Министерство образования и науки РФ

Федеральное государственное автономное образовательное учреждение высшего образования

«Национальный исследовательский Нижегородский государственный университет

им. Н.И. Лобачевского»

Институт информационных технологий, математики и механики

**Отчет по учебной практике**

**Прототип файлового менеджера**

**Выполнил:**

Студент группы 381806-3

Живайкин Д.Е.

**Проверил:**

Доцент кафедры МОСТ ИИТММ

Кустикова В.Д.

Нижний Новгород 2018 г.

Оглавление

[1 Введение. 3](#_Toc533633504)

[2 Постановка задачи. 4](#_Toc533633505)

[3 Руководство пользователя. 5](#_Toc533633506)

[4 Руководство программиста. 8](#_Toc533633507)

[4.1 Структура программы. 8](#_Toc533633508)

[4.2 Описание алгоритмов. 8](#_Toc533633509)

[4.2.1 Сортировка выбором. 8](#_Toc533633510)

[4.2.2 Сортировка простыми вставками. 8](#_Toc533633511)

[4.2.3 Сортировка пузырьком. 9](#_Toc533633512)

[4.2.4 Сортировка подсчетом. 9](#_Toc533633513)

[4.2.5 Быстрая сортировка. 10](#_Toc533633514)

[4.2.6 Сортировка слиянием. 10](#_Toc533633515)

[4.3 Описание фукций 11](#_Toc533633516)

[5 Заключение. 13](#_Toc533633517)

[6 Литература. 14](#_Toc533633518)

# Введение.

Сортировка является важной и полезной процедурой при работе с большой группой элементов. Обрабатывать информацию проще всего тогда, когда перед нами понятный отсортированный список элементов Для наглядного представления произведем операцию сортировки файлов по их размеру. Так, например, при удалении ненужных файлов, будет проще определить, какие именно файлы занимают большой объем памяти

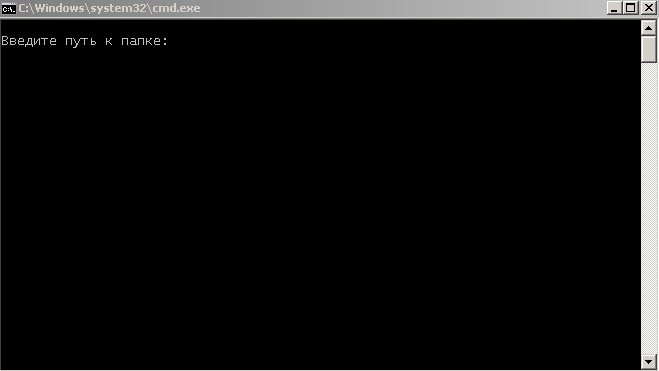
# Постановка задачи.

Реализовать программу файлового менеджера на языке программирования C, который сортирует файлы в необходимой директории выбранным способом файлы по возрастанию/убыванию их размера.

# Руководство пользователя.

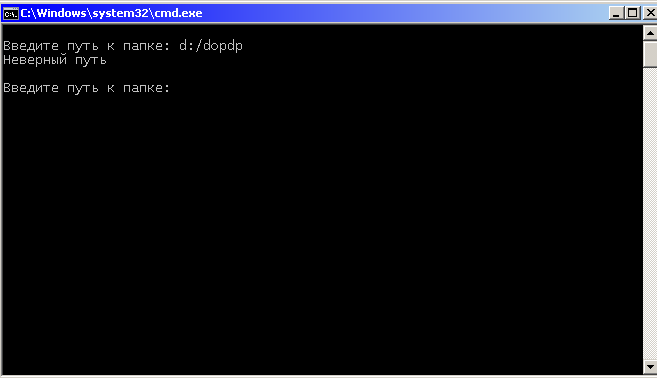
Здесь содержится описание действий, приводящих к успешной работе программы:

Запустить Practice5.exe (Рис.1).



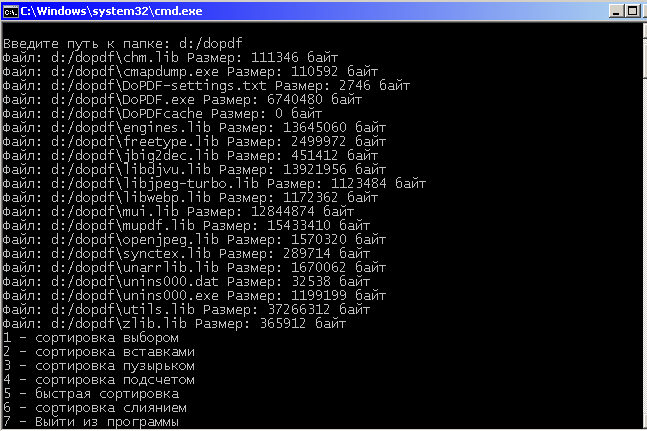
1. Запуск программы.

