Задача A3b

id посылки: 349351630
github: https://github.com/kurokolover/algorithms_hm/tree/main/A3_solution
(все файлы с данными из A2 находятся в https://github.com/kurokolover/algorithms_hm/tree/main/A2_solution

1 Этап 1. Подготовка тестовых данных

по условию задачи A3b используются те же тестовые данные, что и в задаче A2. тестовые массивы имеют следующие характеристики:

- Типы массивов:
  - массивы **рандомных чисел** (равномерно случайные значения);
  - массивы, **отсортированные по невозрастанию**;
  - **почти отсортированные** массивы: исходный массив отсортирован по неубыванию, после чего вносится небольшое количество случайных перестановок.
- Размеры массивов:
$$N \in \{500, 600, 700, \ldots, 9900, 10000\},$$
то есть от 500 до 10000 элементов с шагом 100.
- Для каждого размера N и каждого типа массива генерируется несколько независимых экземпляров (например, 5–10). Время сортировки усредняется по этим запускам.

Для задачи A2 дополнительно исследовались пороговые значения длины подмассива для перехода на INSERTION SORT (столбцы `30`, `50`, `100`, `125`, `170` в файле `help_list.txt`) и соответствующие времена работы гибридного алгоритма при различных порогах. В рамках задачи A3b эти же сгенерированные массивы используются в качестве входных данных для сравнения двух алгоритмов: стандартного QUICK SORT и INTROSORT.

2 Этап 2. Эмпирический анализ стандартного алгоритма QUICK SORT

Для стандартной реализации QUICK SORT проводятся замеры времени сортировки для всех подготовленных массивов.

2.1 Рандомные числа

-

Таблица 1: Время работы QUICK SORT на массивах рандомных чисел

Размер N	Время, мс (среднее)	Дисперсия/СКО (при необходимости)
500	`[данные]`	`[данные]`
600	`[данные]`	`[данные]`
⋮	⋮	⋮
10000	`[данные]`	`[данные]`

