Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное

учреждение высшего профессионального образования

Нижегородский государственный университет им. Н.И. Лобачевского

Институт информационных технологий, математики и механики

## Отчет по лабораторной работе

## Прототип файлового менеджера с функцией показа файлов в заданном каталоге.

**Выполнил**:

студент группы 381806-3

Кузьмичёва А.И.

**Проверил**:

Доцент кафедры МОСТ ИИТММ Кустикова В.Д.

Нижний Новгород

2018

Содержание

[Введение 3](file:///C:\Users\1\Downloads\Отчет%20Тюрмина.docx#_Toc533556253)

[1 Постановка задач 4](file:///C:\Users\1\Downloads\Отчет%20Тюрмина.docx#_Toc533556254)

[2 Руководство пользователя 5](file:///C:\Users\1\Downloads\Отчет%20Тюрмина.docx#_Toc533556255)

[3 Руководство программиста 7](file:///C:\Users\1\Downloads\Отчет%20Тюрмина.docx#_Toc533556256)

[3.1 Структура программы 7](file:///C:\Users\1\Downloads\Отчет%20Тюрмина.docx#_Toc533556257)

[3.2 Описание алгоритмов 7](file:///C:\Users\1\Downloads\Отчет%20Тюрмина.docx#_Toc533556258)

[3.2.1 Сортировка выбором 7](file:///C:\Users\1\Downloads\Отчет%20Тюрмина.docx#_Toc533556259)

[3.2.2 Сортировка пузырьком 7](file:///C:\Users\1\Downloads\Отчет%20Тюрмина.docx#_Toc533556260)

[3.2.3 Сортировка вставками 8](file:///C:\Users\1\Downloads\Отчет%20Тюрмина.docx#_Toc533556261)

[3.2.4 Сортировка подсчетом 9](file:///C:\Users\1\Downloads\Отчет%20Тюрмина.docx#_Toc533556262)

[3.2.5 Сортировка слиянием 10](file:///C:\Users\1\Downloads\Отчет%20Тюрмина.docx#_Toc533556263)

[3.2.6 Быстрая сортировка(Хоара) 10](file:///C:\Users\1\Downloads\Отчет%20Тюрмина.docx#_Toc533556264)

[3.3 Описание функций 11](file:///C:\Users\1\Downloads\Отчет%20Тюрмина.docx#_Toc533556265)

[Заключение 13](file:///C:\Users\1\Downloads\Отчет%20Тюрмина.docx#_Toc533556266)

[Литература 14](file:///C:\Users\1\Downloads\Отчет%20Тюрмина.docx#_Toc533556267)

Введение

Под сортировкой понимают процесс перестановки объектов данного множества в определенном порядке.

Цель сортировки — облегчить последующий поиск элементов в отсортированном множестве. Следовательно, методы сортировки важны особенно при обработке данных

Зависимость выбора алгоритма от структуры данных — явление довольно частое, и в случае сортировки она настолько сильна, что методы сортировки обычно разделяют на две категории: сортировка массивов (внутренняя сортировка) и сортировка файлов (внешняя сортировка). Массивы обычно располагаются в оперативной памяти, для которой характерен быстрый произвольный доступ; файлы хранятся в более медленной, но более вместительной внешней памяти, на дисках

Основное требование к методам сортировки массивов — экономное использование памяти. Это значит, что перестановки элементов нужно выполнять *на том же месте оперативной памяти,*где они находятся, и что методы, которые пересылают элементы из массива *А*в массив *В,*не представляют интереса. Таким образом, выбирая метод сортировки, руководствуясь критерием экономии памяти, классификацию алгоритмов, проводят в соответствии с их *эффективностью,*т.е. *быстродействием.*Удобная мера *эффективности*получается при подсчете числа необходимых *сравнений ключей С*и *пересылок элементов М.*Эти параметры зависят от числа сортируемых элементов *п.*Хорошие алгоритмы сортировки требуют порядка *nlogn*сравнений

