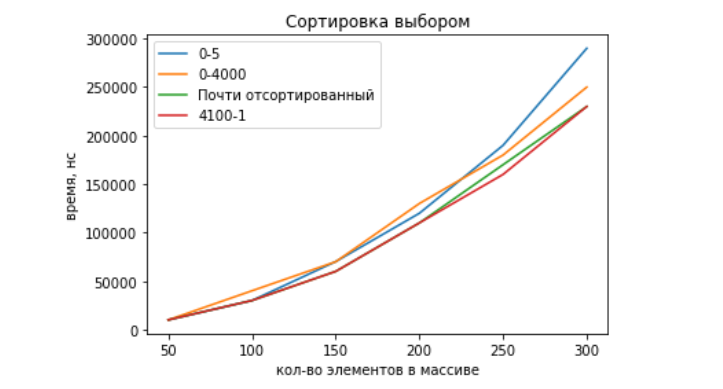
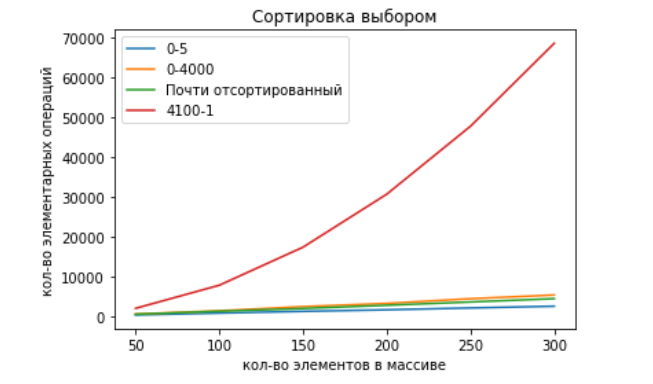
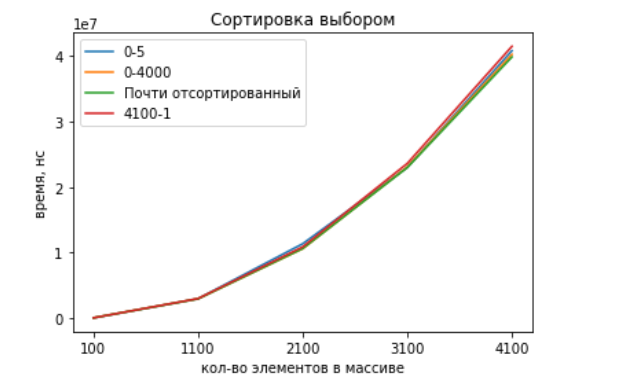
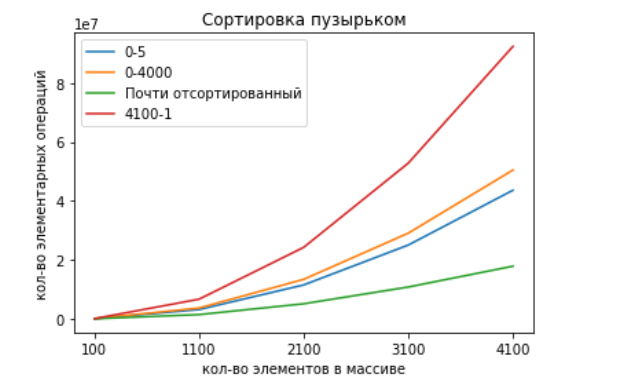
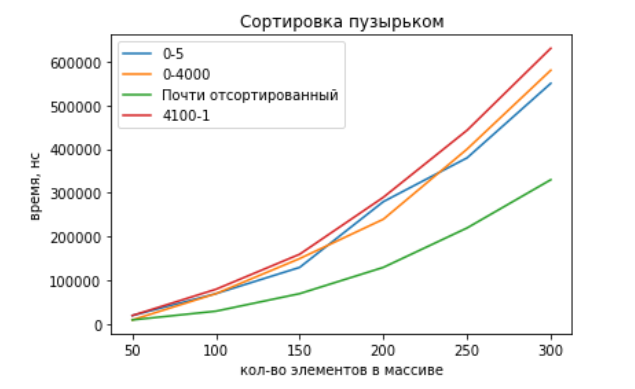
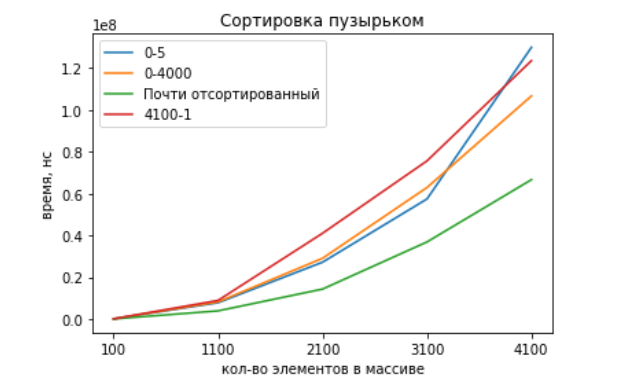
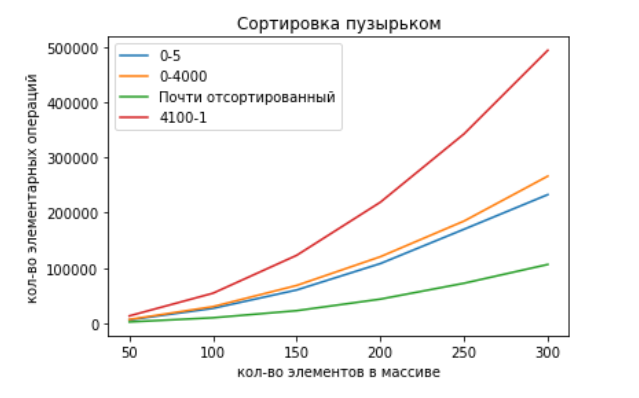
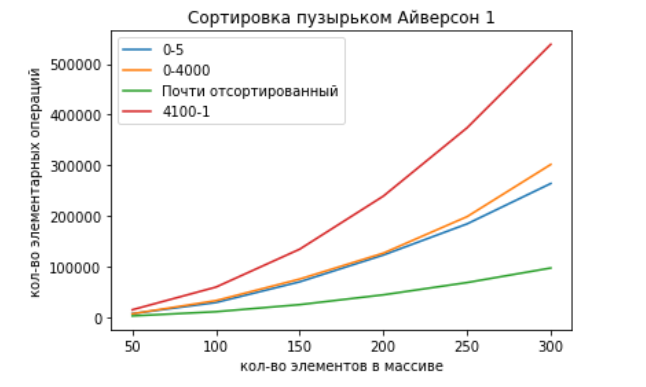
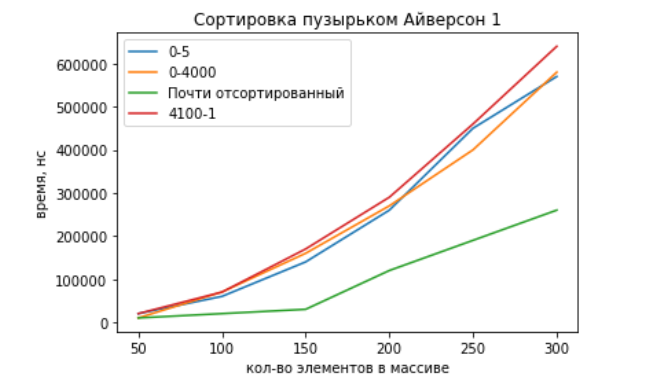
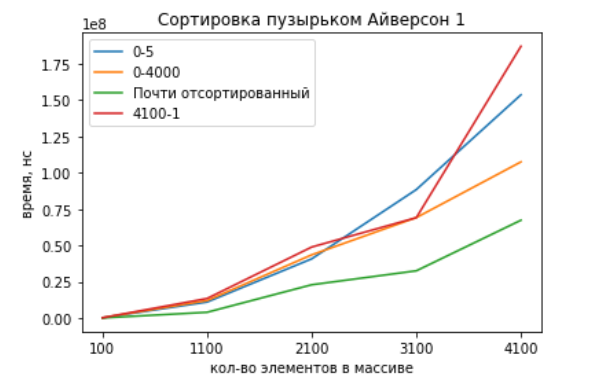
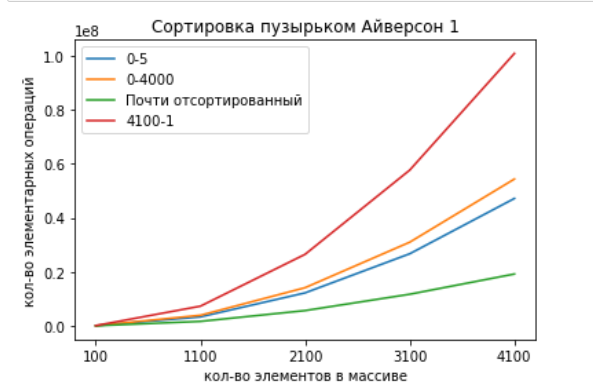
**Сортировка выбором**