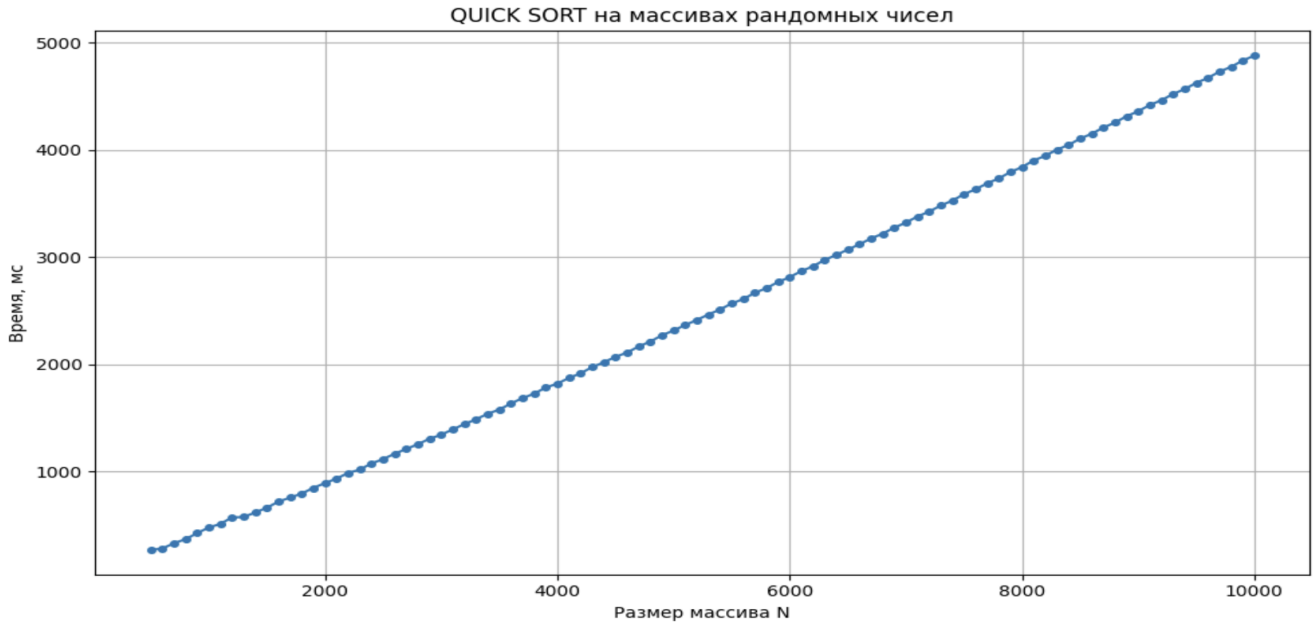


Рис. 1: Зависимость времени работы QUICK SORT от размера массива (рандомные числа)

2.2 Массивы, отсортированные по невозрастанию

-

Таблица 2: Время работы QUICK SORT на массивах, отсортированных по невозрастанию

Размер N	Время, мс (среднее)	Дисперсия/СКО (при необходимости)
500	[данные]	[данные]
600	[данные]	[данные]
⋮	⋮	⋮
10000	[данные]	[данные]

2.3 Почти отсортированные массивы

Таблица 3: Время работы QUICK SORT на почти отсортированных массивах

Размер N	Время, мс (среднее)	Дисперсия/СКО (при необходимости)
500	[данные]	[данные]
600	[данные]	[данные]
⋮	⋮	⋮
10000	[данные]	[данные]