Ввести путь к какой-либо папке на вашем компьютере и нажмите Enter. Если ввести несуществующий путь, то программа выдаст ошибку (Рис.2).

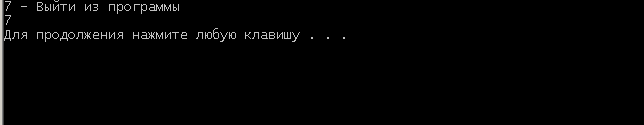


1. Ошибка при вводе неверного пути.

Программа выведет файлы из указанной папки с размерами в байтах и меню, включающее в себя все методы сортировки, используемые в программе (Рис.3). Далее вам следует выбрать номер сортировки в списке и нажать **Enter**. Если ввести 7, то программа завершит свою работу (Рис.4).

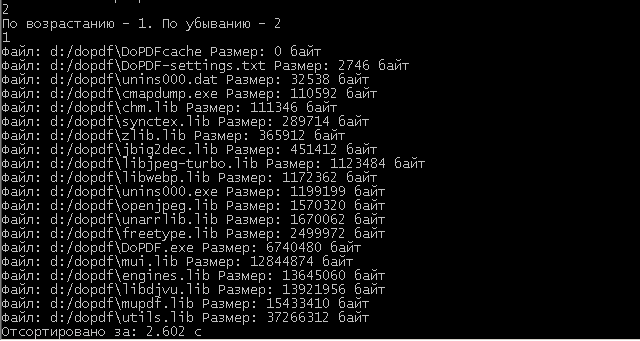


1. Вывод списка файлов и меню выбора сортировки.



1. Выход из программы.

Когда вы выбрали метод сортировки, программа выведет отсортированный список файлов по возрастанию или убыванию (будет предложен доп. Выбор), а также время сортировки (Рис.5). Если вы хотите выйти из программы, введите 7 и нажмите Enter.



1. Вывод отсортированного списка и времени сортировки.

# Руководство программиста.

## Структура программы.

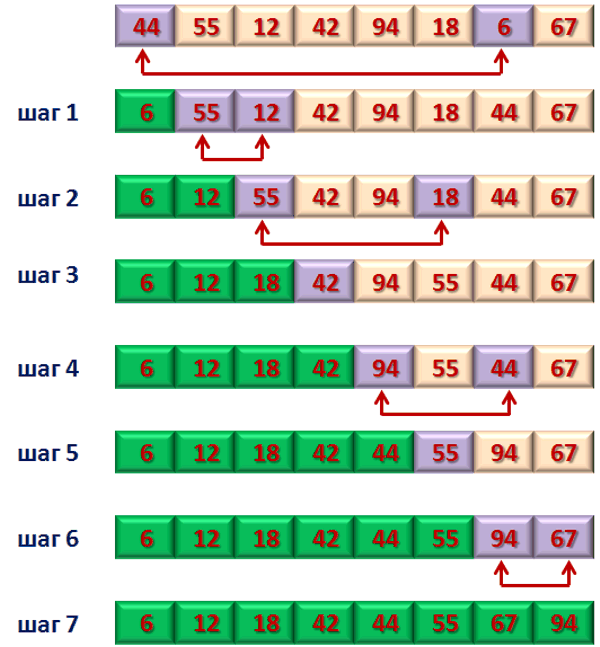
Файл **main5.c**, в котором содержится полный код всей программы.

## Описание алгоритмов.

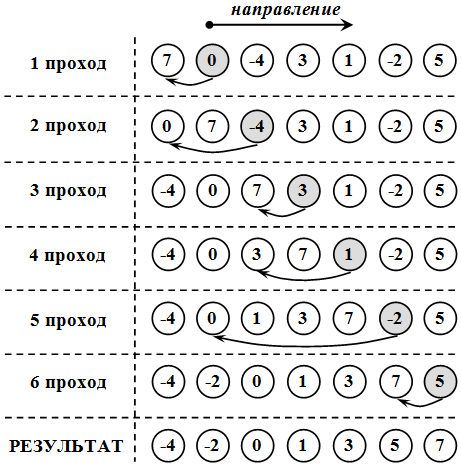
В этом пункте содержится описание всех методов сортировки, используемых в программе.