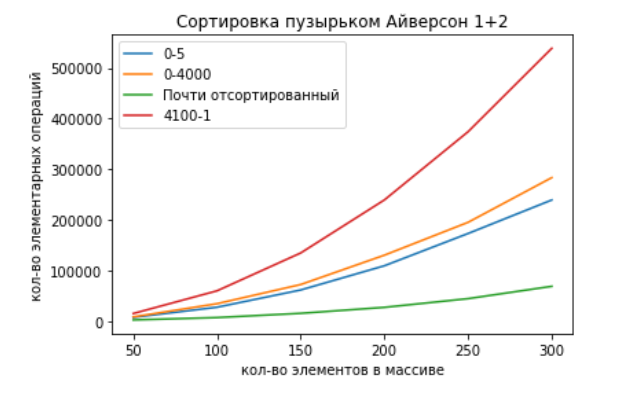
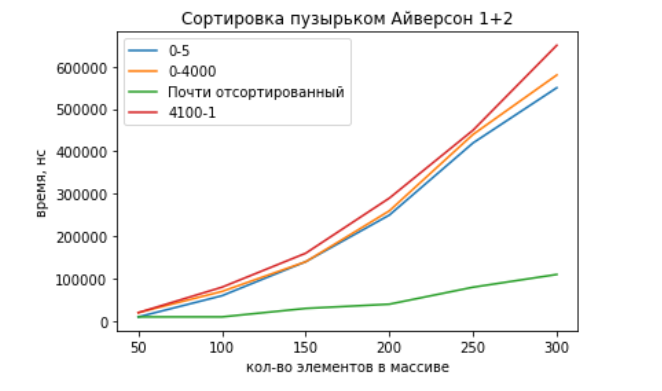
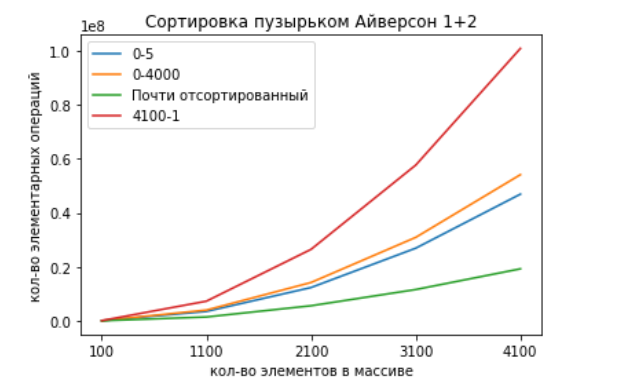
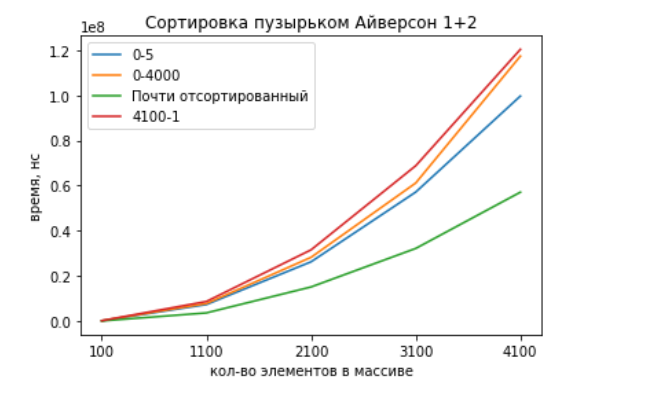
* Худший случай – обратно отсортированный массив, поэтому количество операций так сильно выделяется при нем.
* Асимптотическая сложность в среднем O(n^2), поэтому худший случай на время выполнения почти не влияет
* ****Вывод: Теоретические данные подтвердились экспериментально, худший случай правильно распознан

