Задачи для 1 курса (2024-2025 уч. год.)

3. Задачи на сортировку массивов

Требуется реализовать и протестировать различные алгоритмы сортировки — медленные, быстрые, специальные. Тестирование означает следующее.

Для медленных и быстрых сортировок каждый алгоритм надо реализовать в трех вариантах:
1. для массива значений типа double
void Sort1(double *a, int n);
2. для массива значений типа double с заданной функцией сравнения
void Sort2(double *a, size_t n, int (*cmp)(double, double));
3. для массива произвольных объектов с заданной функцией сравнения (по типу qsort)
void Sort3(void *a, size_t n, size_t elem_length, int (*cmp)(const void *, const void *));

Для специальных сортировок обрабатывается массив unsigned int, поэтому остается только один вариант
void Sort(unsigned int *a, size_t n, bool ascending);
где параметр ascending определяет как сортировать — по возрастанию или убыванию.

Для каждой из этих реализаций надо измерить время сортировки одного и того же случайного массива длины N и вывести это время в виде сравнительной таблицы для всех реализаций.

Для быстрых и специальных сортировок нужно также сравнить время работы ваших реализаций с сортировкой того же массива с помощью библиотечной функции qsort.

Тест должен также проверять фактическую упорядоченность отсортированного массива.

Тест нужно провести для нескольких различных распределений исходных массивов, а именно, случайный набор, отсортированный в обратном направлении набор, "волнообразный" (несколько участков возрастания и убывания).

Время работы каждой сортировки замеряется с помощью функции clock() или любым другим подходящим способом.

В отчете по заданию должна быть представлена табличка с указанием метода сортировки, типа массива, длины массива и полученного времени работы. Например такого вида

Быстрая сортировка, однократная рекурсия, случайный double массив (секунды).

длина	10^6	$2 \cdot 10^6$	$4 \cdot 10^6$	$8 \cdot 10^6$	10^7
Sort1	0.011	0.025	0.061	0.135	0.144
Sort2	0.014	...	...	...	...
Sort3	0.019	...	...	...	...
qsort	0.009	...	...	...	...

и аналогичные таблички для всех остальных случаев.

Алгоритмы медленной сортировки:

1. Простая сортировка обменами (перестановка максимума).
2. Пузырьковая восходящая сортировка.
3. Сортировка просеиванием (нисходящая пузырьковая).
4. Сортировка просеиванием с бинарным поиском позиции вставки.
5. Сортировка Шелла с простейшим вариантом выбора подпоследовательностей.

Алгоритмы "быстрой" сортировки:

6. Сортировка слиянием с доп. памятью $O(N)$ — рекурсивный нисходящий вариант.
7. Сортировка слиянием с доп. памятью $O(N)$ — итерационный восходящий вариант.
8. Сортировка слиянием без доп. памяти с трудоемкостью $O(N \log^2 N)$.
9. Быстрая сортировка (quicksort) с двойной рекурсией.
10. Быстрая сортировка (quicksort) с одинарной рекурсией.
11. Пирамидальная сортировка (heapsort).

"Специальные" сортировки:

12. Сортировка подсчетом для целого массива (с фактическим перемещением элементов).
13. Сортировка массива целых чисел битовым аналогом быстрой сортировки.
14. Сортировка массива целых чисел байтовым radix методом с сортировкой подсчетом.
15. Сортировка массива целых чисел полубайтовым radix методом с сортировкой подсчетом.
16. Сортировка массива целых чисел битовым radix методом.