Постановка задачи

В данной работе необходимо реализовать программу файлового менеджера, которая отсортирует файлы по возрастанию/убыванию размера. Используя при этом различные методы сортировки:

* «пузырьком»
* Выбором
* Вставками
* Слиянием
* Хоара (так же известная, как «Быстрая»)
* подсчетом

Входные данные:

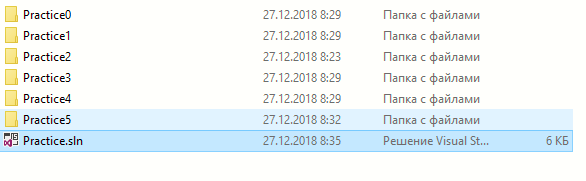
1. путь до директории, в которой необходимо отсортировать содержимое
2. метод сортировки
3. выбор сортировки по возрастанию или убыванию

Выходные данные:

1. отсортированный список имен файлов с указанием размера
2. время сортировки

Руководство пользователя

Запустить проект «Practice.sln»

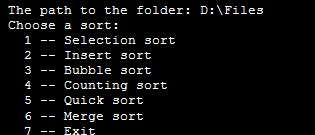


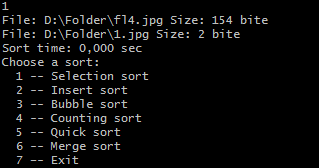
Запускаем 

После запуска программы вводим название папки, в которой нужно сделать сортировку



Введя верный путь предлагается один из методов сортировки

 Программа выводит на экран отсортированные файлы с их размерами в указанных единицах измерения.



Руководство программиста

* 1. Структура программы

Файл Practice5.c в котором содержатся весь код программы. Так же программа состоит из функции сортировок, функции вывода меню команд, функции ввода директории, функции вывода отсортированных названий файлов, их размеров и времени сортировки.

* 1. Описание алгоритмов
     1. Сортировка выбором

Идея метода состоит в том, чтобы создавать отсортированную последовательность путем присоединения к ней одного элемента за другим в правильном порядке.   
Будем строить готовую последовательность, начиная с левого конца массива. Алгоритм состоит из n последовательных шагов, начиная от нулевого и заканчивая (n-1)-м.   
На i-м шаге выбираем наименьший из элементов a[i] ... a[n] и меняем его местами с a[i]

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Индекс  Шаг | Массив | | | | | |
| 0 | 1 | 2 | 3 | 4 | 5 |
| Исходный | 4 | 9 | 7 | 6 | 2 | 3 |
| 0 | **2** | 9 | 7 | 6 | 4 | 3 |
| 1 | **2** | **3** | 7 | 6 | 4 | 9 |
| 2 | **2** | **3** | **4** | 6 | 7 | 9 |
| 3 | **2** | **2** | **3** | **6** | 7 | 9 |
| 4 | **2** | **3** | **4** | **6** | **7** | 9 |
| Отсортированный | **2** | **3** | **4** | **6** | **7** | **9** |
| Таб1. Пример сортировки массива. | | | | | | |

3.2.2. Пузырьком

Идея метода: шаг сортировки состоит в проходе снизу вверх по массиву. По пути просматриваются пары соседних элементов. Если элементы некоторой пары находятся в неправильном порядке, то меняем их местами.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 2-6 | 2-7 | 2-9 | | 2-4 |
| 4 | 4 | 4 | | 4 | 2 |
| 9 | 9 | 9 | | 2 | 4 |
| 7 | 7 | 2 | | 9 | 9 |
| 6 | 2 | 7 | | 7 | 7 |
| 2 | 6 | 6 | | 6 | 6 |
| 3 | 3 | 3 | | 3 | 3 |

После нулевого прохода по массиву "вверху" оказывается самый "легкий" элемент - отсюда аналогия с пузырьком. Следующий проход делается до второго сверху элемента, таким образом второй по величине элемент поднимается на правильную позицию...   
  
