# 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: Рекурсия, циклы, рекурсивный обход файловой системы

Студентка гр. 3341	Яковлева А.А.
Преподаватель	Глазунов С.А.

Санкт-Петербург

2024

# Цель работы

Целью работы является освоение работы с рекурсивными функциями и файловой системой, а также ее рекурсивным обходом.

Для достижения поставленной цели требуется решить следующие задачи:

- 1) Ознакомиться с понятием рекурсии;
- 2) Освоить написание рекурсивных функций в языке Си;
- 3) Изучить работу с файловой системой в языке Си;
- 4) Написать программу для рекурсивного обхода всех файлов в папке, в том числе во вложенных папках.

#### 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.

- ! Регистрозависимость
- ! Могут встречаться файлы, в имени которых есть несколько букв и эти файлы использовать нельзя.
  - ! Одна буква может встречаться один раз.

Ваше решение должно находиться в директории /home/box, файл с решением должен называться solution.c. Результат работы программы должен быть записан в файл result.txt. Ваша программа должна обрабатывать директорию, которая называется tmp.

#### Выполнение работы

Макросы:

- *START\_DIR* название директории, которую обрабатывает программа
- *NAME\_RESULT\_FILE* имя файла, в который будет записан результат работы программы
- *FILE\_NAME\_TEMPLATE* шаблон имени файла ".txt", т.к. требуется найти файлы данного вида
- *END LINE* символ переноса строки
- MAX STR LENGTH максимальная длина считанной строки
- *MAX\_FILE\_NAME\_LENGTH* максимальная длина имени файла (длина всегда равна 6, т.к. ищем файлы вида " .txt")
- *CURRENT\_DIR* ссылка на текущую директорию "."
- *PARENT\_DIR* ссылка на родительскую директорию ".." Функции:
- *char \*path\_cat(const char \*path1, const char \*path2)* принимает два указателя на две части пути, выделяет память размера длин двух строк + 2(для символов '/' и '\0'), функцией *sprintf* создаёт строку, состоящую из первого пути, символа '/' и второго пути, возвращает указатель на полученную строку.
- *int is\_not\_curr\_or\_parent\_dir(const char \*dir\_name)* принимает название директории. Если это не ссылка на текущую или родительскую директорию возвращает 1, иначе возвращает 0.
- void search\_full\_path\_file(const char \*dir\_name, char\* name\_required\_file, FILE \*result\_file) принимает указатель на название директории, название искомого файла и на файл с результатом работы программы. Сначала с помощью функции opendir открывает директорию dir\_name, циклом while проходит по всем элементам из потока директории. Если тип файла данного элемента DT\_REG, т.е. это обычный файл, и имя файла совпадает с искомым, то то в result\_file функцией fputs записывается полный путь к файлу, полученный функцией path\_cat, и END\_LINE. Иначе если тип файла данного элемента DT\_DIR, т.е. это директория, и это не ссылка на текущую или родительскую директорию, то функцей path\_cat

конкатенируются имена родительской и вложенной директории, далее рекурсивного вызывается функция для обработки вложенной директории. После прохода по всем элементам директории её поток закрывается функцией *closedir*.

• *main*. Сначала создаётся строка *name\_required\_file*, в которую функцией *strcpy* копируется шаблон искомого файла *FILE\_NAME\_TEMPLATE*. Функцией *fopen* открывается файл *NAME\_RESULT\_FILE* на запись. Далее создаётся и считывается строка *str\_containing\_names\_of\_files*. Затем циклом *for* проходим по всем символом считанной строки, для получения имени искомого файла первый символ *name\_required\_file* заменяем текущим символом и вызываем функцию *search\_full\_path\_file*. После цикла закрываем файловый поток функцией *fclose*.

Разработанный программный код см. в приложении А.

# Тестирование

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные	Выходные данные	Комментарии
	данные		
1.	HeLlO	tmp/asdfgh/mkoipu/H.txt	Выводятся полные пути
		tmp/qwerty/e.txt	файлов H.txt, e.txt, L.txt,
		tmp/qwerty/qwert/L.txt	l.txt, O.txt.
		tmp/asdfgh/l.txt	
		tmp/asdfgh/O.txt	
2.	123	tmp/1.txt	Выводятся полные пути
		tmp/third/fourth/2.txt	файлов 1.txt, 2.txt, 3.txt.
		tmp/first/sec/3.txt	

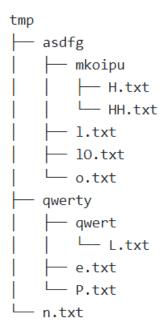


Рисунок 1 – расположение файлов в 1 тесте

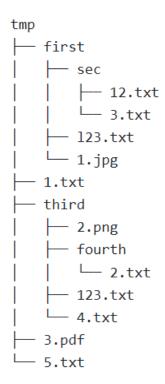


Рисунок 2 – расположение файлов во 2 тесте

# Выводы

В ходе выполнения работы были решены следующие задачи:

- изучены рекурсивные функции и их написание в языке Си;
- изучена работа с файловой системой в языке Си;
- написана программа для рекурсивного обхода всех файлов в папке, в том числе во вложенных папках, и поиска нужного файла.

#### ПРИЛОЖЕНИЕ А

# ИСХОДНЫЙ КОД ПРОГРАММЫ

### Название файла: solution.c

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <dirent.h>
#define START DIR "./tmp"
#define NAME RESULT FILE "result.txt"
#define FILE NAME TEMPLATE " .txt"
#define END LINE "\n"
#define MAX STR LENGTH 100
#define MAX FILE NAME LENGTH 6
#define CURRENT DIR "."
#define PARENT DIR ".."
char *path cat(const char *path1, const char *path2)
    int res path len = strlen(path1) + strlen(path2) + 2;
   char *res path = (char*) malloc(res path len * sizeof(char));
    sprintf(res path, "%s/%s", path1, path2);
   return res path;
}
int is not curr or parent dir(const char *dir name)
    if(strcmp(dir name, CURRENT DIR) && strcmp(dir name, PARENT DIR))
return 1;
   return 0;
void search full path file (const char *dir name, char* name required file,
FILE *result file)
   DIR *dir = opendir(dir name);
   struct dirent *de = readdir(dir);
   while (de)
        if(de->d type == DT REG && !strcmp(de->d name, name required file))
            fputs(path cat(dir name, de->d name), result file);
            fputs(END LINE, result file);
        }
                             (de->d type ==
                                                        DT DIR
        else
                  if
                                                                       & &
is not curr or parent dir(de->d name))
            char *new dir = path cat(dir name, de->d name);
            search full path file (new dir,
                                                     name required file,
result file);
           free (new dir);
        de = readdir(dir);
    closedir(dir);
```

```
int main()
{
    char*    name_required_file = strcpy(name_required_file,
FILE_NAME_TEMPLATE);
    FILE *result_file = fopen(NAME_RESULT_FILE, "w");
    char str_containing_names_of_files[MAX_STR_LENGTH];
    fgets(str_containing_names_of_files, MAX_STR_LENGTH, stdin);

    for(int i = 0; i < strlen(str_containing_names_of_files); i++)
    {
        name_required_file[0] = str_containing_names_of_files[i];
        search_full_path_file(START_DIR, name_required_file, result_file);
    }
    free(name_required_file);
    fclose(result_file);
    return 0;
}</pre>
```