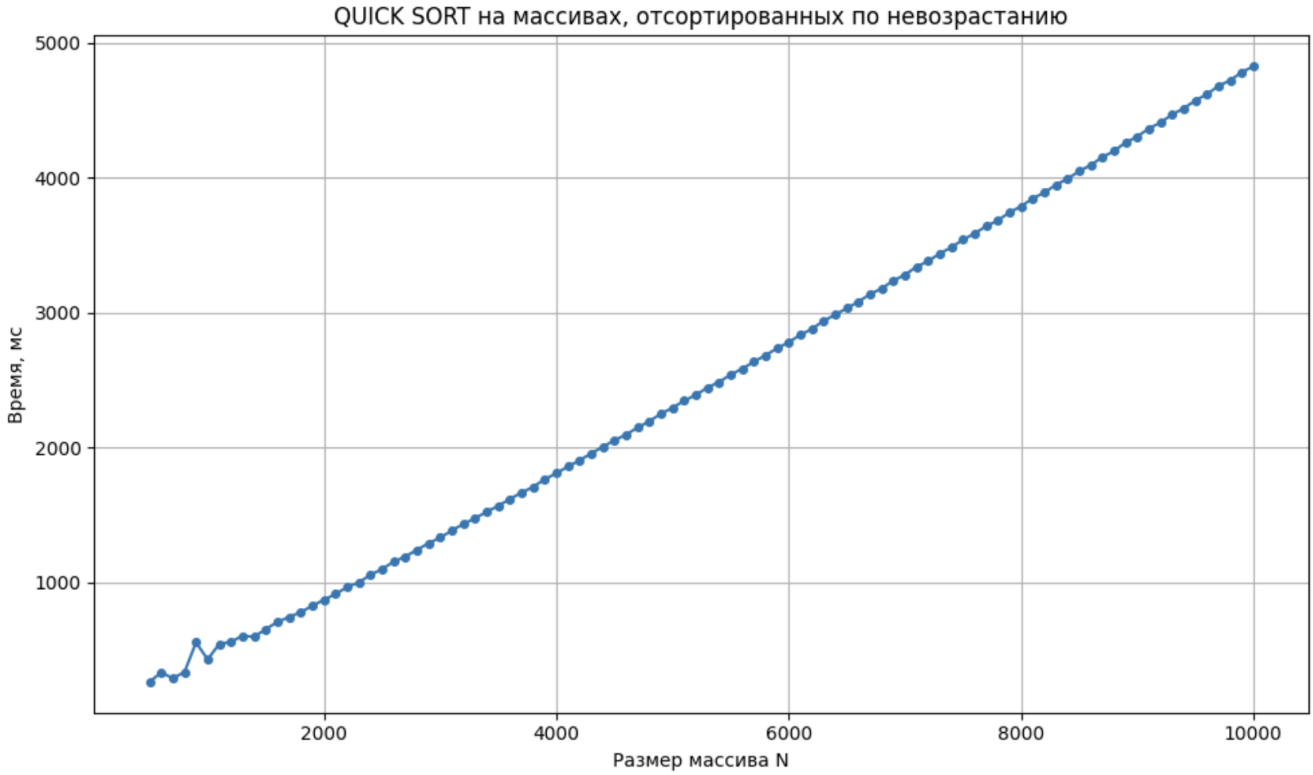


Рис. 2: Зависимость времени работы QUICK SORT от размера массива (невозрастающий порядок)

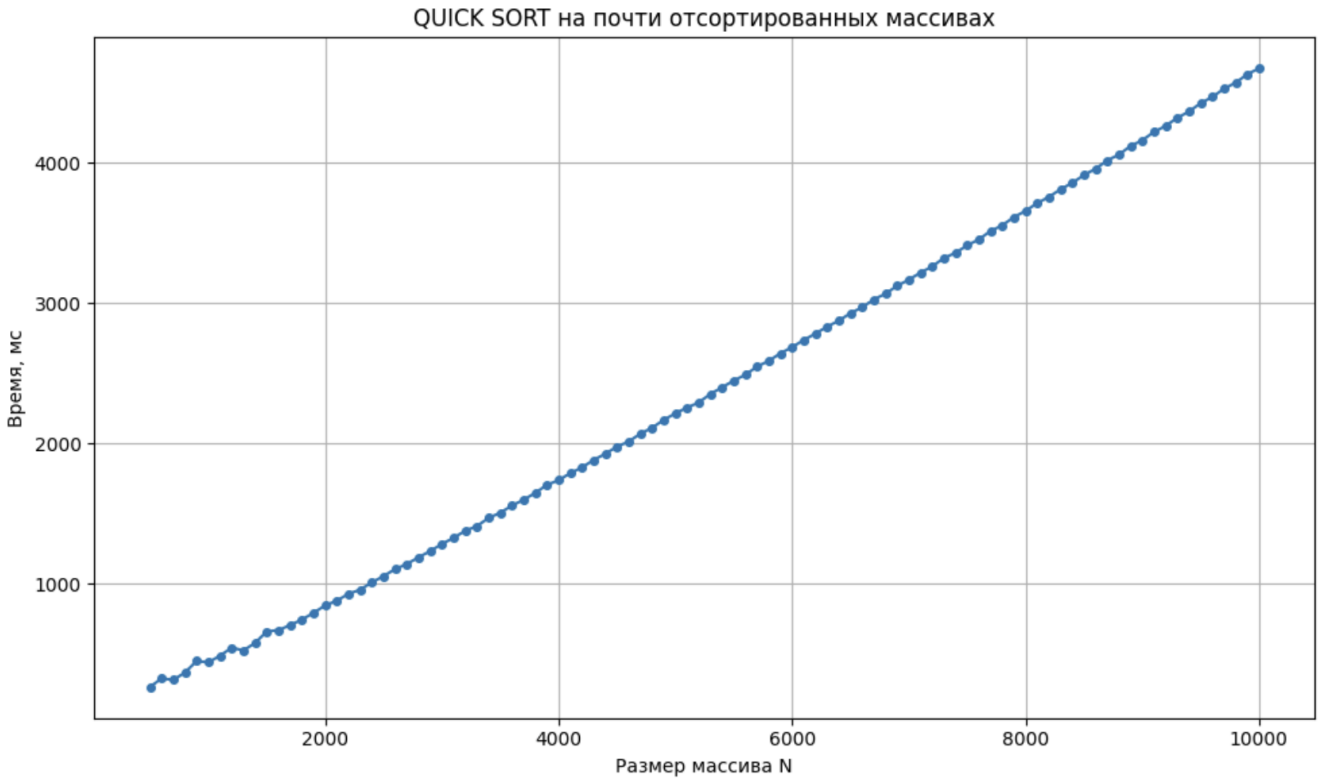


Рис. 3: Зависимость времени работы QUICK SORT от размера почти отсортированного массива

3 Этап 3. Эмпирический анализ гибридного QUICK+HEAP+I SORT

Аналогичные измерения проводятся для гибридного алгоритма INTROSORT. Те же самые массивы (рандомные, отсортированные по невозрастанию и почти отсортированные) сортируются с использованием INTROSORT.