Делаем проходы по все уменьшающейся нижней части массива до тех пор, пока в ней не останется только один элемент. На этом сортировка заканчивается, так как последовательность упорядочена по возрастанию.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| проход | 0 | 1 | 2 | 3 | 4 | 5 |
|  | 4 | **2** | **2** | **2** | **2** | **2** |
|  | 9 | 4 | **3** | **3** | **3** | **3** |
|  | 7 | 9 | 4 | **4** | **4** | **4** |
|  | 6 | 7 | 9 | 9 | **6** | **6** |
|  | 2 | 6 | 7 | 7 | 9 | **7** |
|  | 3 | 3 | 6 | 6 | 7 | **9** |

* + 1. Сортировка вставками

Суть алгоритма заключается в том, что на каждом шаге выбирается элемент из неотсортированной части массива, после чего он вставляется на нужную позицию в уже отсортированной части массива. Выбор элемента из массива можно осуществлять произвольно, но удобнее брать их по порядку.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Индекс  Шаг | Массив | | | | | | |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| 1 | ***-1*** | 4 | 9 | -3 | 8 | -8 | 0 |
| 2 | ***-1*** | ***4*** | 9 | -3 | 8 | -8 | 0 |
| 3 | ***-1*** | ***4*** | ***9*** | -3 | 8 | -8 | 0 |
| *-1* | *4* | -3 | *9* | 8 | -8 | 0 |
| *-1* | -3 | *4* | *9* | 8 | -8 | 0 |
| *-3* | -1 | *4* | *9* | 8 | -8 | 0 |
| 4 | ***-3*** | ***-1*** | ***4*** | ***9*** | 8 | -8 | 0 |
| *-3* | *-1* | *4* | 8 | *9* | -8 | 0 |
| 5 | ***-3*** | ***-1*** | ***4*** | ***8*** | ***9*** | -8 | 0 |
| *-3* | *-1* | *4* | *8* | *-8* | 9 | 0 |
| *-3* | *-1* | *4* | *-8* | *8* | 9 | 0 |
| *-3* | *-1* | *-8* | *4* | *8* | 9 | 0 |
| *-3* | *-8* | *-1* | *4* | *8* | 9 | 0 |
| *-8* | *-3* | *-1* | *4* | *8* | 9 | 0 |
| 6 | ***-8*** | ***-3*** | ***-1*** | ***4*** | ***8*** | ***9*** | 0 |
| *-8* | *-3* | *-1* | *4* | *8* | 0 | *9* |
| *-8* | *-3* | *-1* | *4* | 0 | *8* | *9* |
| *-8* | *-3* | *-1* | 0 | *4* | *8* | *9* |
| Отсортированный массив | ***-8*** | ***-3*** | ***-1*** | **0** | ***4*** | ***8*** | ***9*** |

3.2.4 Сортировка подсчетом

Суть алгоритма заключается в том, что на каждом шаге выбирается элемент из неотсортированной части массива, после чего он вставляется на нужную позицию в уже отсортированной части массива. Выбор элемента из массива можно осуществлять произвольно, но удобнее брать их по порядку.

При сортировке методом подсчета упорядоченная последовательность элементов создается на свободном участке памяти. В отсортированной последовательности, элемент, занимающий позицию с номером К+1, превышает ровно К элементов, поэтому в процессе сортировки методом подсчета на каждом i-ом проходе мы попарно сравниваем i-й элемент со всеми элементами массива. Если установлено, что mass[i] > mass[j], то увеличиваем счетчик К на единицу (в начале К = 0). По окончании текущего прохода счетчик К указывает количество элементов, меньших, чем mass[i], поэтому элемент mass[i] занимает в отсортированной последовательности позицию К + 1