**Сортировка пузырьком**

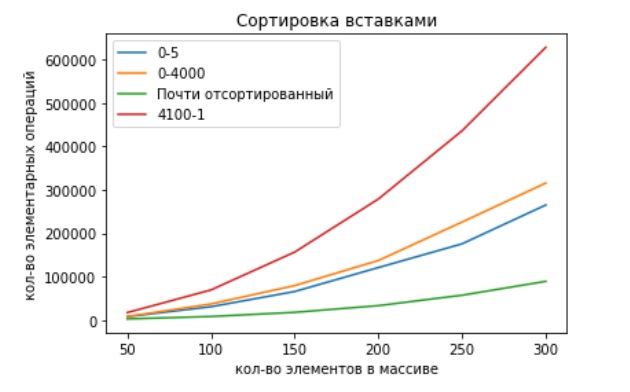
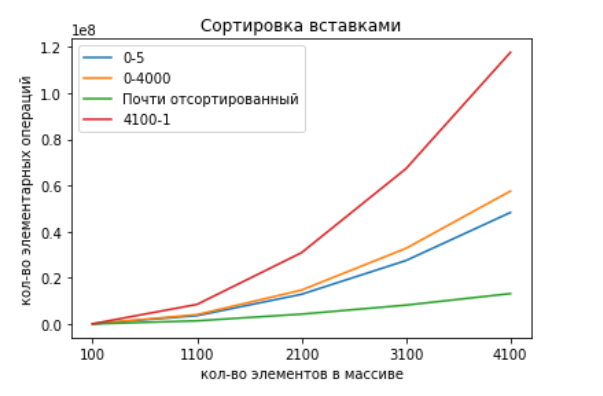
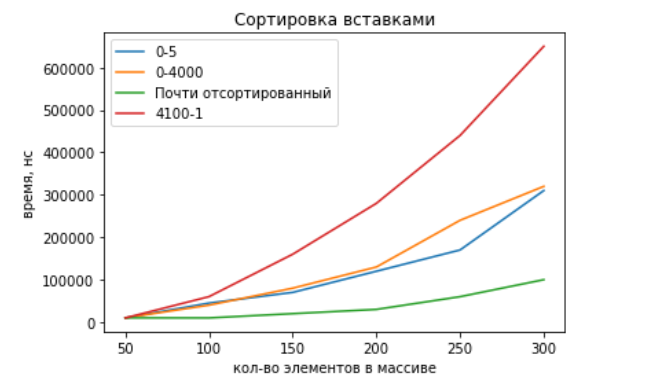
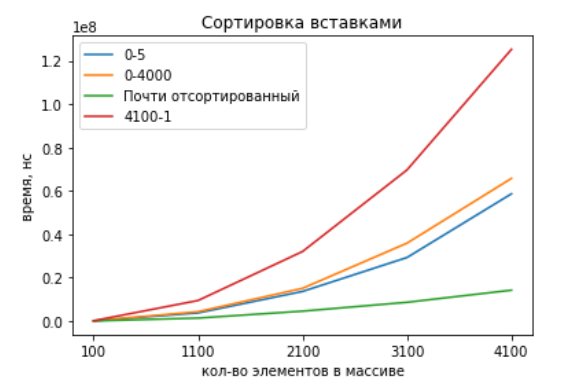
* Худший случай – обратно отсортированный массив, поэтому количество операций выделяется при нем.
* Лучший случай – почти отсортированный массив, что видно из графика
* Асимптотическая сложность в среднем O(n^2), поэтому худший случай на время выполнения почти не влияет
* Вывод: Теоретические данные подтвердились экспериментально, худший и лучший случаи правильно распознаны

**Сортировка пузырьком Айверсон 1**

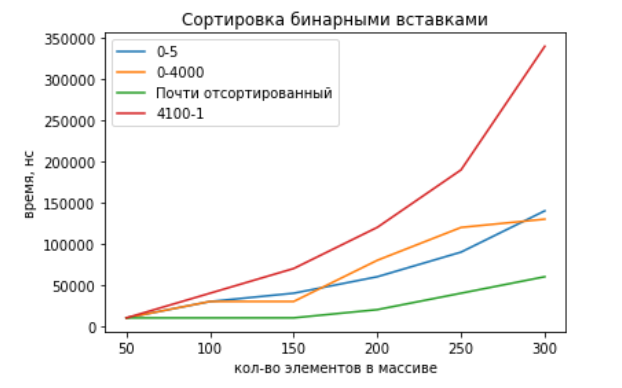
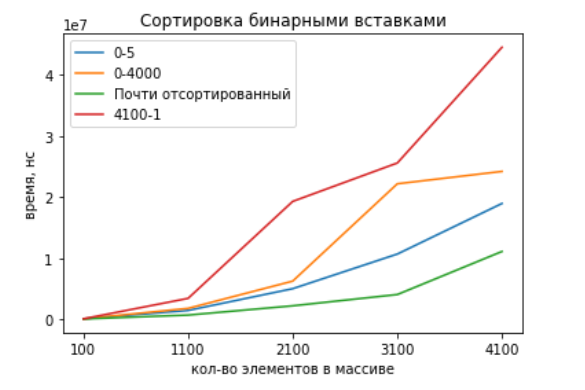
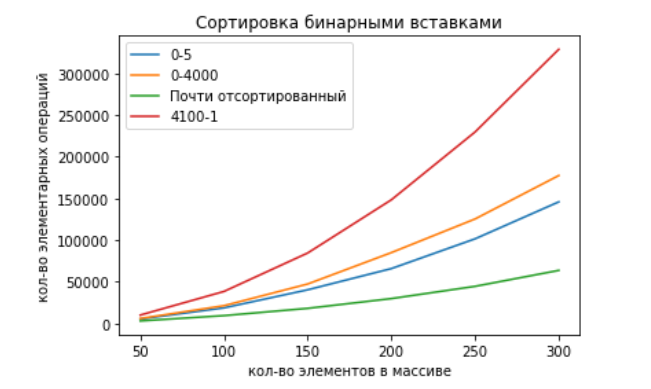
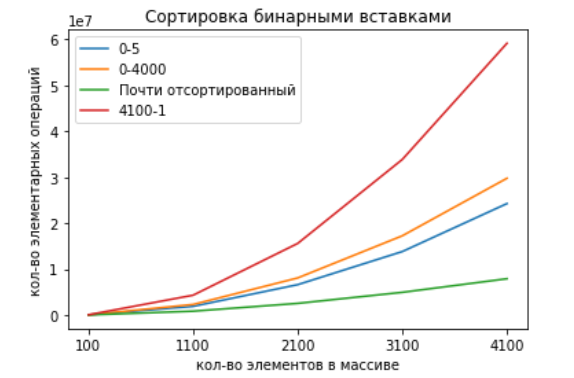
* Выводы аналогичны обычному пузырьку, однако в лучшем случае на отсортированном массиве сортировка работает за O(n). Это влияет и на время, и на число операций

**Сортировка пузырьком Айверсон 1+2**

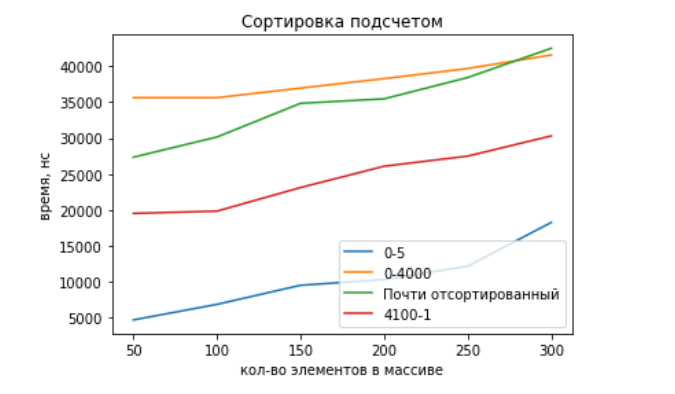
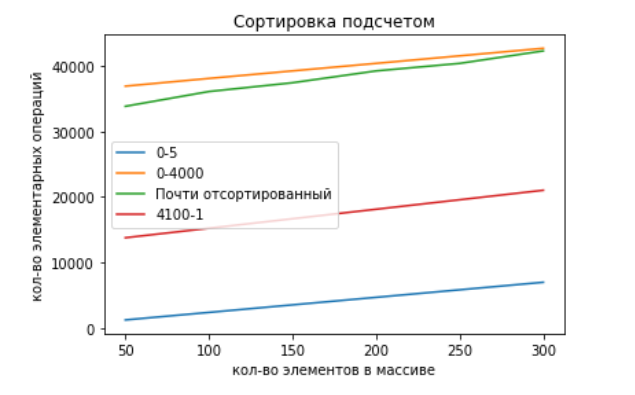
* Асимптотика аналогична Айверсону 1, однако уменьшено число обменов, а значит и число элементарных операций
* Вывод: данная оптимизация является лучшей из трех видов сортировок пузырьком, что подтвердилось экспериментально

