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 Гудов Н.Р.

Преподаватель Берленко Т.А.

Санкт-Петербург
2021

Цель работы.

Изучение принципов работы с деректориями на ОС Linux на языке Си.

Задание.

Вариант 1

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 .txt. Требуется найти файл, который содержит строку "Minotaur" (файл-минотавр). Файл, с которого следует начинать поиск, всегда называется file.txt (но полный путь к нему неизвестен). Каждый текстовый файл, кроме искомого, может содержать в себе ссылку на название другого файла (эта ссылка не содержит пути к файлу). Таких ссылок может быть несколько.

Основные теоретические положения.

В программировании рекурсия — вызов функции (процедуры) из неё же самой, непосредственно (простая рекурсия) или через другие функции (сложная или косвенная рекурсия), например, функция A вызывает функцию B, а функция B — функцию A. Количество вложенных вызовов функции или процедуры называется глубиной рекурсии. Рекурсивная программа позволяет описать повторяющееся или даже потенциально бесконечное вычисление, причём без явных повторений частей программы и использования циклов.

Выполнение работы.

В функции main() определяется начальное положение для поиска слова, создаются массивы символов для записи текущей деректории/файла и пути. Далее вызывается функция *RARR(),* которая пишет текущий к текущей деректории. После-*Minotavr()*-функция, проверяющая дальнейшие пути к файлу. Если на данном шаге не достигнут результат или тупик, функция вызывается повторно. В это время в функции *RARR()* пишется путь к текущей успешной

папке-выполняется рекурсивно. Помимо стандартной библиотеки языка, были использованы dirent.h и sys/stat.h.

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии

1.	Test 1	`./root/add/add/file.txt`
		`./root/add/mul/add/file4.txt`
		`./root/add/mul/file2.txt`
		`./root/add/mul/file3.txt`
			Верно

Выводы.

В ходе лабораторной работы изучены принципы работы с деректориями на ОС Linux на языке Си.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.c

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

int RARR( char *labirint, char** names, char** ways, int* count);

int Minotavr(char* name, char** names, char** ways, int count, char**
result, int* _count);

int main(){

    int count=0;
    int _count = 0;

    char root[]="./labyrinth";
    char res[]="result.txt";

    char* names[5000];
    char* ways[5000];
    char* result[5000];
    if (RARR( root, names, ways, &count) == 0)
        Minotavr("file.txt", names, ways, count, result, &_count);
    FILE * file = fopen(res, "w");
    while(_count!=0){
        fputs(result[_count-1], file);
        fputs("\n", file);
        printf("%s\n",result[_count-1]);
        _count--;
    }

    fclose (file);
    return 0;
}

int RARR( char *labirint, char** names, char** ways, int* count){
    char wway[PATH_MAX + 1];
    DIR *dir = NULL;
    struct dirent *indir = NULL;
    dir = opendir(labirint);
    if(dir==NULL){ return -1;}
    indir=readdir(dir);
    while(indir != NULL){
        struct stat inf;
        if((strncmp(  indir->d_name,  ".",  PATH_MAX)  ==  0)  ||
(strncmp( indir->d_name, "..", PATH_MAX) == 0)){
        indir = readdir(dir);
        continue;
        }
        strncpy(wway, labirint, PATH_MAX);
        strncat(wway, "/", PATH_MAX);
```

```
            strncat(wway, indir->d_name, PATH_MAX);
            if(lstat(wway, &inf)== 0){
                if(S_ISDIR(inf.st_mode)){
                    RARR(wway, names, ways, count);
                }
                else
                if(S_ISREG(inf.st_mode)){
                names[*count] = malloc(300*sizeof(char));
                (names[*count])[0]='\0';
                ways[*count] = malloc(300*sizeof(char));
                (ways[*count])[0]='\0';

                    strcat(names[*count], indir->d_name);
                    strcat(ways[*count], labirint);

                    (*count)+=1;
                }
            }
            indir=readdir( dir );
        }

        closedir(dir);
        return 0;
    }

    int Minotavr(char* name, char** names, char** ways, int count, char**
result, int* _count){
        int i;
        for(i=0; i<count;i++){if (strcmp(names[i], name)==0) break;}
        char* way[300];
        way[0]='\0';
        strcat(way, ways[i]);
        strcat(way, "/");
        strcat(way, names[i]);

        FILE * file = fopen(way, "r");
        char str[300];
        str[0]='\0';
        char fname[300];
        char a[]="Minotaur";
        char b[]="Deadlock";
        char c[]="@include";
        fname[0]='\0';

        while(fscanf(file, "%s", str) != EOF){
            if(strcmp(str, c)==0){
                fscanf(file, "%s\n", fname);
                if (Minotavr(fname, names, ways, count, result, _count)){
                result[*_count] = malloc(300*sizeof(char));
                (result[*_count])[0] = '\0';
                strcat(result[*_count], way);
                (*_count)++;
                fclose (file);
                return 1;
                }
            }
            else
            if(strcmp(str, a)==0){
```

```
            result[*_count] = malloc(300*sizeof(char));
            (result[*_count])[0] = '\0';
            strcat(result[*_count], way);
            (*_count)++;
            fclose (file);
            return 1;
        }
        else
        if(strcmp(str, b)==0){
            fclose (file);
            return 0;
        }
    }
    fclose (file);
    return 0;
}
```