строка впоследствии будет использована для задания пути директорий, вложенных в текущую. В следующей строке происходит открытие директории. После этого с помощью функции `readdir` происходит извлечение очередного файла/директории. Перебор осуществляется с помощью цикла `while`. В теле цикла происходит проверка на то, является ли файл директорией. Если нет, значит файл может потенциально являться ответом. Для данной проверки была написана функция `check`.

Функция `check` принимает на вход слово, переданное программе и название файла, который необходимо проверить. Возвращает функция позицию, на которую надо поставить путь данного файла или -1, если файл не подходит. Для этого сначала проверяется, что название файла состоит из одного символа. Если это не так, то функция возвращает -1. Затем перебираются все символы переданного слова и названия файла. Если один из них совпадет, то функция вернет его индекс.

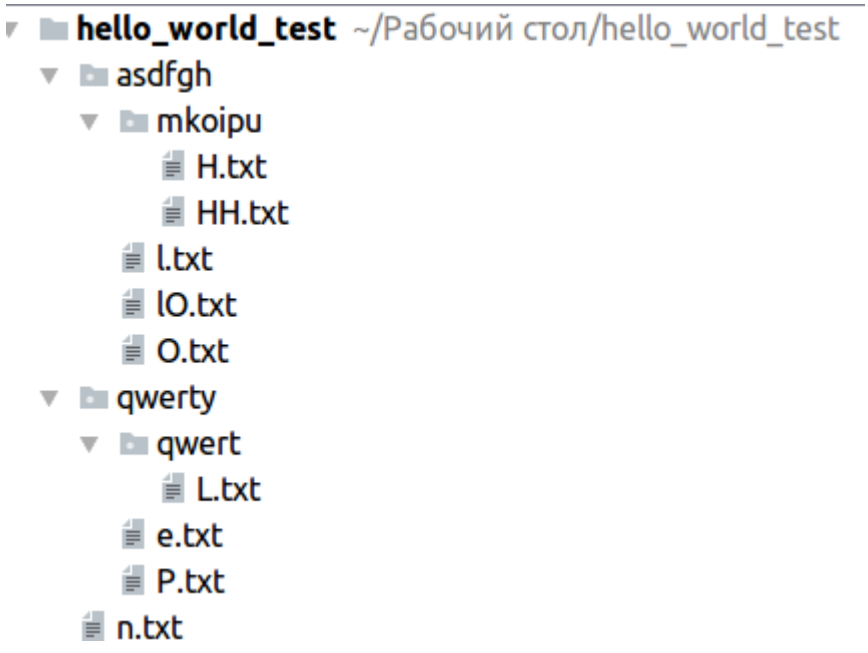
В случае получения положительного ответа функции `check`, функция `listDir` копирует содержимое массива `next` в строку с полученным индексом, а затем с помощью функции `strcat` конкатенирует с названием файла.

Если же была получена директория, не являющаяся текущей или предыдущей, что проверяется с помощью функции `strcmp`, то ее название присоединяется к текущему пути и подается на вход рекурсивного вызова. После окончания работы вызванной функции массив `next` принимает исходное значение путем добавления нулевого символа. После перебора всех файлов в директории она закрывается путем вызова функции `closedir`.

Ответ записывается с помощью функции `write`. Осуществляется создание файла с помощью открытия его в режиме записи. Туда через знак переноса строки записываются все полученные строки, после чего файл закрывается.

Тестирование.

Файловая структура:



Переданное слово:

HeLlO

result.txt:

hello_world_test/asdfgh/mkoipu/H.txt

hello_world_test/qwerty/e.txt
hello_world_test/qwerty/qwert/L.txt
hello_world_test/asdfgh/l.txt
hello_world_test/asdfgh/O.txt

Заключение:

Программа работает верно.

Выводы.

Были применены знания рекурсии и языка Си для обхода файловой системы. Была написана программа, подбирающая каждому символу из переданной строки соответствующий файл с указанием полного пути до него.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

main.c

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

void write(char** answer, int len){
    int i;
    FILE* fp;
    fp = fopen("result.txt", "w");
    for (i = 0; i < len; i++){
        fprintf(fp, "%s\n", answer[i]);
    }
    fclose(fp);
}

int check(char* name, const char* word){
    /*
    проверяет, является ли название файла
    ответом. если да, то возвращает нужный
    индекс.
    */
    if (name[1] != '.'){
        return -1;
    }
    int i, len = strlen(word);
    for (i = 0; i < len; i++){
        if (name[0] == word[i]){
            return i;
        }
    }
    return -1;
}

void listDir(const char* path, const char* word, char** answer){
    // готовим следующий путь
    char next[150];
    strcpy(next, path);
    strcat(next, "/");

    // открываем текущую директорию и
    // перебираем файлы и папки
    DIR* dir = opendir(path);
    struct dirent* de = readdir(dir);
    while (de){
        if (de->d_type == DT_REG){
            // если это файл, то проверяем, не
            // является ли он частью ответа
```

```
            int answer_i = check(de->d_name, word);
            if (answer_i != -1){
                strcpy(answer[answer_i], next);
                strcat(answer[answer_i], de->d_name);
            }
        }
        if (de->d_type == DT_DIR && strcmp(de->d_name, ".") \
            && strcmp(de->d_name, "..")){
            int len = strlen(next);
            strcat(next, de->d_name);
            listDir(next, word, answer);
            next[len] = '\0';
        }
        de = readdir(dir);
    }
    closedir(dir);
}

int main(){
    char word[50];
    scanf("%s", word);
    int i, len = strlen(word);
    char** answer = malloc(sizeof(char*) * len);
    for (i = 0; i < len; i++){
        answer[i] = malloc(sizeof(char) * 100);
        answer[i][0] = '\0';
    }
    listDir("./tmp", word, answer);
    write(answer, len);
    for (i = 0; i < len; i++){
        free(answer[i]);
    }
    free(answer);
    return 0;
}
```