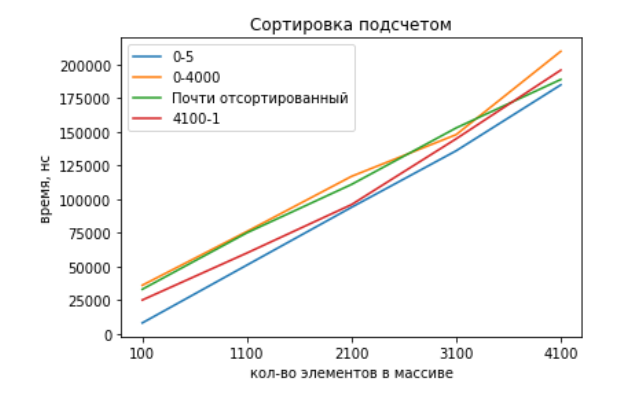
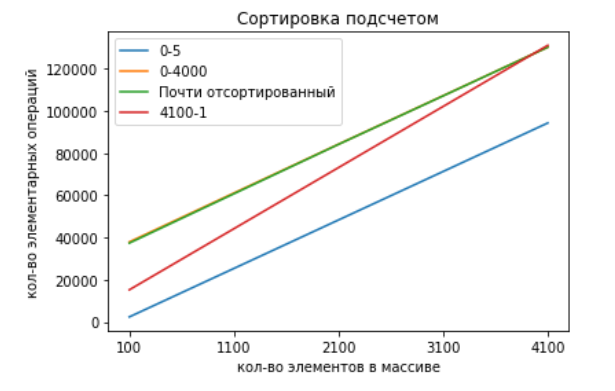
**Сортировка вставками**

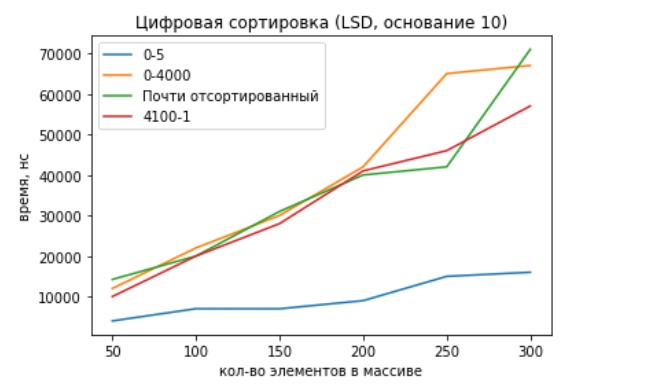
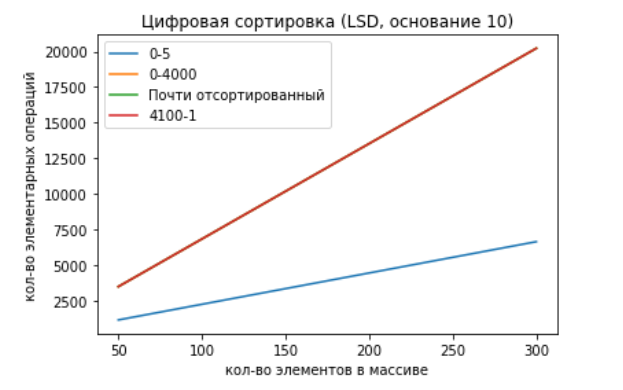
* Худший случай – обратно отсортированный массив, что подтвердилось экспериментально
* Вывод: теоретические данные подтверждены экспериментально

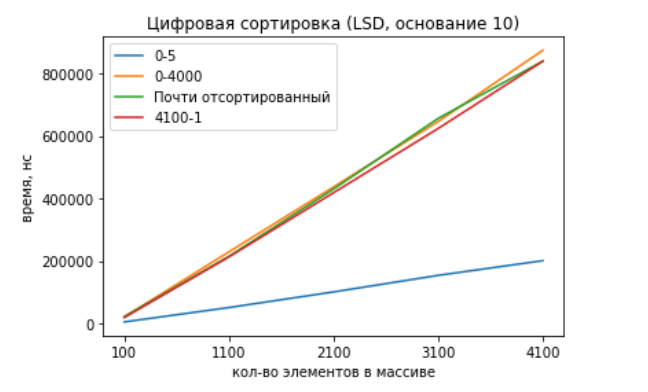
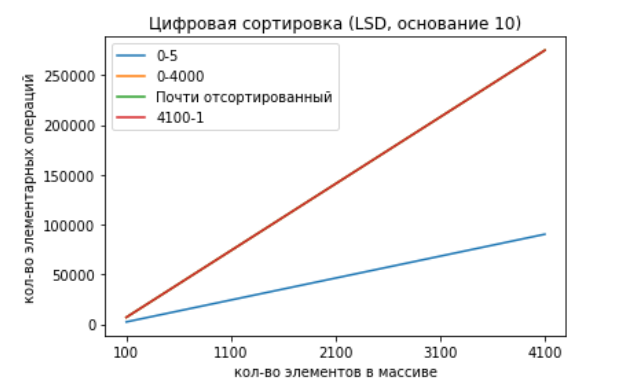
**Сортировка бинарными вставками**