### Сортировка выбором.

Сортировка выбором основана на линейном поиске минимального элемента в массиве, после которого этот элемент меняется с крайним левым элементом массива. После чего левая часть, состоящая из одного элемента, считается отсортированной. Затем эта часть мысленно отбрасывается, и те же действия проводятся с остальными элементами массива. Пример показан на рисунке:

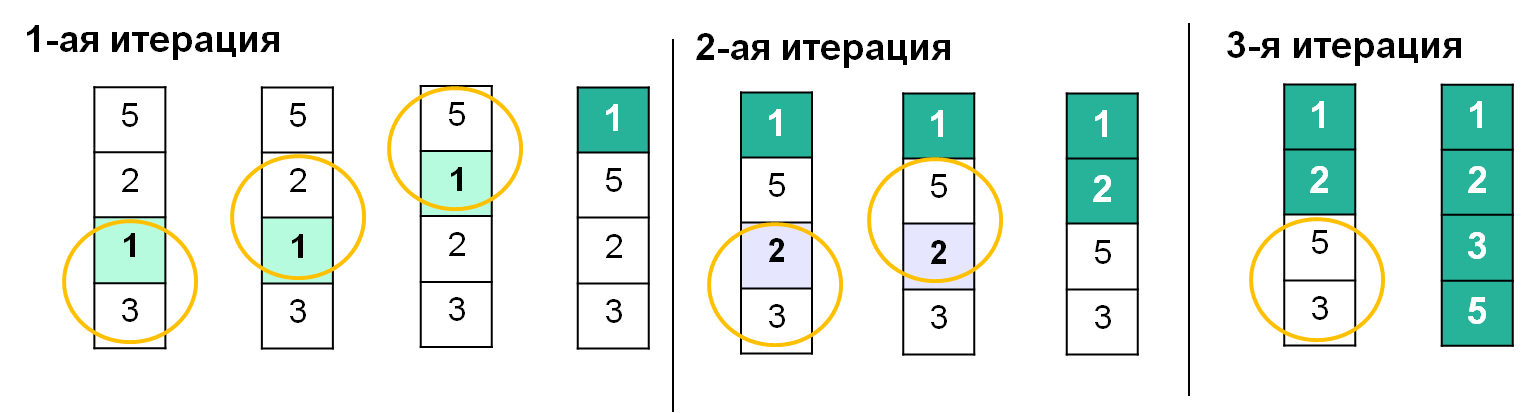


### Сортировка простыми вставками.

Сортировка вставками (*Insertion Sort*) — это простой алгоритм сортировки. Суть его заключается в том что, на каждом шаге алгоритма мы берем один из элементов массива, находим позицию для вставки и вставляем.

### Сортировка пузырьком.

Алгоритм данной сортировки основан на «всплытии» меньшего элемента массива. Массив представляется вертикально (для наглядности), затем нижний элемент сравнивается с элементом выше. Если выбранный элемент меньше, то они меняются местами. Так продолжается до тех пор, пока элемент не встретит меньший элемент. Далее все те же операции проводятся для элементов ниже. Пример:



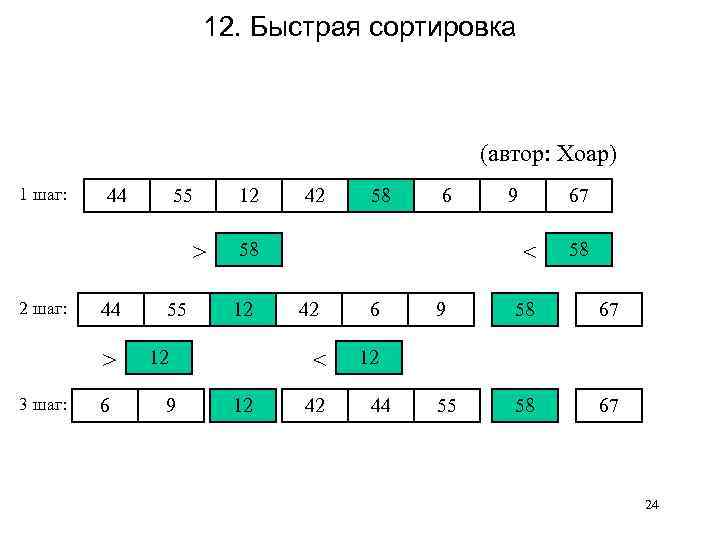
### Сортировка подсчетом.

Алгоритм этой сортировки основан на том, что известен диапазон значений элементов массива. В новый массив записывается количество одинаковых элементов (после прохода по массиву). Затем проходим по новому массиву, и в исходный массив записываем столько элементов, сколько написано в массиве подсчета.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Индексы | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| 1 шаг | Исходный массив: | | | | | | | | |
|  | 4 | 6 | 1 | 0 | 7 | 5 | 3 | 1 | 2 |
| 2 шаг | Массив подсчета (элемент 0 индекса – кол-во нулей и т.д.): | | | | | | | | |
|  | 1 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 3 шаг | Отсортированный массив: | | | | | | | | |
|  | 0 | 1 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |

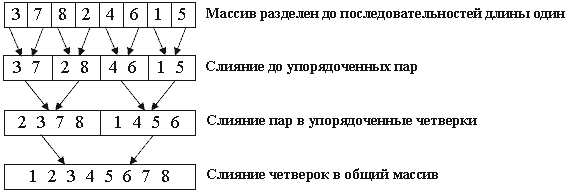
### Быстрая сортировка.

