Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИУ, Информатика и системы управления

КАФЕДРА ИУ7, Программное обеспечение ЭВМ и информационные технологии

ОТЧЕТ
к лабораторной работе №2

по курсу

“Операционные системы”
На тему: “Файлы и каталоги”

Студент ИУ7-64Б _____ **А.А. Андреев**
(Группа) (Подпись, дата) (И.О.Фамилия)

Преподаватель _____ **Н.Ю. Рязанова**
(Подпись, дата) (И.О.Фамилия)

2022 г.

Оглавление

Задание

1. Структурировать исходный код программы в листинге 4.7.

2. Изменить программу так, чтобы она выводила на экран дерево каталогов в наглядной форме при помощи символов *, ../, | и тд.

3. Изменить функцию myfwt() так, чтобы она передавала функции lstat() не полный путь к файлу, а только его имя каждый раз, когда встречается каталог.

Реализация

В листингах 1-3 приведен код программы.

Листинг 1

```
1. #include <stdio.h>
2. #include <dirent.h>   //opendir()/readdir()/closedir()
3. #include <sys/stat.h>
4. #include <string.h>
5. #include <unistd.h>
6. #include <errno.h>
7.
8. #define OK 0
9. #define ERROR -1
10.   #define ERROR_STAT 1
11.   #define ERROR_OPEN_DIR 2
12.
13.   #define START_DEPTH 0
14.
15.   // Рекурсивный проход по пути, его вывод и возврат ошибки
16.   static int dopath(const char *filename, int
   recursion_depth);
17.
18.   // Проверка наличия аргумента пути
19.   int is_dir_in_arg(int argc);
20.
21.   int main(int argc, char *argv[])
22.   {
23.      int error_status;
24.
25.      // Параметр отсутствует
26.      if (is_dir_in_arg(argc) == OK)
27.           error_status = dopath(argv[1], START_DEPTH);     //
   начиная с папки в параметре
28.      else
29.           error_status = dopath("./", 0); // начиная с текущей
   папки
30.
31.      return error_status;
32.   }
33.
34.   int is_dir_in_arg(int argc)
35.   {
36.      if (argc == 2)
37.           return OK;
38.      return ERROR;
39.   }
40.
41.   static int dopath(const char *filename, int
   recursion_depth)
42.   {
43.      struct stat statbuf;
44.      struct dirent *dirp;
45.      DIR *dp;
```

```
46.
```

Листинг 2

```
47.
48.      //'.' - указатель на сам каталог
49.      //'..' - указатель на родительский каталог
50.      if (strcmp(filename, ".") == 0 || strcmp(filename, "..")
   == 0)
51.            return OK;
52.
53.      //Вызов lstat() идентичен stat(), но в случае, если
   filename является символьной ссылкой, то возвращается
   информация о самой ссылке, а не о файле, на который она
   указывает; возвращает информацию об указанном файле
54.      if (lstat(filename, &statbuf) < 0) {
55.            switch(errno)
56.            {
57.                  case EBADF:
58.                        printf("Неверный описатель файлового
   дескриптора.");
59.                        break;
60.                  case ENOENT:
61.                        printf("Компонент полного имени файла
   filename не существует или полное имя является пустой
   строкой.");
62.                        break;
63.                  case ENOTDIR:
64.                        printf("Компонент пути не является
   каталогом. ");
65.                        break;
66.                  case ELOOP:
67.                        printf("При поиске файла встретилась
   символьная ссылка.");
68.                        break;
69.                  case EFAULT:
70.                        printf("Некорректный адрес. ");
71.                        break;
72.                  case EACCES:
73.                        printf("Запрещен доступ. ");
74.                        break;
75.                  case ENOMEM:
76.                        printf("Недостаточно памяти в
   системе.");
77.                        break;
78.                  case ENAMETOOLONG:
79.                        printf("Слишком длинное название файла.
   ");
80.                        break;
81.                  case EOVERFLOW:
82.                        printf("Некоторые значения были слишком
   большими, чтобы быть представленными в возвращаемой
   структуре.s");
83.                        break;
84.            }
```

```
85.
```

Листинг 3

```
86.   printf("Ошибка вызова функции stat");
87.           return ERROR_STAT;
88.
89.       }
90.
91.       //печать отступов
92.       for (int i = 0; i < recursion_depth; ++i)
93.           printf("       |");
94.
95.
96.       /* печатаем имя файла */
97.       // s_isdir проверяет, является ли данный файл каталогом
98.       if (S_ISDIR(statbuf.st_mode) == 0) {
99.           printf("%s\n", filename);
100.          return OK;
101.      }
102.
103.      /* каталог */
104.      printf("%s\n", filename);
105.      if ((dp = opendir(filename)) == NULL) {
106.          printf("couldn't open directory '%s'\n", filename);
107.          return ERROR_OPEN_DIR;
108.      }
109.
110.      // chdir() изменяет текущий рабочий каталог вызывающего
      процесса на каталог, указанный в path; PATH — переменная
      окружения, представляющая собой набор каталогов, в которых
      расположены исполняемые файлы; нужен для коротких имен
111.
112.      chdir(filename);
113.
114.      // для каждого элемента каталога
115.      // Функция readdir() возвращает указатель на следующую
      запись каталога в структуре dirent, прочитанную из потока
      каталога. Каталог указан в dir. Функция возвращает NULL по
      достижении последней записи или если была обнаружена ошибка.
116.      while ((dirp = readdir(dp)) != NULL)
117.          dopath(dirp -> d_name, recursion_depth + 1);
118.      chdir("..");
119.
120.      //закрывает поток каталога и освобождает ресурсы,
      выделенные ему.
121.      //Она возвращает 0 в случае успеха и -1 при наличии
      ошибки.
122.      closedir(dp);
123.
124.      return OK;
125.  }
126.
```

