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. 3342 _____ Пушко К.Д.

Преподаватель _____ Глазунов С.А.

Санкт-Петербург
2024

Цель работы

Изучить принцип работы с рекурсивными алгоритмами и файловой системой на языке программирования Си.

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.

Особенности:

Регистрозавсисмость;

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

В начале работы программа считывает искомое слово. Далее идет выделение памяти для количества строк, равное длине слова. После этого программа начинает рекурсивно проходить по всему файловому дереву. Если программа встретила папку, то цикл рекурсии начинается сначала, но с текущего места, если файл, то идет проверка на удовлетворение условиям.

Если название файла и его расширение соответствуют нужным нам, то путь к этому файлу записывается в массив строк под индексом, равному индексу символа, который необходимо найти.

После того как программа прошла по всем файлам, результат построчно записывается в текстовый документ. Далее идет закрытие потока текстового документа и очистка выделенной памяти.

Разработанный программный код см. в приложении А.

Выводы

Был изучены библиотека dirent.h, принцип работы с рекурсивными алгоритмами и файловой системой на языке программирования Си. Написана программа, находящая пути до файлов, которые образуют слово, записывающая результат работы в файл result.txt.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.c

```
#include <stdio.h>
#include <string.h>
#include <dirent.h>
#include <stdlib.h>
#include <regex.h>

#define MAX_PATH 256
#define MAX_FILENAME 128


int dirValidator(char* dirname){
    int res;

    if(strcmp(dirname, ".") && strcmp(dirname, "..")){
        res = 1;
    }else{
        res = 0;
    }

    return res;
}

void dirLookup(char* root,char* findWord,char** outputPaths)
{
    char tmpDir[MAX_PATH];
    strncpy(tmpDir, root, MAX_FILENAME);

    DIR* rootDir = opendir(tmpDir);
    if(rootDir == NULL){
        return;
    }

    struct dirent* dir = readdir(rootDir);

    while(dir){
        if(dir->d_type == DT_REG){
            if(strlen(dir->d_name) > MAX_FILENAME){
                perror("FileName size error");
                exit(0);
            }

            strncat(tmpDir, "/", strlen(tmpDir)+1);
            strncat(tmpDir, dir->d_name, strlen(dir->d_name));

            //readFile(tmpDir);
            regex_t regex;
            regmatch_t groups[4];
            char* pattern = "^.\\.txt";
            regcomp(&regex, pattern, REG_EXTENDED);
            if(regexec(&regex,dir->d_name,4,groups,0)==0)
```

```
                {

                    for (int i = 0; i < strlen(findWord); ++i)
                    {
                        if(strncmp(&findWord[i], &dir->d_name[0],1) == 0)
                        {
                            strncat(outputPaths[i],tmpDir,MAX_PATH);
                            break;
                        }
                    }
                }

                tmpDir[strlen(tmpDir) - 1 - strlen(dir->d_name)] = '\0';

            }else       if(dir->d_type       ==       DT_DIR       &&
dirValidator(dir->d_name)){
                if(strlen(dir->d_name) > MAX_FILENAME){
                    perror("FileName size error");
                    exit(0);
                }

                strncat(tmpDir, "/", strlen(tmpDir)+1);
                strncat(tmpDir, dir->d_name, strlen(dir->d_name));


                dirLookup(tmpDir,findWord,outputPaths);
                tmpDir[strlen(tmpDir) - 1 - strlen(dir->d_name)] = '\0';
            }
            dir = readdir(rootDir);
        }
        closedir(rootDir);
    }

    int main()
    {
        char* findWord = (char*)malloc(1000 * sizeof(char));
        if (findWord == NULL)
        {
            exit(1);
        }
        scanf("%s",findWord);



        char**              outputPaths             =             (char
**)malloc(strlen(findWord)*sizeof(char*));
        if (outputPaths == NULL)
        {
            exit(1);
        }
        for (int i = 0; i < strlen(findWord); ++i) {
            outputPaths[i] = malloc(MAX_FILENAME*sizeof(char));
            if (outputPaths[i] == NULL)
        {
            exit(1);
        }
        }
```

```
    FILE* file = fopen("result.txt","w");



    dirLookup("./tmp",findWord,outputPaths);

    if(file)
    {
        for (int i = 0; i < strlen(findWord); ++i)
        {
            fprintf(file,"%s\n",outputPaths[i]);
        }
        fclose(file);
    }

    for (int i = 0; i < strlen(findWord); ++i) {
        free(outputPaths[i]);
    }
    free(outputPaths);
    free(findWord);
    return 0;
}
```