

ПРАКТИЧЕСКАЯ РАБОТА №2

СОРТИРОВКА ЧИСЛОВОЙ ПОСЛЕДОВАТЕЛЬНОСТИ

Часть 2.1: УСОВЕРШЕНСТВОВАННЫЕ И БЫСТРЫЕ СОРТИРОВКИ. АСИМПТОТИЧЕСКАЯ СЛОЖНОСТЬ АЛГОРИТМА ПРОСТОЙ СОРТИРОВКИ

Цель: актуализировать знания алгоритмов сортировки числовой последовательности, получить практические умения и навыки по анализу вычислительной сложности алгоритмов сортировки и определению наиболее эффективного алгоритма.

Задание 1. Эмпирическая оценка эффективности простого, ускоренного и быстрого алгоритмов сортировки

Требования по выполнению задания

1. Разработать алгоритм *ускоренной* сортировки, определённой в варианте (приложение 1), реализовать код на языке C++. Сформировать таблицу 1.1 результатов эмпирической оценки сложности сортировки по формату табл. 1 для массива, заполненного случайными числами¹.
2. Определить ёмкостную сложность алгоритма ускоренной сортировки.

Таблица 1. Сводная таблица результатов

п	время, мс	C_n	M_n	$T_n = C_n + M_n$
100				
200				
500				
1000				
2000				
5000				
10000				
100000				
200000				
500000				
1000000				

3. Разработать алгоритм *быстрой* сортировки, определённой в варианте (приложение 1), реализовать код на языке C++. Сформировать таблицу 1.2 результатов эмпирической оценки сортировки по формату табл. 1 для массива, заполненного случайными числами.
4. Определить ёмкостную сложность алгоритма быстрой сортировки.
5. Добавьте в отчёт данные по работе любого из алгоритмов простой сортировки в среднем случае, полученные в предыдущей практической работе (в отчёте – таблица 1.3).

¹ Для соблюдения равенства условий при анализе эффективности, случайные массивы всех заданных длин должны быть одними и теми же для всех рассматриваемых алгоритмов.

6. Представить на общем сравнительном графике зависимости $T_n(n) = C_{\phi} + M_{\phi}$ для трёх анализируемых алгоритмов.
7. На основе сравнения полученных данных определите наиболее эффективный из трёх алгоритмов в среднем случае (отдельно для небольших массивов при n до 1000 и для больших массивов с $n > 1000$).
8. По аналогии с п.п. 1, 3, 5 и 6 проанализируйте эффективность простой, ускоренной и быстрой сортировок на массивах, отсортированных а) строго в убывающем и б) строго возрастающем порядке значений элементов. Заполнить по этим данным для каждого алгоритма соответствующие таблицы 1.4, 1.5, 1.6 по формату табл. 1, также сравнительный график.
9. Сделайте вывод о зависимости (или независимости) алгоритмов сортировок от исходной упорядоченности массива на основе результатов, представленных в таблицах.

Задание 2. Асимптотический анализ сложности алгоритмов

Требования по выполнению задания

1. Из материалов предыдущей практической работы приведите в отчёте формулы $T_r(n)$ функций роста алгоритма простой сортировки в лучшем и худшем случае (того же алгоритма, что и в задании 1).
2. На основе определений соответствующих нотаций получите асимптотическую оценку вычислительной сложности простого алгоритма сортировки²:
 - в O -нотации (оценка сверху) для анализа худшего случая;
 - в Ω -нотации (оценка снизу) для анализа лучшего случая.
3. Получите (если это возможно) асимптотически точную оценку вычислительной сложности алгоритма в нотации θ .
4. Реализуйте на одном графике визуальное представление функции роста $T(n)$ и полученных в п.2 асимптотических границ сверху $c_1 * f(n)$ и снизу $c_2 * g(n)$. Покажите на нём пороговое значение n_0 .
5. Найдите справочную информацию об асимптотической сложности усовершенствованного и быстрого алгоритмов сортировки, заданных в вашем варианте.

