Министерство образования Республики Беларусь

Учреждение образования

“Брестский государственный технический университет”

Кафедра ИИТ

**Лабораторная работа №2**

По дисциплине: “Алгоритмы и структуры данных”

Тема: ”Алгоритмы сортировки”

Вариант №6

**Выполнил:**

Студент 1-го курса

Группы ПО-7

Комиссаров А.Е.

**Проверила:**

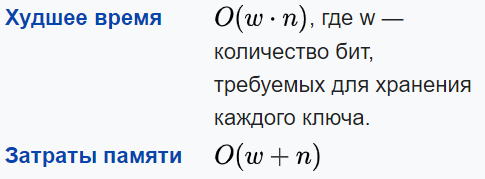
Глущенко Т.А.

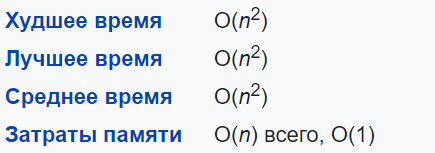
Брест, 2021

**Цель работы:** Изучение алгоритмов сортировки и определение временной сложности алгоритмов.

**Постановка задачи:** в соответствии с номером в журнале необходимо реализовать 3 алгоритма сортировки и провести экспериментальное исследование их эффективности.

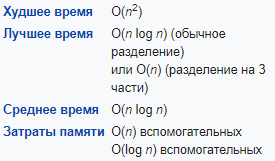
**Описание алгоритмов:**

1. Сортировка подсчётом - сравнение производится поразрядно: сначала сравниваются значения одного крайнего разряда, и элементы группируются по результатам этого сравнения, затем сравниваются значения следующего разряда, соседнего, и элементы либо упорядочиваются по результатам сравнения значений этого разряда внутри образованных на предыдущем проходе групп, либо переупорядочиваются в целом, но сохраняя относительный порядок, достигнутый при предыдущей сортировке. Затем аналогично делается для следующего разряда, и так до конца.
2. Сортировка выбором. Шаги алгоритма: находим номер минимального значения в текущем списке, производим обмен этого значения со значением первой неотсортированной позиции (обмен не нужен, если минимальный элемент уже находится на данной позиции), теперь сортируем хвост списка, исключив из рассмотрения уже отсортированные элементы.



1. Быстрая сортировка, сортировка Хоара (англ. quicksort), часто называемая qsort (по имени в стандартной библиотеке языка Си) — алгоритм сортировки, разработанный английским информатиком Тони Хоаром во время его работы в МГУ в 1960 году. Общая идея алгоритма состоит в следующем:

* Выбрать из массива элемент, называемый опорным. Это может быть любой из элементов массива. От выбора опорного элемента не зависит корректность алгоритма, но в отдельных случаях может сильно зависеть его эффективность (см. ниже).
* Сравнить все остальные элементы с опорным и переставить их в массиве так, чтобы разбить массив на три непрерывных отрезка, следующих друг за другом: «элементы меньшие опорного», «равные» и «большие».
* Для отрезков «меньших» и «больших» значений выполнить рекурсивно ту же последовательность операций, если длина отрезка больше единицы.



**Конфигурация машины на которой были произведены эксперименты:**

Процессор: AMD Phenom(tm) II X4 B50 Processor 3.21 GHz

Оперативная память: 8,00 ГБ

Тип системы: 64-разрядная Windows

**Результаты экспериментов:**

1. **Поразрядная сортировка**

Таблица результатов:

|  |  |  |
| --- | --- | --- |
| № | Количество элементов в массиве, n | Время выполнения алгоритма, с |
| 1 | 50 000 | 0.03 |
| 2 | 250 000 | 0.16 |
| 3 | 500 000 | 0.40 |
| 4 | 750 000 | 0.53 |
| 5 | 1 000 000 | 0.72 |

График результатов:

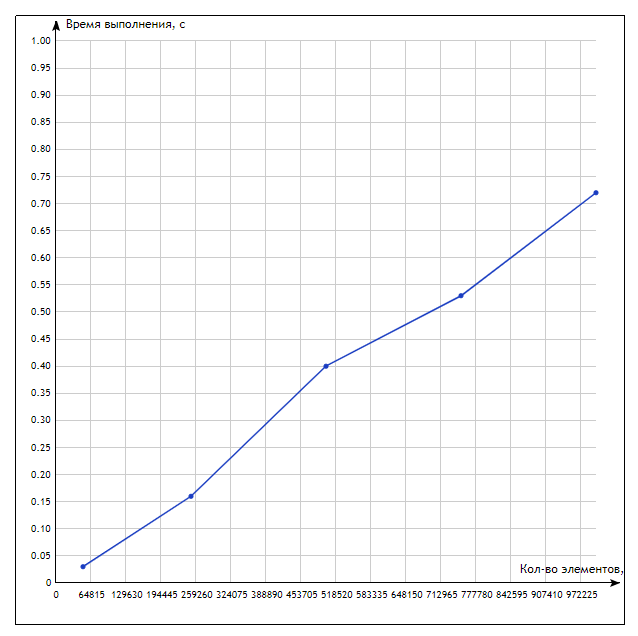
****

Рис. 1. Зависимость времени выполнения алгоритмa Radix Sort от количества элементов в массиве

1. **Сортировка выбором**

Таблица результатов:

|  |  |  |
| --- | --- | --- |
| № | Количество элементов в массиве, n | Время выполнения алгоритма, с |
| 1 | 10 000 | 0.16 |
| 2 | 25 000 | 1.15 |
| 3 | 50 000 | 4.28 |
| 4 | 75 000 | 9.48 |
| 5 | 100 000 | 16.61 |

