Московский государственный университет имени М. В. Ломоносова Факультет вычислительной математики и кибернетики

Отчёт по заданию №1

**«Методы сортировки»**

**Вариант 1/3/2/3**

Выполнил:

Студент 102 группы

Мартьянов А.О.

Преподаватель:

Кулагин А.В.

Москва

2022

**Содержание**

**Постановка задачи 3**

**Результаты экспериментов 4**

**Структура программы и спецификация функций 6**

**Отладка программы, тестирование функций 7**

**Анализ допущенных ошибок 8**

**Список цитируемой литературы 9**

**Постановка задачи**

В ходе выполнения задания требовалось:

* Реализовать два алгоритма сортировки массива чисел на языке С: **метод Шелла** и **метод простого выбора**
* Сгенерировать три вида массивов, состоящих из случайных **чисел типа int**:
  + Упорядоченные в прямом порядке
  + Упорядоченные в обратном порядке
  + неупорядоченные
* Отсортировать сгенерированные массивы в **порядке неубывания модулей элементов**
* Экспериментально сравнить алгоритмы: **количество сравнений и обменов** элементов
* Провести теоритическую оценку каждого из алгоритмов
* Проанализировать ошибки, допущенные во время выполнения задания

**Результаты экспериментов**

**Сортировка методом Шелла**

Результаты работы:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **n** | **Параметр** | **Номер сгенерированного массива** | | | | **Среднее значение** |
| 1 | 2 | 3 | 4 |
| 10 | Сравнения | 22 | 27 | 31 | 28 | 27.0 |
| Перемещения | 0 | 13 | 13 | 13 | 9.75 |
| 100 | Сравнения | 503 | 668 | 861 | 891 | 730.75 |
| Перемещения | 0 | 260 | 414 | 438 | 278.0 |
| 1000 | Сравнения | 8006 | 11716 | 14947 | 14842 | 12377.75 |
| Перемещения | 0 | 4700 | 7423 | 7377 | 4875.0 |
| 10000 | Сравнения | 120005 | 172578 | 266709 | 270218 | 207377.5 |
| Перемещения | 0 | 62560 | 151657 | 155346 | 92390.75 |

Таблица 1: результат работы алгоритма сортировки методом Шелла.

Теоритическая оценка работы алгоритма[1]:

Время выполения алгоритма зависит от выбора промежутка, через который мы сравниваем элементы. Существуют различные модификации, но в нашем случае этот промежуток выбирается по Шеллу, где d1= n / 2; di = di – 1 / 2.

Среднее число сравнений ≈ n \* log2 n

Оценка сложности:

* Худший случай: O(n2)
* Средний случай: O(n1.5)
* Лучший случай: O(n \* log22 n)

**Сортировка методом простого выбора**

Результаты работы:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **n** | **Параметр** | **Номер сгенерированного массива** | | | | **Среднее значение** |
| 1 | 2 | 3 | 4 |
| 10 | Сравнения | 45 | 45 | 45 | 45 | 45.0 |
| Перемещения | 10 | 10 | 10 | 10 | 10.0 |
| 100 | Сравнения | 4950 | 4950 | 4950 | 4950 | 4950.0 |
| Перемещения | 100 | 100 | 100 | 100 | 100.0 |
| 1000 | Сравнения | 499500 | 499500 | 499500 | 499500 | 499500.0 |
| Перемещения | 1000 | 1000 | 1000 | 1000 | 1000.0 |
| 10000 | Сравнения | 49995000 | 49995000 | 49995000 | 49995000 | 49995000.0 |
| Перемещения | 10000 | 10000 | 10000 | 10000 | 10000.0 |

Таблица 2: результат работы алгоритма сортировки методом простого выбора

Теоритическая оценка работы алгоритма[2]:

Количество сравнений постоянно: (n2 – n) / 2

Обмены:

* Худший случай(обратный порядок упорядоченности) – n/2 – каждая пара элементов в инвертированном порядке
* Лучший слуачай(прямой порядок упорядоченности) – 0 – обмены не требуются, все элементы стоят в нужном порядке
* Среднее значение: n(ln(n) + g), где g ≈ 0.58 – константа Эйлера

**Структура программы и спецификация функций**

**Глобальные переменные**

* **long long** compare\_count – количество сравнений
* **long long** change\_count – количество обменов элементов

**Функции**

Здесь и далее int \*a, int n – указатель на массив а и его длина n.

* int compare\_for\_straight\_sort(const void \*a, const void \*b) – компаратор для использования функции qsort при создании упорядоченного по неубыванию модулей массива
* int compare\_for\_reverse\_sort(const void \*a, const void \*b) - компаратор для использования функции qsort при создании упорядоченного по невозрастанию модулей массива
* void reverse\_array\_generate(int n, int\* a) – генерация массива случайных чисел, упорядоченных по невозрастанию модулей
* void straight\_array\_generate(int n, int\* a) - генерация массива случайных чисел, упорядоченных по неубыванию модулей
* void random\_array\_generate(int n, int \*a) – генерация массива случайных неупорядоченных чисел
* void shell\_sort(int n, int \*a) – реализация сортировки методом Шелла
* void simple\_selection\_sort(int n, int \*a) - реализация сортировки методом простого выбора
* void print\_array(int n, int \*a) – функция печати массива на стандартный поток вывода

**Отладка программы, тестирование функций**

Алгоритмы несколько раз запускались на тестовых наборах случайных чисел, после чего были исправлены небольшие неточности.

|  |  |  |
| --- | --- | --- |
| **№** | **Входной массив** | **Отсортированный массив** |
| 1 | 419612 82430 -377281 -52496 -445647 | -52496 82430 -377281 419612 -445647 |
| 2 | -224688 51691 445119 52539 -153243 -484065 -312348 | 51691 52539 -153243 -224688 -312348 445119 -484065 |
| 3 | -115332 477889 -281685 | -115332 -281685 477889 |

Таблица 3: тесты для сортировки методом Шелла

|  |  |  |
| --- | --- | --- |
| **№** | **Входной массив** | **Отсортированный массив** |
| 1 | -7085 -18398 5501 -13267 10818 | 5501 -7085 10818 -13267 -18398 |
| 2 | -49042 41638 -30866 -35192 -25190 | -25190 -30866 -35192 -41638 -49042 |
| 3 | 2727 -1666 -2082 -927 -544 -2249 | -544 -927 -1666 -2082 -2249 2727 |

Таблица 4: тесты для сортировки методом простого выбора

**Анализ допущенных ошибок**

В реализации алгоритмов ошибок допущено не было. Были исправлены некоторые неточности в функциях сравнения элементов и генерации массивов. Также были добавлены коммментарии.

**Список литературы**

[1]Лорин Г. Сортировка и системы сортировкию –М.:Наука, 1983.

[2]Вирт Н. Алгоритмы и структуры данных. –М.:Мир, 1989.