Алгоритм быстрой сортировки основан на выборе элемента массива (случайного или центрального), а затем перемещении всех элементов, меньших его, влево, а всех элементов, больших его, вправо от самого элемента. Затем в следующей части выбирается элемент и с ним проводится та же операция что и над первым.



### Сортировка слиянием.

Исходный массив разбивается на 2 массива пополам нацело, далее – каждая часть еще на 2 массива. Так продолжается до тех пор, пока части не будут состоять из одного элемента. Затем две такие части «сливают» в массив (упорядоченно). Так делают со всеми частями, т.е. проходят обратно, в итоге получая исходный отсортированный массив. Пример:



## Описание фукций

void choose\_sort(int \*a, ULONGLONG \*size, int n)

Назначение: сортировка выбором для указанного массива указанного размера.

Входные параметры: size – массив размеров файлов, которые нужно упорядочить; a – количество упорядочиваемых файлов; n – массив индексов файлов.

Выходные параметры: указатель на массив индексов файлов.

void insert\_sort(int \*a, ULONGLONG \*size, int n)

Назначение: сортировка простыми вставками для массива указанного размера.

Входные параметры: size – массив размеров файлов, которые нужно упорядочить; a – количество упорядочиваемых файлов; n – массив индексов файлов.

Выходные параметры: указатель на массив индексов файлов.

void bubble\_sort(int \*a, ULONGLONG \*size, int n)

Назначение: сортировка пузырьком для массива указанного размера.

Входные параметры: size – массив размеров файлов, которые нужно упорядочить; a – количество упорядочиваемых файлов; n – массив индексов файлов.

Выходные параметры: указатель на массив индексов файлов.

void counting\_sort(int \*a, ULONGLONG \*size, int n)

Назначение: сортировка подсчетом для массива указанного размера.

Входные параметры: size – массив размеров файлов, которые нужно упорядочить; a – количество упорядочиваемых файлов; n – массив индексов файлов.

Выходные параметры: указатель на массив индексов файлов.

int\* quick\_sort(int \*a, ULONGLONG \*size, int n1, int n2)

Назначение: быстрая сортировка для массива указанного размера.

Входные параметры: size – массив размеров файлов, которые нужно упорядочить; a – количество упорядочиваемых файлов; n1 – первый индекс; n2 – последний индекс.

Выходные параметры: указатель на массив индексов файлов.

void merge\_sort(int\* a, ULONGLONG\* size, int l, int m, int r)

Назначение: вспомогательная функция для сортировки слиянием. Соединяет подмассивы.

Входные параметры: a - массив индексов; size - массив размеров файлов; l – индекс первого элемента; m – индекс среднего элемента; r – индекс последнего элемента.

void mergesort(int\* a, ULONGLONG\* size, int l, int r)

Назначение: сортировка слиянием для массива.

Входные параметры: a - массив индексов; size - массив размеров файлов; l – индекс первого элемента; r – индекс последнего элемента.

Выходные параметры: указатель на массив индексов файлов.

int ListDirectoryContents(const wchar\_t\* sDir, ULONGLONG\*\* sizes, wchar\_t\*\* names)

Назначение: подсчет количества элементов в указанной папке и получение их названий и размеров.

Входные параметры: sDir – строка, содержащая путь к папке; names, sizes – массивы для записи полных имен и размеров файлов.

void output(int \*Idxes, wchar\_t \*\*names, ULONGLONG \*sizes, int count)

Назначение: печать названий файлов.

Входные параметры: размеры файлов, количество файлов, названия файлов, индексы файлов .

void Menu()

Назначение: вывод командного меню.

Входные параметры: -

Выходные параметры: -

# Заключение.

Была разработана программа-прототип файлового менеджера для сортировки файлов в указанной директории по размеру занимаемой памяти. Использовались следующие методы сортировки:

1. Сортировка выбором.
2. Сортировка простыми вставками.
3. Сортировка пузырьком.
4. Сортировка подсчетом.
5. Быстрая сортировка.
6. Сортировка слиянием.

# Литература.

# <https://habr.com/post/181271/> - В мире алгоритмов: Сортировка Вставками

# <https://tproger.ru/translations/sorting-for-beginners/> Алгоритмы и структуры данных для начинающих: сортировка