3.1 Рандомные числа

Таблица 4: Время работы INTROSORT на массивах рандомных чисел

Размер N	Время, мс (среднее)	Дисперсия/СКО (при необходимости)
500	[данные]	[данные]
600	[данные]	[данные]
⋮	⋮	⋮
10000	[данные]	[данные]

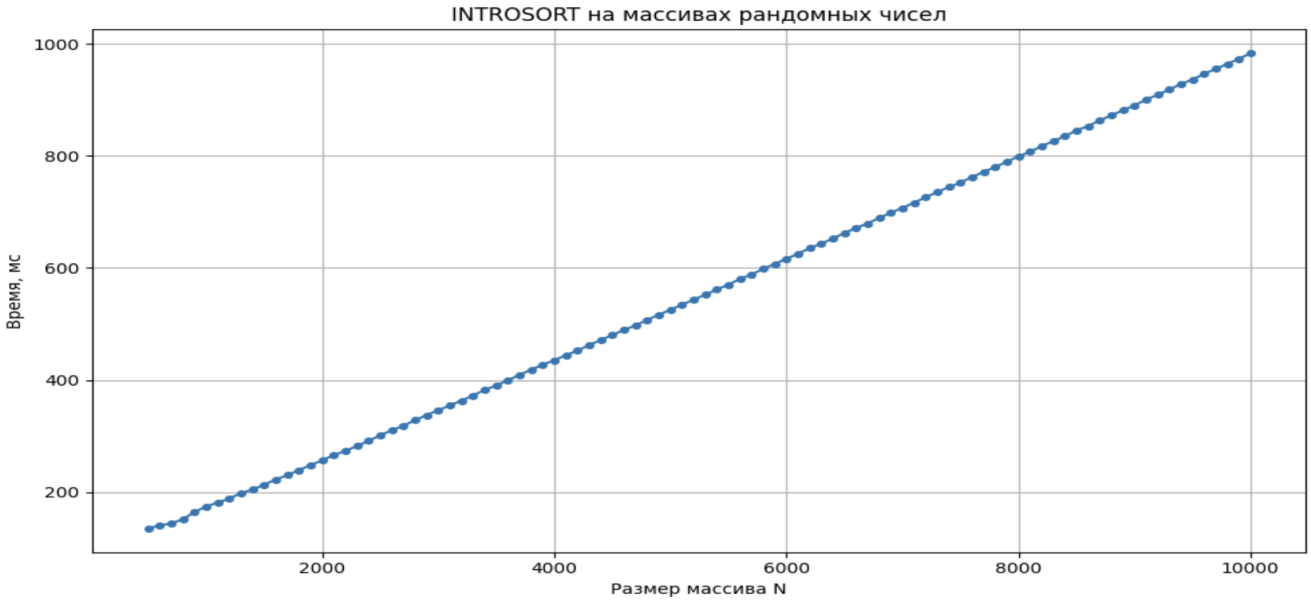


Рис. 4: Время работы INTROSORT на массивах рандомных чисел

3.2 Массивы, отсортированные по невозрастанию

Таблица 5: Время работы INTROSORT на массивах, отсортированных по невозрастанию

Размер N	Время, мс (среднее)	Дисперсия/СКО (при необходимости)
500	[данные]	[данные]
600	[данные]	[данные]
⋮	⋮	⋮
10000	[данные]	[данные]