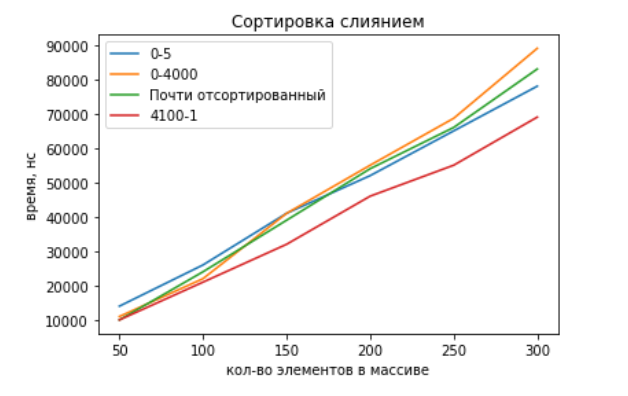
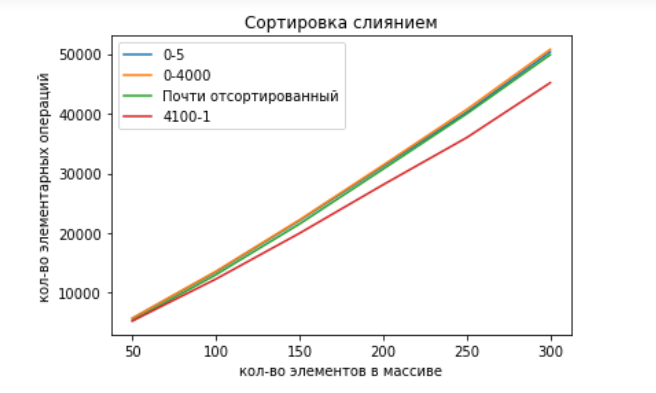
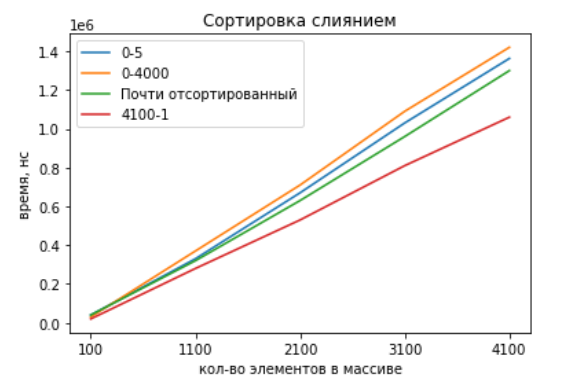
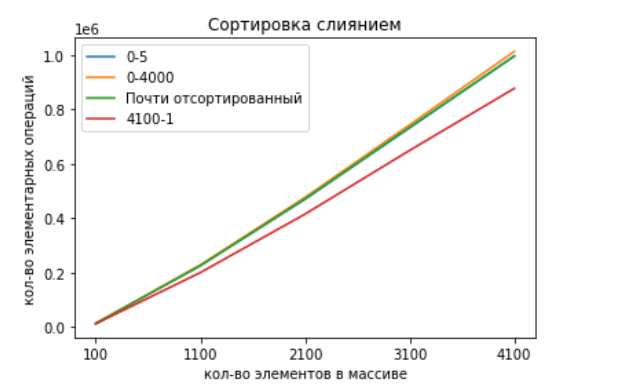
* Вывод: скачки на графиках обусловлены работой счетчика для элементарных операций. Время сортировки по сравнению с обычными вставками не должно было измениться, однако мы видим, что сортировка стала чуть быстрее

**Сортировка подсчетом**

* 0-5 самая быстрая, так как там всего 6 различных элементов.
* Дальше 4100-1, потому что там фиксированное число элементов, зависящее от размера массива (если размер 50, то различных элементов тоже 50).
* 0-4000 и почти отсортированный массив примерно на одном уровне, так как и там и там идет разброс элементов от 0 до 4000.
* Прямые не похожи на y=x в связи с неправильными пропорциями осей (по y разброс намного больший чем по x. Чтобы график был похож на y=x, необходимо для каждого размера строить отдельный график).
* График с количеством элементарных операций еще раз подтверждает все сказанное выше.
* На бОльших размерах массива график принимает свой нормальный вид, однако по количеству элементарных операций все равно можно проследить разницу в работе в зависимости от вида массива.
* Вывод: Сортировка подсчетом очень чувствительна к максимальному числу разрядов чисел в массиве, однако взамен она дает очень хорошую асимптотику

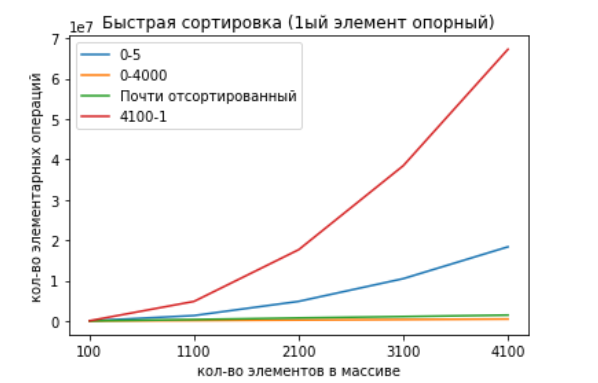
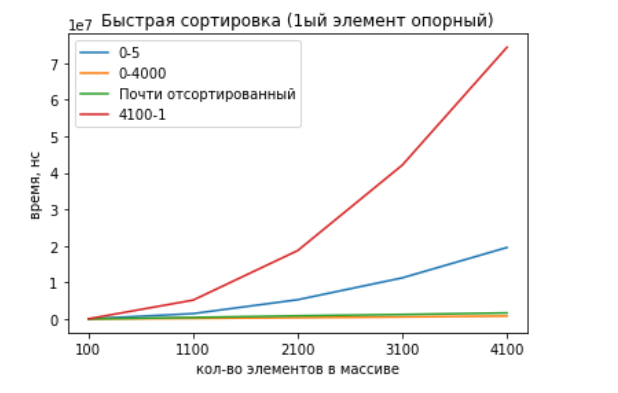
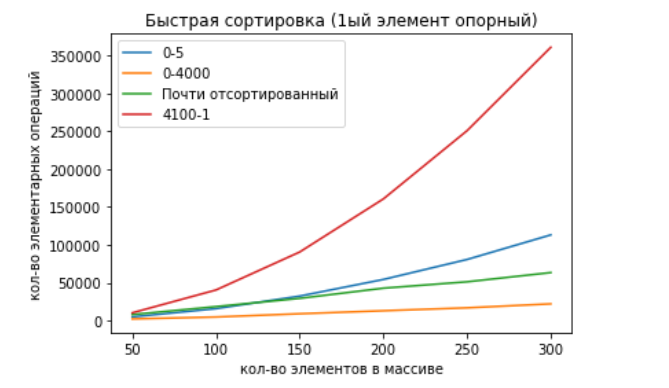
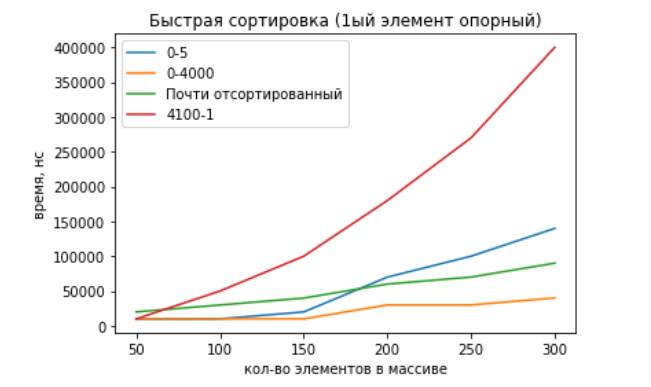
**Цифровая сортировка (LSD, основание 10)**

* 3 из 4 сортировок работают примерно за одно время. Выделяется только 0-5, так как при подобном заполнении массива все значения будут состоять только из 1 разряда.
* Количество элементарных операций у всех сортировок кроме 0-5 одинаковое, так как во всех них одинаковое максимальное число разрядов – 4.
* На более больших данных графики получились более красивыми 😊
* Объяснения все те же, что и для маленьких размеров массива
* Вывод: Цифровая сортировка очень чувствительна к максимальному числу разрядов чисел в массиве, однако взамен она дает очень хорошую асимптотику