Таблица 2. Сводная таблица результатов

Алгоритм ³	Асимптотическая сложность алгоритма			
	Наихудший случай (сверху)	Наилучший случай (снизу)	Средний случай (точная оценка)	Ёмк-я сложн-ть
Простой				
Усовершенствованный				
Быстрый				

² Здесь требуется математически доказать существование коэффициентов, при которых истинны неравенства, лежащие в основе определения каждой нотации (O , Ω , θ).

³ В отчёте в этом столбце таблицы укажите названия соответствующих алгоритмов сортировки.

6. Общие результаты п.п. 2, 3 и 5 сведите в табл. 2. Сделайте вывод о наиболее эффективном алгоритме из трёх.

Отчёт

В отчёте по каждой сортировке необходимо привести словесное описание алгоритма и его блок-схему, а также программный код (с комментариями), результаты тестирования на массиве $n=10$ и контрольных прогонов на массивах длиной 100, 200, 500, 1000, 2000, 5000, 10000, 100000, 200000, 500000 и 1000000 элементов.

По итогам выполнения каждого задания сформулируйте соответствующие выводы.

Приложение 1. Варианты индивидуальных заданий.

Вариант	Усовершенствованный алгоритм	Быстрый алгоритм
1	Сортировка обмeнами с условием Айверсона	Простое слияние
2	Шейкерная сортировка с условием Айверсона	Простое слияние
3	Сортировка Шелла со сдвигами Хиббарда	Простое слияние
4	Сортировка Шелла со сдвигами Д. Кнута. Способ 1	Простое слияние
5	Шелла со сдвигами Д. Кнута. Способ 2	Простое слияние
6	Шелла со сдвигами Р. Седжвика.	Простое слияние
7	Пирамидальная сортировка	Простое слияние
8	Турнирная сортировка	Простое слияние
9	Сортировка Шелла со сдвигами Пратта	Простое слияние
10	Сортировка обмeнами с условием Айверсона	Быстрая сортировка (Хоара)
11	Шейкерная сортировка с условием Айверсона	Быстрая сортировка (Хоара)
12	Сортировка Шелла со сдвигами Хиббарда	Быстрая сортировка (Хоара)
13	Сортировка Шелла со сдвигами Д. Кнута. Способ 1	Быстрая сортировка (Хоара)
14	Шелла со сдвигами Д. Кнута. Способ 2	Быстрая сортировка (Хоара)
15	Шелла со сдвигами Р. Седжвика.	Быстрая сортировка (Хоара)
16	Пирамидальная сортировка	Быстрая сортировка (Хоара)

17	Турнирная сортировка	Быстрая сортировка (Хоара)
18	Сортировка Шелла со смещениями Пратта	Быстрая сортировка (Хоара)

Приложение 2. Методы определения смещения для сортировки Шелла, предложенные Д. Кнутом и Р. Седжвиком

Перед выполнением сортировки происходит вычисление длин промежутков (значения d из примера сортировки Шелла), которые записываются в массив, например, d .

По Седжвику

Значение смещения, записываемого в элемент массива d , вычисляется по формуле:

$$d[i] = \begin{cases} 9 * 2^i - 9 * 2^{i/2} + 1 & \text{при } i - \text{четном} \\ 8 * 2^i - 6 * 2^{(i+1)/2} + 1 & \text{при } i - \text{нечетном} \end{cases}$$

Остановить создание и заполнение массива d на значении $d[i-1]$, если $3 * d[i] > n$ (размера массива).

По Кнуту

Если t – количество смещений, то:

Способ 1:

$t = \log_3(n-1)$ $d_0 = 1$, $d[i-1] = 3 * d[i] + 1$ т.е. 1, 4, 13, 40, 121,

Способ 2:

$t = \log_2(n-1)$ $d_0 = 1$, $d[i-1] = 2 * d[i] + 1$ т.е. 1, 3, 7, 13, 31,