График результатов:

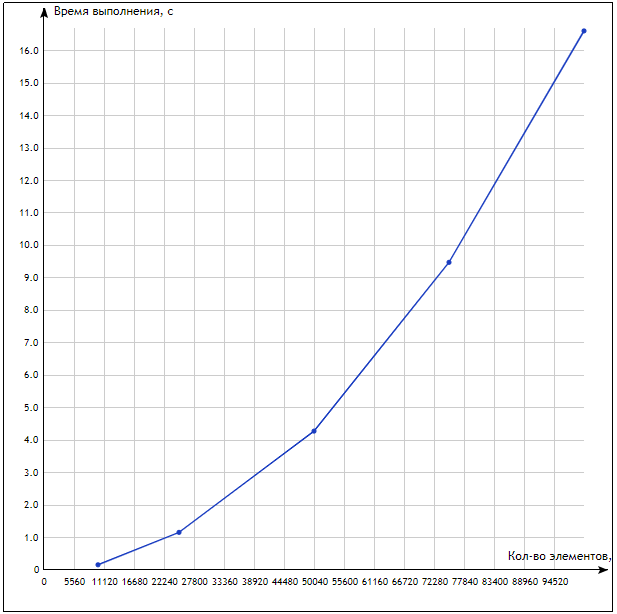


Рис. 2. Зависимость времени выполнения алгоритмa Choice Sort от количества элементов в массиве

1. **Сортировка Хоара \ Быстрая сортировка**

Таблица результатов:

|  |  |  |
| --- | --- | --- |
| № | Количество элементов в массиве, n | Время выполнения алгоритма, с |
| 1 | 50 000 | 0.008 |
| 2 | 250 000 | 0.05 |
| 3 | 500 000 | 0.1 |
| 4 | 1 000 000 | 0.2 |
| 5 | 10 000 000 | 2.12 |

График результатов:

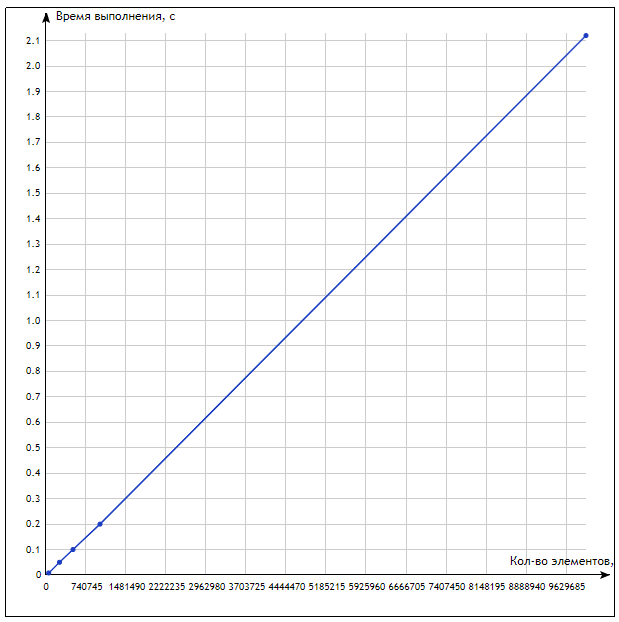


Рис. 3. Зависимость времени выполнения алгоритмa Quick Sort от количества элементов в массиве

**Совместный график трёх алгоритмов:**

(выше – хуже, медленней)

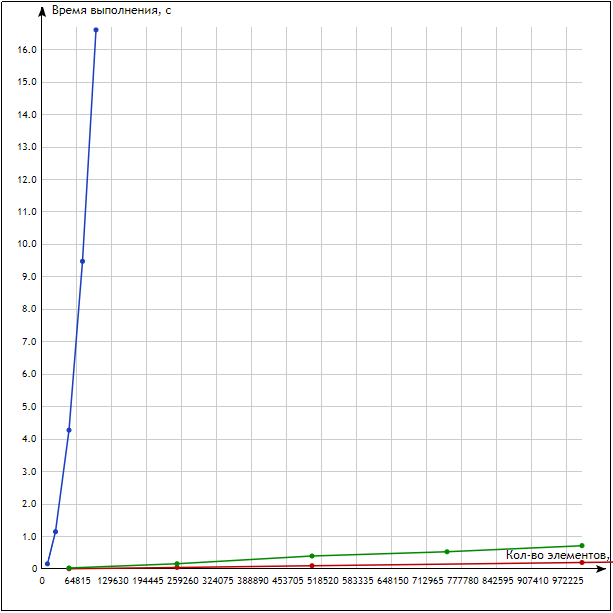
****

Рис. 4. Зависимость времени выполнения алгоритмов Quick Sort, Radix Sort, Choice Sort от количества элементов в массиве. Синий – сортировка выбором, зелёный – поразрядная сортировка, красный – сортировка Хоара \ быстрая сортировка.

**Bывод:** по результатам экспериментальной проверки, на первом месте в скорости выполнения алгоритма является быстрая сортировка, после неё идёт поразрядная сортировка, а самым медленным оказался алгоритм сортировки выбором.

Алгоритм быстрой сортировки является одним из самых быстрых известных универсальных алгоритмов сортировки массивов: в среднем O(n log n) обменов при упорядочении n элементов; из-за наличия ряда недостатков на практике обычно используется с некоторыми доработками.