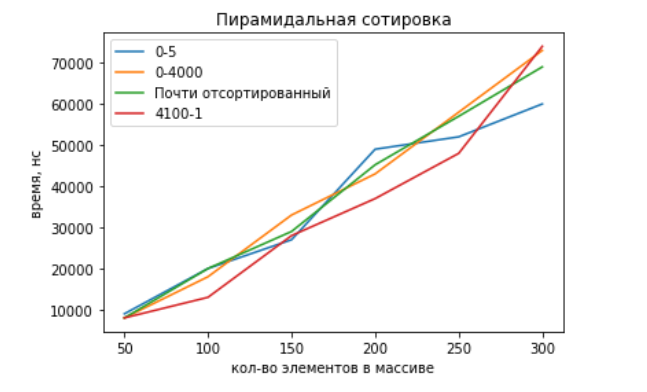
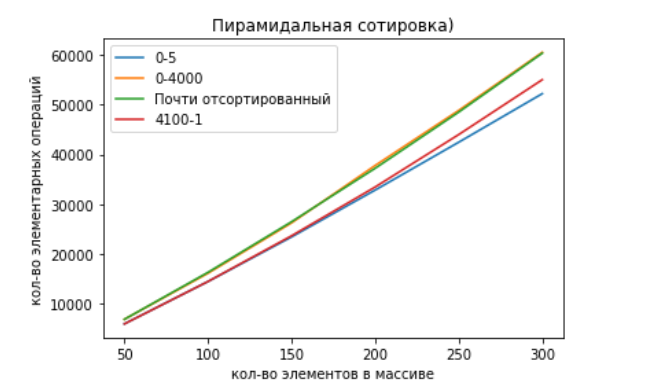
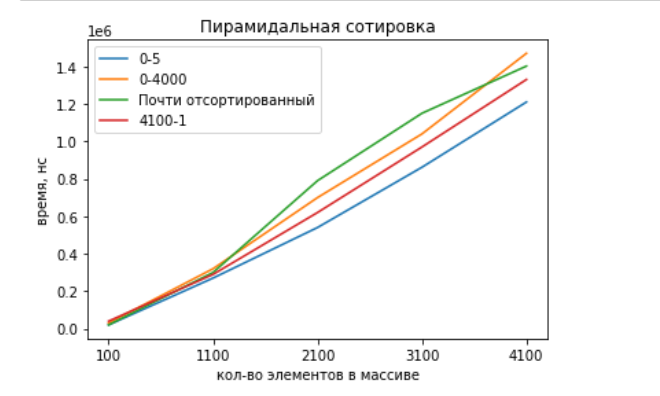
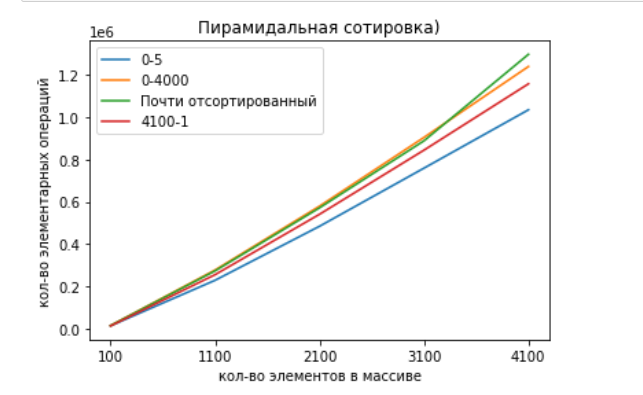
**Сортировка слиянием**

* Вывод: сортировка слиянием дает хорошую асимптотику. Теоретически, число операций не должно меняться в зависимости от данных массива, однако по графикам мы видим, что небольшое различие все-таки есть

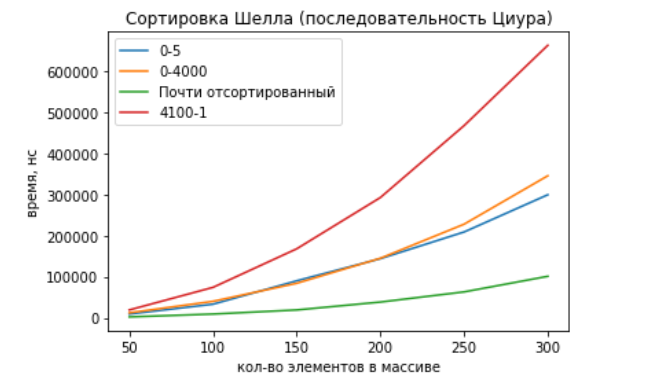
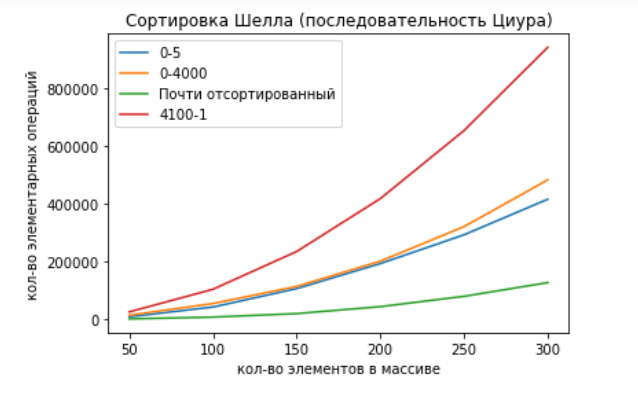
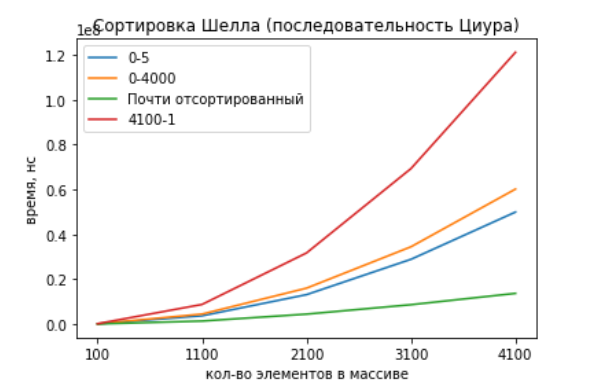
**Быстрая сортировка**

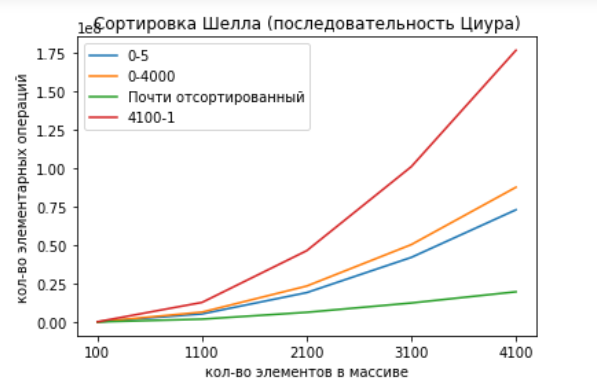
********

* Худший случай – обратно отсортированный массив, поэтому на нем и время и число операций увеличивается, что видно из графиков. Асимптотика в таком случае O(n^2)
* Вывод: В среднем быстрая сортировка дает хороший результат, однако на почти сортированных и маленьких массивах ее не стоит применять