3.3 Почти отсортированные массивы

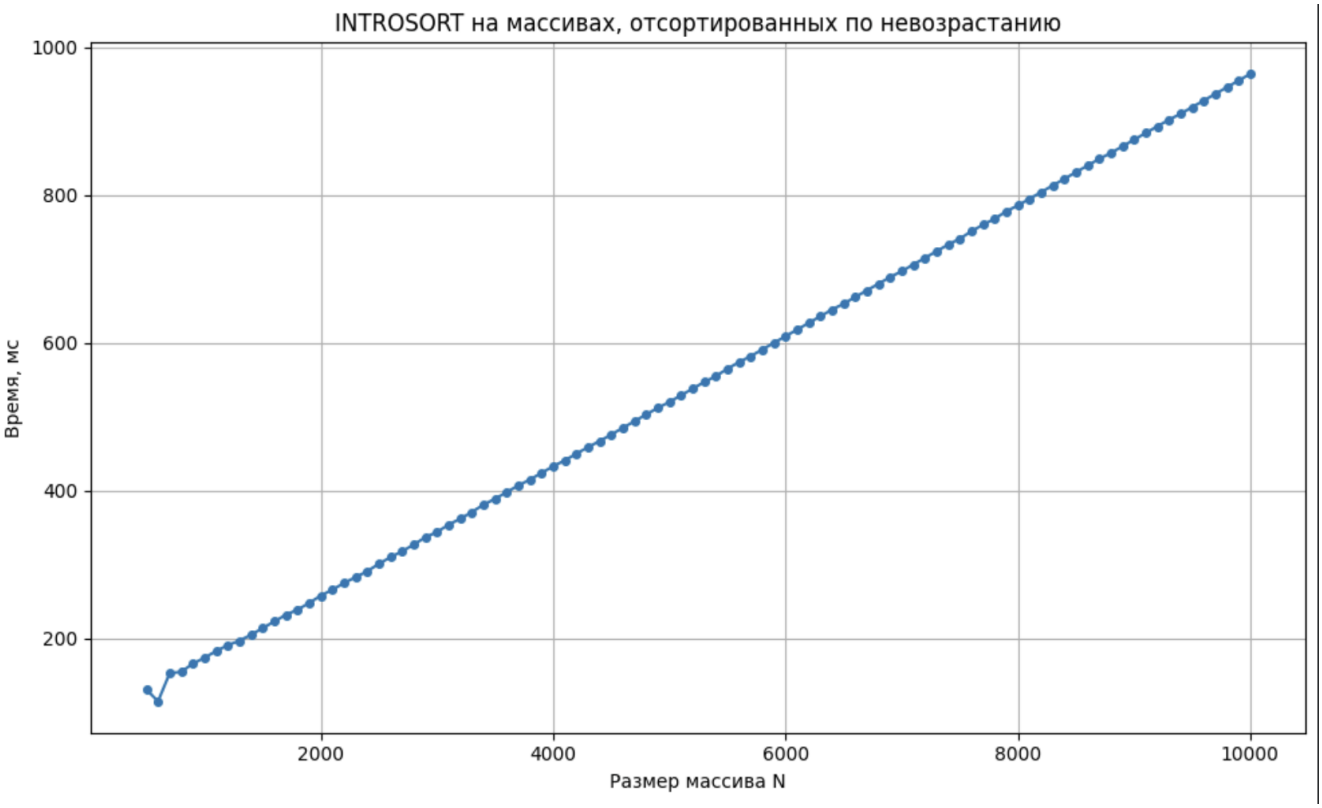


Рис. 5: Время работы INTROSORT на массивах, отсортированных по невозрастанию

Таблица 6: Время работы INTROSORT на почти отсортированных массивах

Размер N	Время, мс (среднее)	Дисперсия/СКО (при необходимости)
500	[данные]	[данные]
600	[данные]	[данные]
⋮	⋮	⋮
10000	[данные]	[данные]