Результат работы программы

На рисунках 1 и 2 приведен результат работа программы с аргументом пути к директории и без.

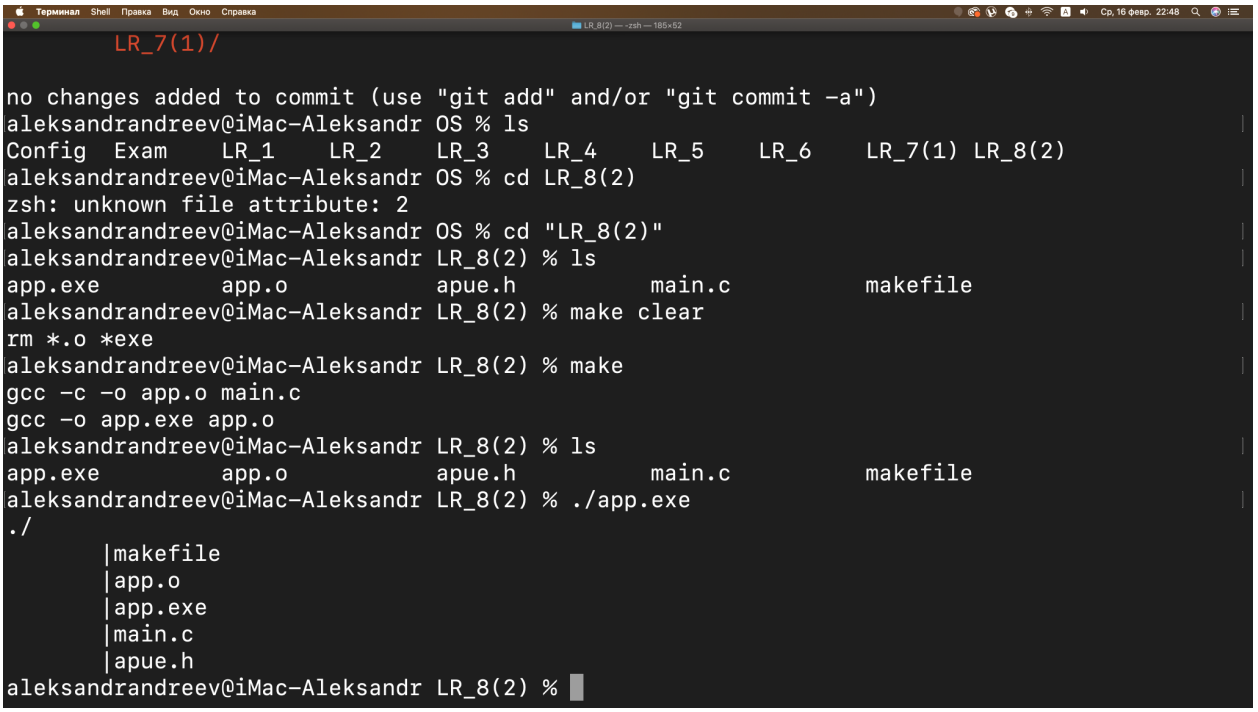

Рисунок 1: Результат работы программы без аргумента директории