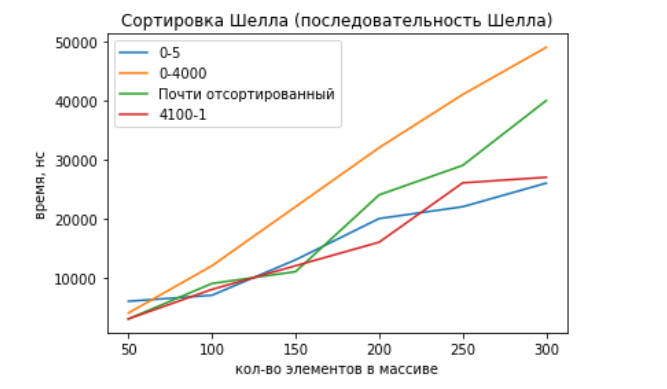
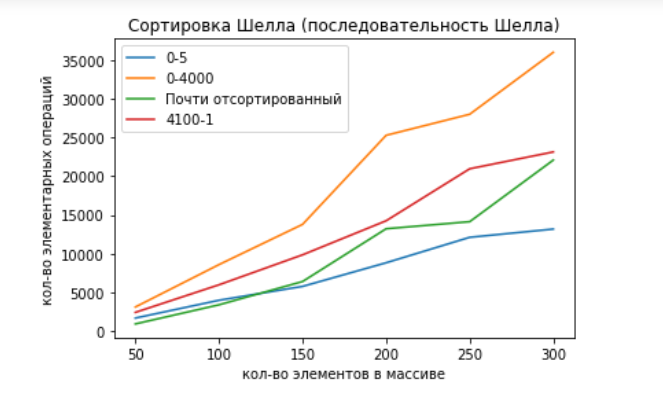
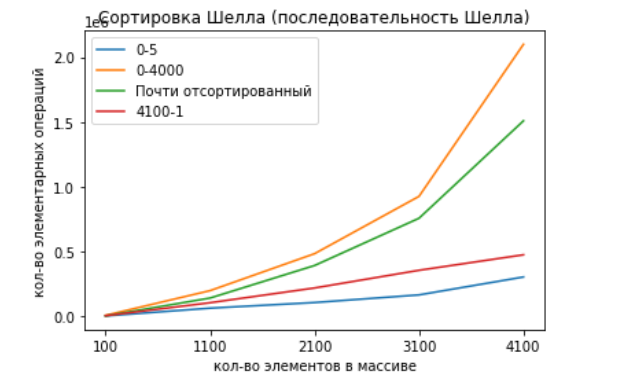
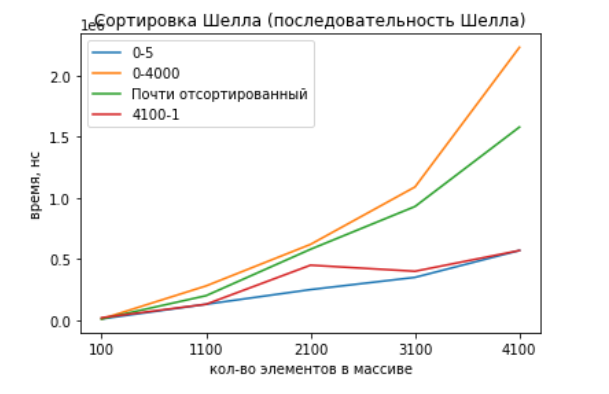
**Пирамидальная сортировка**

* На обратно отсортированном массиве пирамидальная сортировка должна работать лучше, однако на графике не видно существенных улучшений в данном случае
* Вывод: Теоретические данные не полностью подтвердились. Это может быть связано с погрешностью от изменения элементарных операций

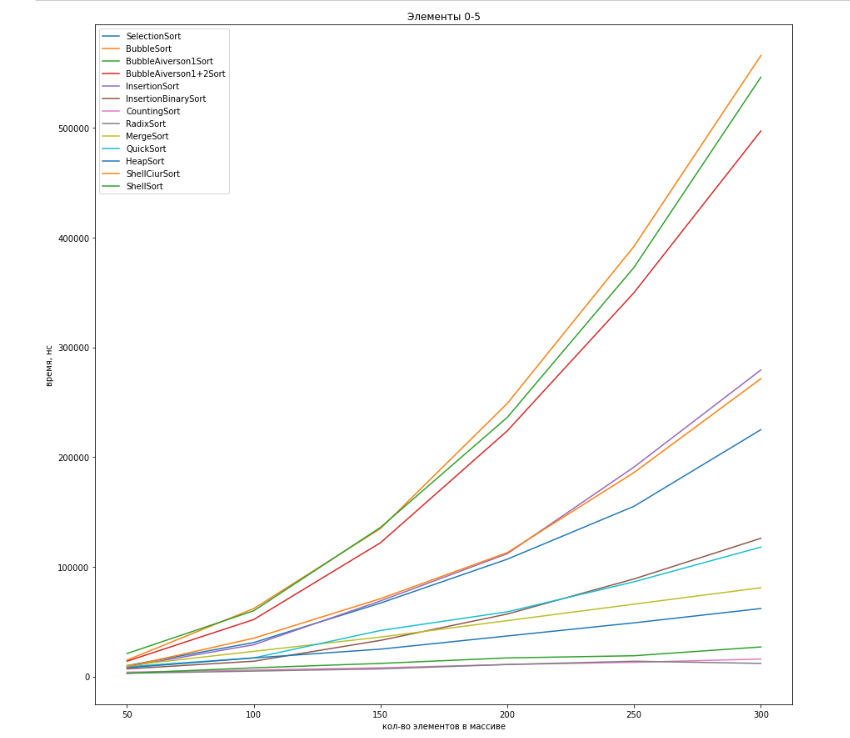
**Сортировка Шелла (последовательность Циура)**

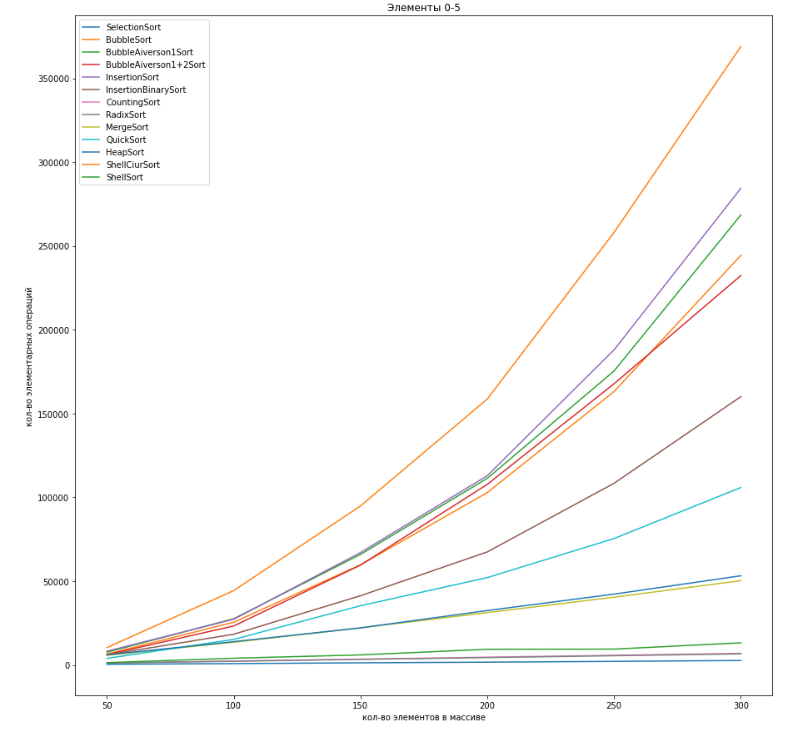
****

* Худший случай – большое число инверсий, что видно из графиков
* Лучший случай – отсортированный или почти отсортированный массив, что тоже видно из графиков
* Вывод: Теоретические данные подтвердились экспериментально