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Индекс | Массив | | | | | | |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|  | 2 | 1 | 0 | 1 | 2 | 3 | 2 |
| *0* | 1 | 0 | 1 | 2 | 3 | 2 |
| 0 | *1* | *1* | 1 | 2 | 3 | 2 |
| 0 | 1 | 1 | *2* | *2* | *2* | 2 |
| 0 | 1 | 1 | 2 | 2 | 2 | *3* |
| Отсортированный массив | 0 | 1 | 1 | 2 | 2 | 2 | 3 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Значение элемента  Элементы | Вспомогательный массив | | | |
| 0 | 1 | 2 | 3 |
|  | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 |
| 0,1 | 0 | 1 | 1 | 0 |
| 0,1,2 | 1 | 1 | 1 | 0 |
| 0,1,2,3 | 1 | 2 | 1 | 0 |
| 0,1,2,3,4 | 1 | 2 | 2 | 0 |
| 0,1,2,3,4,5 | 1 | 2 | 2 | 1 |
| 0,1,2,3,4,5,6 | 1 | 3 | 2 | 1 |

3.2.5 Сортировка слиянием

Идея сортировки слияния заключается в том, что сначала мы рекурсивно делим исходный массив на массивы, состоящие из одного элемента, а затем на каждом новом шаге, упорядочивая массивы, рекурсивно попарно сливаем их, приводя в виду исходного массива.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Индекс  Шаг | Массив | | | | | | |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| 1 | *-1* | *4* | 9 | -3 | 8 | -8 | 0 |
| -1 | 4 | *9* | *-3* | 8 | -8 | 0 |
| -1 | 4 | -3 | 9 | *8* | *-8* | 0 |
| -1 | 4 | -3 | 9 | -8 | 8 | *0* |
| 2 | *-1* | *4* | *-3* | *9* | -8 | 8 | 0 |
| -3 | -1 | 4 | 9 | *-8* | *8* | *0* |
| 3 | *-3* | *-1* | *4* | *9* | *-8* | *0* | *8* |
| Отсортированный массив | -8 | -3 | -1 | 0 | 4 | 8 | 9 |

### Быстрая сортировка (Хоара)

Принцип работы таков: значение какого-нибудь элемента, обычно центрального, записывается в переменную Х. Просматриваются элементы массива. При движении слева - направо ищем элемент больше или равный Х. А при движении справа – налево ищем элемент меньше или равный Х. Найденные элементы меняются местами и продолжается встречный поиск.   
После этого массив окажется разделенным на две части. В первой находятся элементы меньше либо равные Х, а справа – больше или равные Х.   
Выбор среднего значения можно делать или случайно или поиском некоторого среднего значения, например посередине массива или выбрав действительно средний по значению элемент.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Индекс  Шаг |  | | | | | | |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|  | 9 | 8 | 7 | 4 | 2 | 1 | 1 |
| 1 | 1 | 1 | 2 | 4 | 9 | 8 | 7 |
| 2 | 1 | 1 | 2 | 4 | 7 | 8 | 9 |

* 1. Описание функций

### Функция void SelectionSort(int \*a, ULONGLONG \*size, int n)

Функция реализует сортировку массива размеров файлов сортировкой выбором с изменением массива индексов. Принимает на вход массив индексов (\*a), массив размеров файлов (\*size) и количество файлов в каталоге (n).

### Функция void InsertSort(int \*a, ULONGLONG \*size, int n)

Функция реализует сортировку массива размеров сортировкой вставками с изменением массива индексов. В качестве входных данных принимает массив индексов (\*a), массив размеров файлов (\*size) и количество файлов в каталоге (n).

### Фунцкия void BubbleSort(int \*a, ULONGLONG \*size, int n)

Функция реализует сортировку массива размеров пузырьковой сортировкой с изменением массива индексов. Входными данными являются: массив индексов (\*a), массив размеров файлов (\*size) и количество файлов в каталоге (n).

### Функция void CountingSort(int \*a, ULONGLONG \*size, int n)

Функция реализует сортировку массива размеров сортировкой подчетом с изменением массива индексов. На вход принимает массив индексов (\*a), массив размеров файлов(\*size) и количество файлов в каталоге (n).

### Функция void QuickSplit(int \*a, ULONGLONG \*size, int \*i, int \*j, ULONGLONG p)

Вспомогательная функция для сортировки Хоара (быстрой сортировки). Принимает на вход массив размеров файлов (\*size), адреса первого и второго индексов (\*i и \*j соответственно) и ведущий элемент p.