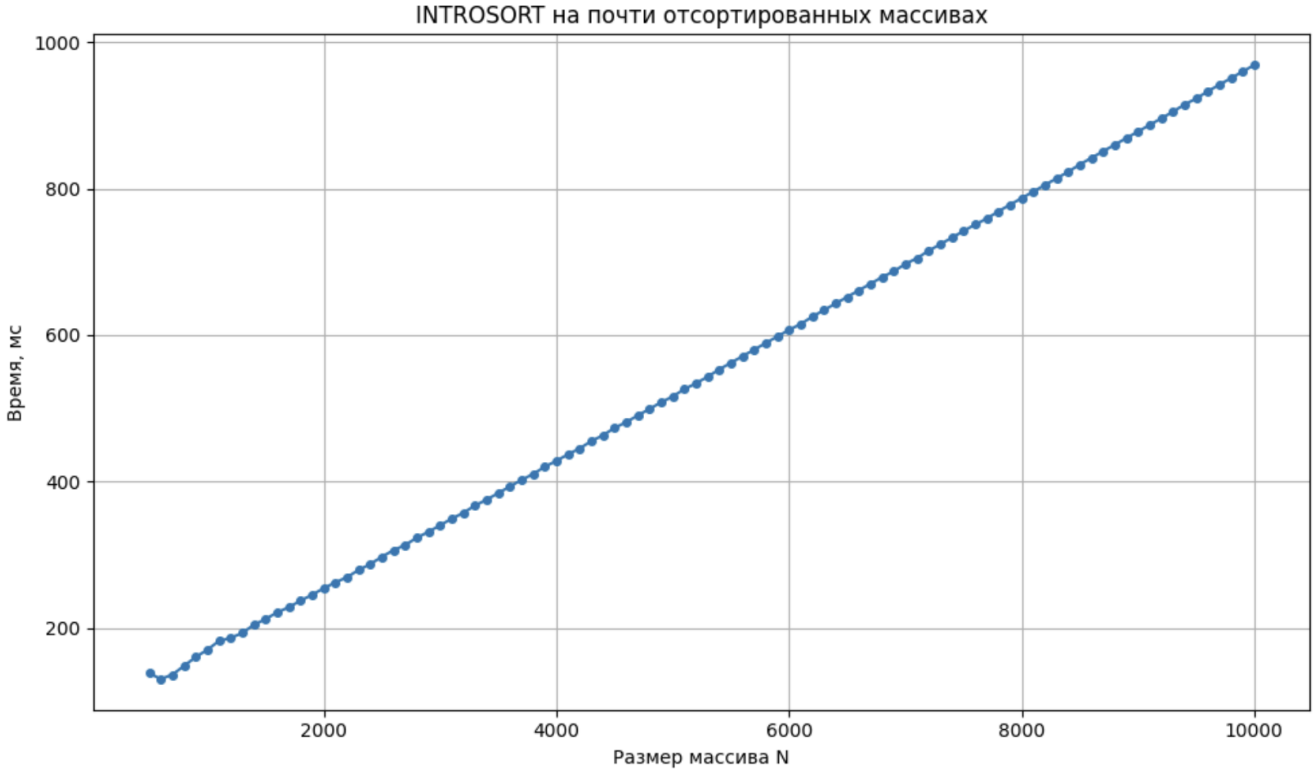


Рис. 6: Время работы INTROSORT на почти отсортированных массивах

4 Этап 4. Сравнительный анализ

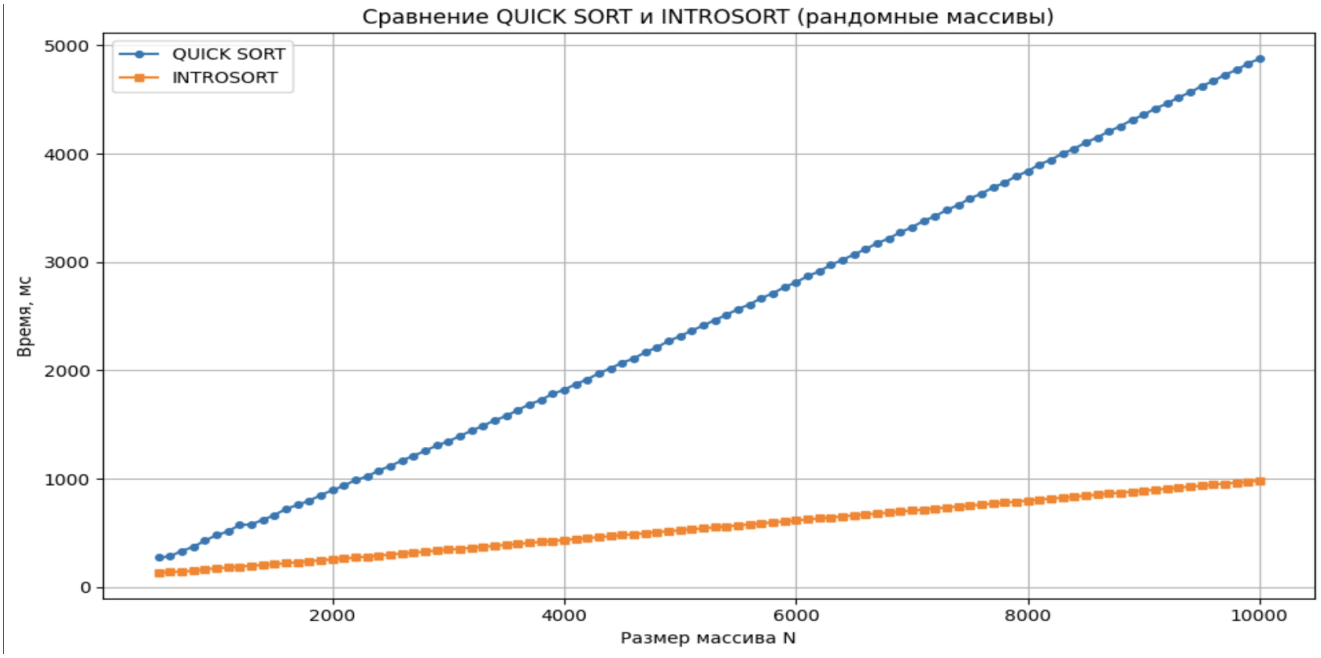


Рис. 7: Сравнение QUICK SORT и INTROSORT на массивах рандомных чисел

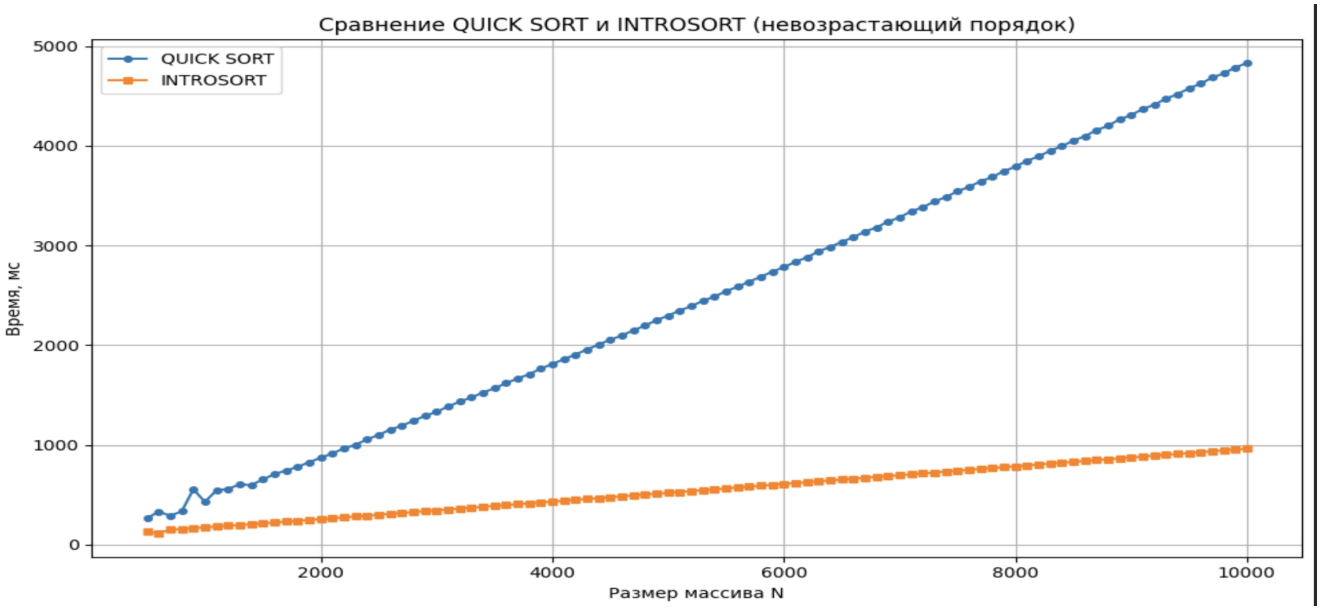
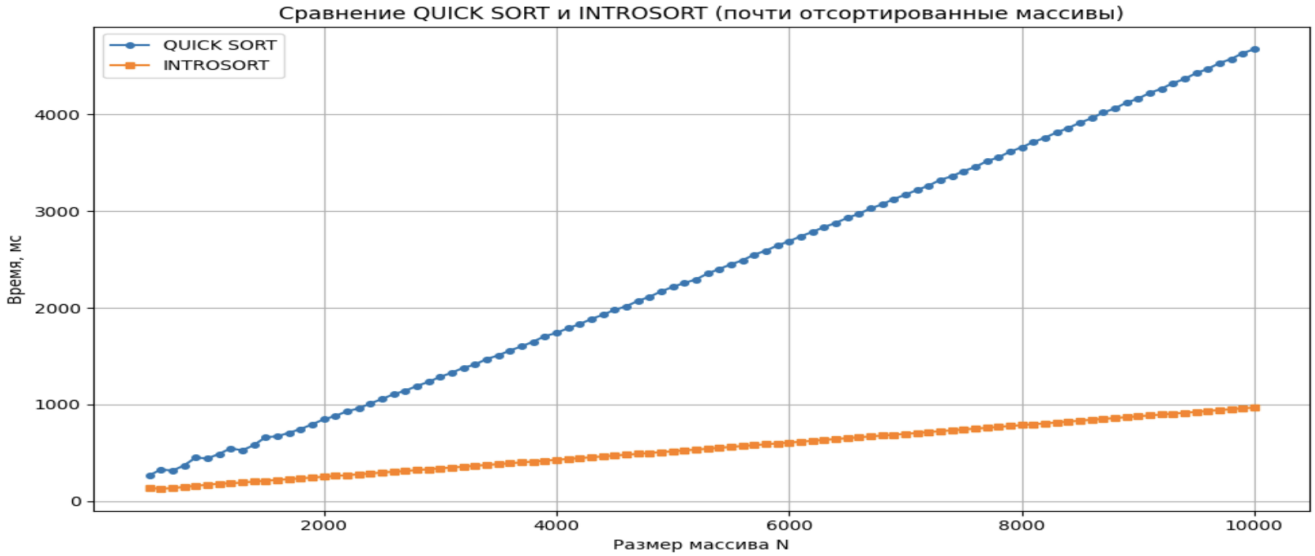


Рис. 8: Сравнение QUICK SORT и INTROSORT на массивах, отсортированных по невозрастанию



"Рис. 9: Сравнение QUICK SORT и INTROSORT на почти отсортированных массивах"

- На массивах рандомных чисел оба алгоритма демонстрируют близкое время работы и рост, близкий к $O(N \log N)$.
- На массивах, отсортированных по невозрастанию, INTROSORT показывает более стабильное время работы благодаря контролю глубины рекурсии и переходу на HEAP SORT.
- На почти отсортированных массивах выигрыш INTROSORT объясняется использованием INSERTION SORT на малых подмассивах.

Заключение

В работе реализован и исследован гибридный алгоритм сортировки QUICK+HEAP+INSERTION

SORT (INTROSORT). Проведено сравнение с классической реализацией QUICK SORT на трёх категориях входных данных: рандомные числа, массивы, отсортированные по невозрастанию, и почти отсортированные массивы. Полученные результаты подтверждают, что INTROSORT обеспечивает время работы, близкое к оптимальному для всех рассмотренных типов массивов.