Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра программного обеспечения информационных технологий

Дисциплина: Архитектура компьютерной техники и операционных систем (АКТиОС)

ОТЧЕТ

по лабораторной работе 5

Выполнил

студент: гр. 151004 Глушаченко Н.С.

Проверил: Леванцевич В.А.

Минск 2022

**Вариант 1**

Написать скрипт для поиска файлов заданного размера в заданном каталоге (имя каталога задаётся пользователем в качестве третьего аргумента командной строки). Диапазон (мин.- мах.) размеров файлов задаётся пользователем в качестве первого и второго аргумента командной строки. Вывести на консоль первые найденные 20 файлов в виде: полный путь, имя файла, его размер.

#!/bin/bash

find $3 -type f -size +$1 -size -$2 -printf "PATH: %h SIZE: %s FILE NAME: %f\n" | sed 1,20p

Отсортировать в заданном каталоге (аргумент 1 командной строки) и во всех его подкаталогах файлы по следующим критериям (аргумент 2 командной строки, задаётся в виде целого числа):1 – по размеру файла, 2 – по имени файла. Записать без сохранения структуры каталогов отсортированные файлы общим списком, в новый каталог (аргумент 3 командной строки

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include <dirent.h>

#include <sys/stat.h>

#define ERROR (-1)

#define SRC 1

#define SORT 2

#define DST 3

#define ARGUMENTS 4

#define SORT\_BY\_SIZE 1

#define SORT\_BY\_ALPHABET 2

typedef struct {

char path[PATH\_MAX];

char name[FILENAME\_MAX];

int size;

} file\_t;

file\_t \*f\_list;

int f\_list\_len = 0;

int cmpName(file\_t file1, file\_t file2) {

return (strcmp(file1.name, file2.name) > 0);

}

int cmpSize(file\_t file1, file\_t file2) {

return ((file1.size - file2.size) < 0);

}

char\* slashAdd(char\* path) {

if (path[strlen(path) - 1] != '/')

strcat(path, "/");

return path;

}

int dirPass(const char \*dir\_name) {

DIR \*directory;

struct dirent \*dir\_item;

directory = opendir(dir\_name);

while((dir\_item = readdir(directory)) != NULL) {

char next\_item[PATH\_MAX];

strcpy(next\_item, dir\_name);

strcat(next\_item, dir\_item->d\_name);

if (strcmp(".", dir\_item->d\_name) != 0 && strcmp("..", dir\_item->d\_name) != 0) {

struct stat statBuf;

lstat(next\_item, &statBuf);

if (S\_ISDIR(statBuf.st\_mode)) {

dirPass(slashAdd(next\_item));

} else if (S\_ISREG(statBuf.st\_mode)) {

file\_t file\_tmp;

strcpy(file\_tmp.name, dir\_item->d\_name);

strcpy(file\_tmp.path, next\_item);

file\_tmp.size = (int) statBuf.st\_size;

f\_list = realloc(f\_list, (++f\_list\_len) \* sizeof(file\_t));

f\_list[f\_list\_len - 1] = file\_tmp;

}

}

}

closedir(directory);

}

void sort(int cmpFunc(file\_t, file\_t)) {

for (int i = 0; i < f\_list\_len; i++) {

for (int j = 0; j < f\_list\_len - i - 1; j++) {

if (cmpFunc(f\_list[j], f\_list[j + 1])) {

file\_t file\_tmp = f\_list[j];

f\_list[j] = f\_list[j + 1];

f\_list[j + 1] = file\_tmp;

}

}

}

}

int main(int argc, char const \*argv[]) {

char dest\_file[PATH\_MAX + 1], dest\_path[PATH\_MAX], srcDir[PATH\_MAX];

int sort\_opt = atoi(argv[SORT]);

realpath(argv[DST], dest\_path);

slashAdd(dest\_path);

if (argc != ARGUMENTS) {

printf("ERROR: Wrong number of arguments\n");

return ERROR;

}

if (sort\_opt != SORT\_BY\_SIZE && sort\_opt != SORT\_BY\_ALPHABET) {

printf("ERROR: Wrong sort option\n");

return ERROR;

}

strcpy(srcDir, argv[SRC]);

dirPass(slashAdd(srcDir));

if (sort\_opt == SORT\_BY\_ALPHABET)

sort(cmpName);

else

sort(cmpSize);

mkdir(dest\_path, 0777);

for (int i = 0; i < f\_list\_len; i++) {

strcpy(dest\_file, dest\_path);

strcat(dest\_file, f\_list[i].name);

if (link(f\_list[i].path, dest\_file) == ERROR)

printf("File already exists %s\n", f\_list[i].name);

else

printf("ADDED\n FILE: %s\n SIZE: %d\n", f\_list[i].name, f\_list[i].size);

}

return 0;

}