### Функция void QuickSort(int \*a, ULONGLONG \*size, int n1, int n2)

Функция, сортирующая массив размеров файлов сортировкой Хоара (быстрой сортировкой) с изменением массива индексов. В качестве входных данных принимает массив индексов (\*a), массив размеров файлов (\*size) и индексы первого и второго элементов (n1 и n2соответственно).

### Функция void Merge(int \*a, ULONGLONG \*size, int l, int m, int r)

Вспомогательная функция для сортировки слиянием, обеспечивающая соединение массивов с их упорядочиванием. Входными данными являются массив индексов (\*a), массив размеров файлов (\*size) и индексы первого, среднего и последнего элементов (l, m и r соответственно).

### Функция void MergeSort(int \*a, ULONGLONG \*size, int l, int r)

Функция реализует сортировку массива размеров файлов сортировкой слиянием с изменением массива индексов. В качестве входных данных используются массив индексов (\*a), массив размеров файлов (\*size) и индексы первого и второго элементов (l и r соответственно).

### Функция int FileDirectory(const wchar\_t \*sDir, ULONGLONG \*\*ssize, wchar\_t \*\*\*nname)

Функция проверяет правильность введенных данных, определяет количество файлов в директории, формирует массивов названий файлов и их размеров, а затем заполняет его. На вход принимает путь до каталога (\*sDir), массив размеров файлов (\*\*ssize) и адрес массива названий (\*\*\*nname). На выходе получаем количество файлов в каталоге (сount).

### Функция void input(wchar\_t \*\*strin)

Эта функция позволяет вводить путь к заданному каталогу. В качестве входных данных используется указатель на массив (\*\*strin). В этот указатель пользователем будет записано некоторое значение.

### Функция void CommandMenu()

Функция, выводящая на экран командное меню.

### Функция void Output(int \*index, wchar\_t \*\*nname, ULONGLONG \*ssize, int count)

Функция выводит на экран отсортированный список имен файлов с указанием их размера и времени сортировки. На вход принимает массив индексов отсортированного массива размеров файлов (\*index), массив названий файлов (\*\*nname), массив размеров файлов (\*ssize) и количество файлов (сount).

Заключение

Выше были приведены различные методы сортировки. Для определения наиболее быстрой сортировки массивов был проведён эксперимент, в котором измерялось время, за которое каждая сортировка упорядочивает файлы по размеру. Для сортировки использовалось 6 методов, которые может выбрать пользователь. В таблице представлено сравнение времени работы и значения память, необходимые для разных видов сортировок.

|  |  |  |
| --- | --- | --- |
| Метод сортировки | Затраченное время | Память |
| Сортировка выбором | O(n2) | O(1) |
| Сортировка вставками | O(n2) | O(1) |
| Пузырьковая сортировка | O(n2) | O(1) |
| Сортировка подсчетом | O(n) | (max(n) – min(n)) + 1 |
| Сортировка слиянием | O(n·log2 n) | O(1) |
| Сортировка Хоара | O(n·log2 n) | O(n) |

Литература

1. Сайт Algolist. Сортировка выбором – <http://algolist.manual.ru/sort/select_sort.php>.
2. Сайт vscode Сортировка вставками - <https://vscode.ru/prog-lessons/sortirovka-metodom-podscheta-na-si.html>
3. Сайт Pascal. Основы программирования. Сортировка пузырьком - <https://pas1.ru/bubbles>
4. Сайт vingrad. Сортировка подсчетом - <http://base.vingrad.ru/view/3011-Sortirovka-podschetom->
5. Сайт Kvodo. Сортировка слиянием - <http://kvodo.ru/mergesort.html>
6. Сайт STUDLAB.com – Студенческая лаборатория. Сортировка Хоара - <http://studlab.com/news/sortirovka_khoara/2011-05-31-116>
7. Подбельский В. В., Фомин С.С. Курс программирования на языке Си. Учебник ДМК-Пресс -2012 104 с.
8. Сайт StudFiles (https://studfiles.net/preview/5707298/page:9/)