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

Студентка гр. 1304 Виноградова М.О.

Преподаватель Чайка К.В.

Санкт-Петербург
2022

Цель работы.

Написать программу в соответствии с условием задачи.

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 .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.*Цепочка, приводящая к файлу-минотавру может быть только одна.*
Общее количество файлов в каталоге не может быть больше 3000. Циклических зависимостей быть не может. Файлы не могут иметь одинаковые имена.
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 **labyrinth**.

Выполнение работы.

Функции:

Int isValid – принимает имя файла и проверяет его имя на соответсвие.

Void readFile – принимает имя файла, количество файлов прочитанных до него(без тупиков), массив с их именами. Функция открывает файл, если он привел в тупик (Deadlock), то его имя не записывается в массив с решением, если он содержит искомое содержимое (Minotaur), то решение записывается в файл result. Если файл сожержит имена других файлов, то они поочередно передаются в функцию listDir.

Void listDir принимает имя начальной папки, искомый файл, количество количество файлов прочитанных до него, массив с их именами. Функция ищет искомый файл по всем папкам, когда она его находит, то вызывается функция readFile.

Struct Pair – структура для хранения строк вида @include name.txt

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.	f1:
	  f2:
	  test-0021.txt
	  test-0022.txt
	  test-0012.txt
	test-0001.txt
	test-0002.txt
	
	test-0001:
	@include test-0002.txt
	@include test-0021.txt
	@include test-0022.txt
	test-0002 и test-0022:
	Deadlock
	test-0021:
	@include test-0012.txt
	test-0012:
	Minotaur
		./test-0001.txt
		./f1/f2/test-0021.txt
		./f1/test-0012.txt
			Программа работает корректно

Вывод.

В соответствии с условием задачи была реализована программа.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.c

```
#include <stdio.h>
#include <string.h>
#include <dirent.h>
#include <sys/types.h>
#include <regex.h>
struct Pair{
   char first[100];
   char sec[100];
};

void listDir(char *startDir, char * need,int num_ans,char answer[100][3000]);
int isValid(char *filename){

   char *regexp = "\\.txt$";
   regex_t regexComp;

   if(regcomp(&regexComp, regexp, REG_EXTENDED)){
      return 0;
```

```
    }

    return regexec(&regexComp, filename, 0, NULL, 0) == 0;
}

void readFile(char * file,int num_ans,char answer[50][3000]){


    struct Pair mas[10];
    FILE *fp=fopen(file,"r");
    FILE *f_wr=fopen("result.txt","w");
    char s[100];

    int kol=0;
    char arr_sent[100][100];
    if(fp==NULL)return;

  while(fgets(s,100,fp)){

        strcpy(arr_sent[kol],s);

        kol++;
    }


    char *istr;
    istr = strtok (arr_sent[0]," \n");
    strcpy(mas[0].first,istr);


    strcpy(answer[num_ans],file);

    num_ans++;

    if(kol==1&&(strcmp(istr,"Deadlock")==0 || strcmp(istr,"Minotaur")==0)){


        if(strcmp(istr,"Deadlock")==0) {

            num_ans--;
        }
        if(strcmp(istr,"Minotaur")==0){

                int q;


            for(q=0;q<num_ans;q++){
                fprintf(f_wr,"%s\n",answer[q]);
```

```
            }

        }
        fclose(fp);
        return;
    }


    istr = strtok(NULL, " \n");
    strcpy(mas[0].sec, istr);
    int i=0;
    for(i=1;i<kol;i++){

        istr = strtok (arr_sent[i]," \n");
        strcpy(mas[i].first,istr);
        istr = strtok(NULL, " \n");
        strcpy(mas[i].sec, istr);

    }
    for( i=0;i<kol;i++){
        //printf("%s %s\n",mas[i].first,mas[i].sec);
        char dir_name[500];

        strcpy(dir_name,"./labyrinth");
        listDir(dir_name,mas[i].sec,num_ans,answer);

    }
    fclose(fp);

}
void listDir(char *startDir, char * need,int num_ans,char answer[100][3000]){
    char nextDir[200]={0};
    strcpy(nextDir, startDir);
    DIR *dir = opendir(startDir);
    if(!dir)
        return;
    struct dirent *de = readdir(dir);
    while(de){
        if(strcmp(de->d_name, ".") != 0 && strcmp(de->d_name, "..") != 0){
            int len = strlen(nextDir);
                    strcat(nextDir, "/");
            strcat(nextDir,de->d_name);
            if(strcmp(de->d_name,need)==0){

                readFile(nextDir,num_ans,answer);

                return;
```

```
            }
            listDir(nextDir, need,num_ans,answer);
            nextDir[len] = '\0';
        }
        if(/*de->d_type == DT_REG &&*/ isValid(de->d_name)){
            int len = strlen(nextDir);
            strcat(nextDir, "/");
            strcat(nextDir,de->d_name);
            //printFile(nextDir, pad);
            nextDir[len] = '\0';
        }
        de = readdir(dir);
    }
    closedir(dir);

}

int main(){
        char dir_name[500];
        strcpy(dir_name,"./labyrinth");
        int num_ans=0;
        char answer[100][3000];
    listDir(dir_name, "file.txt",num_ans,answer);


    return 0;
}
```