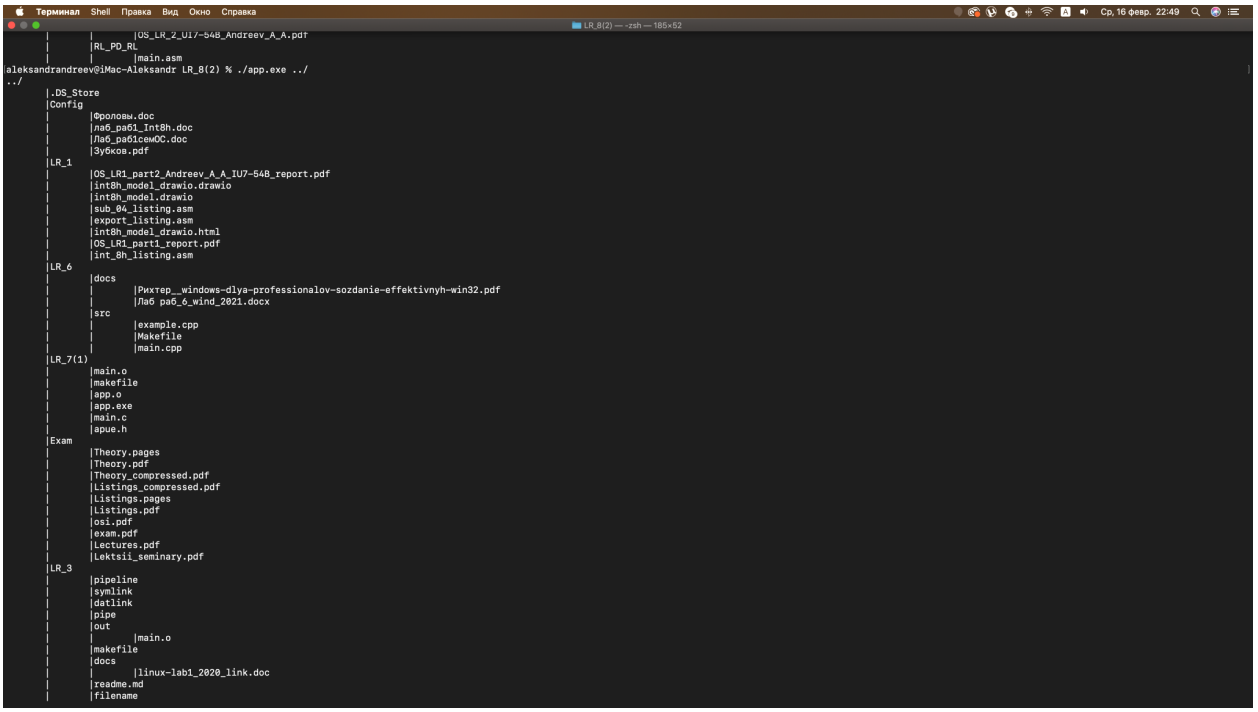

Рисунок 2: Результат работы программы с аргументом директории, Часть 1

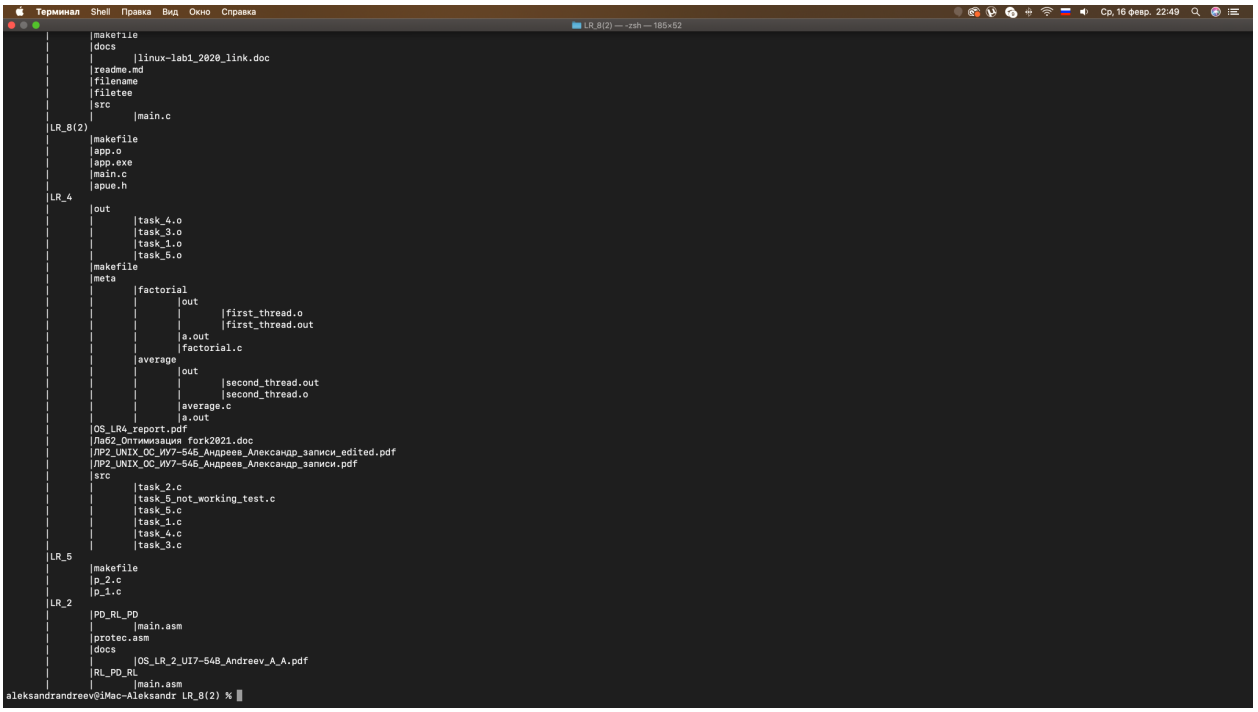

Рисунок 3: Результат работы программы с аргументом директории, Часть 2