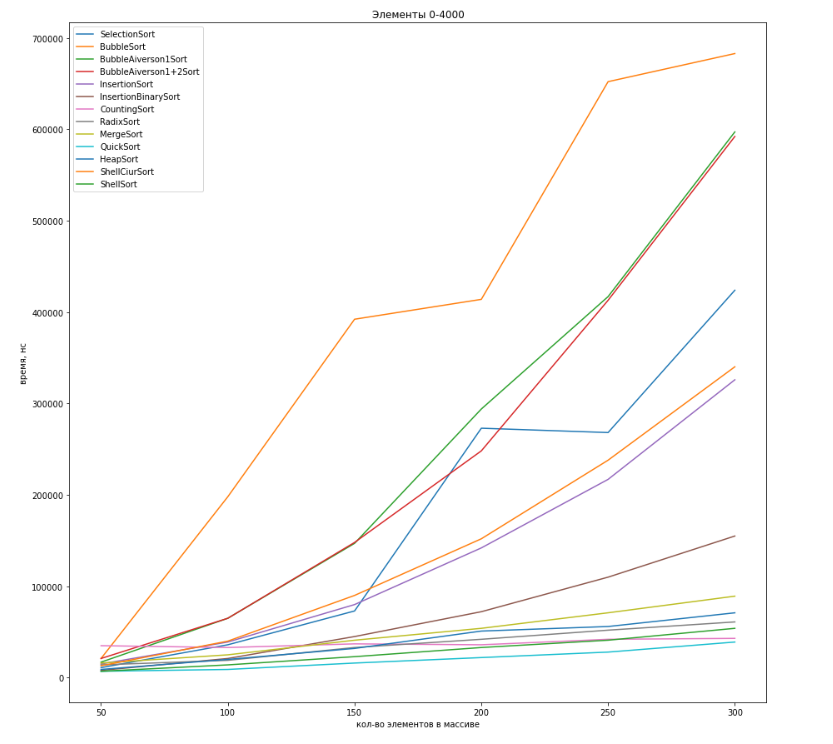
**Сортировка Шелла (последовательность Шелла)**

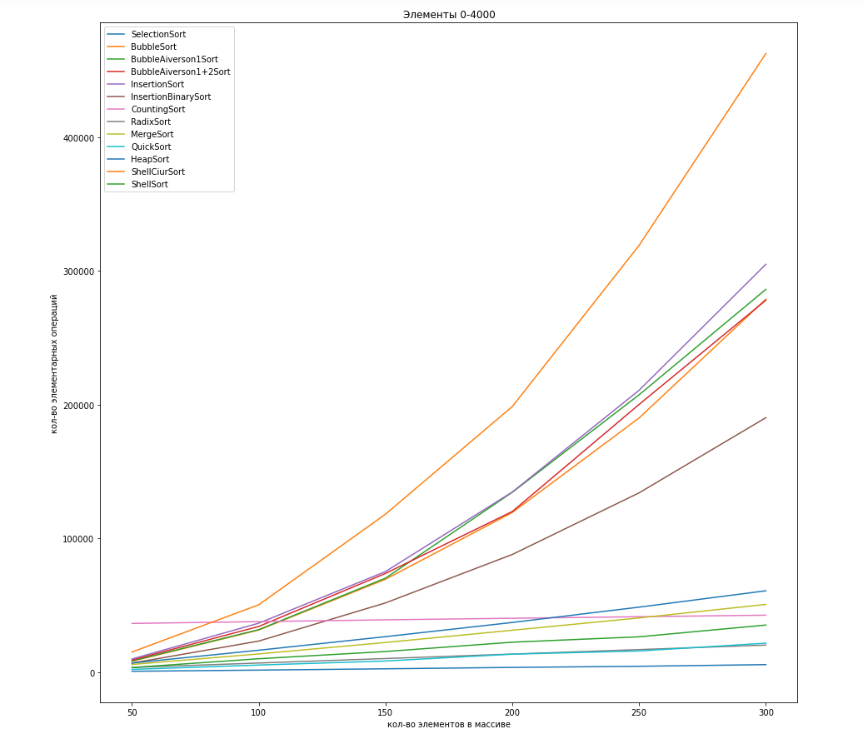
* Худший случай – большое число инверсий
* Лучший случай – отсортированный или почти отсортированный массив
* Вывод: Теоретические данные не подтвердились экспериментально. Возможно, не хватило большего усреднения данных

**50-300, тип 1**

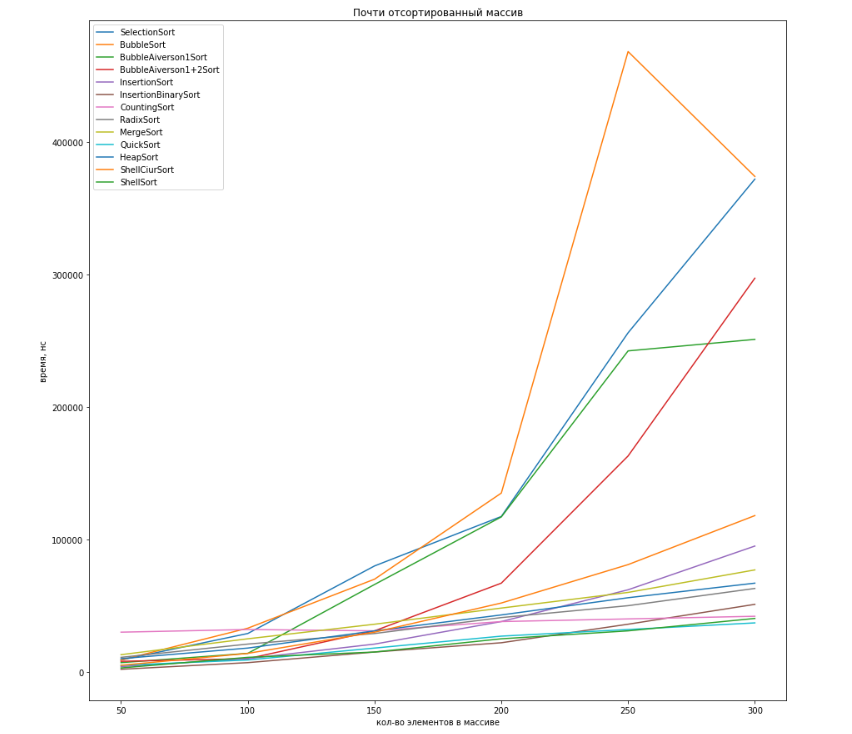
****

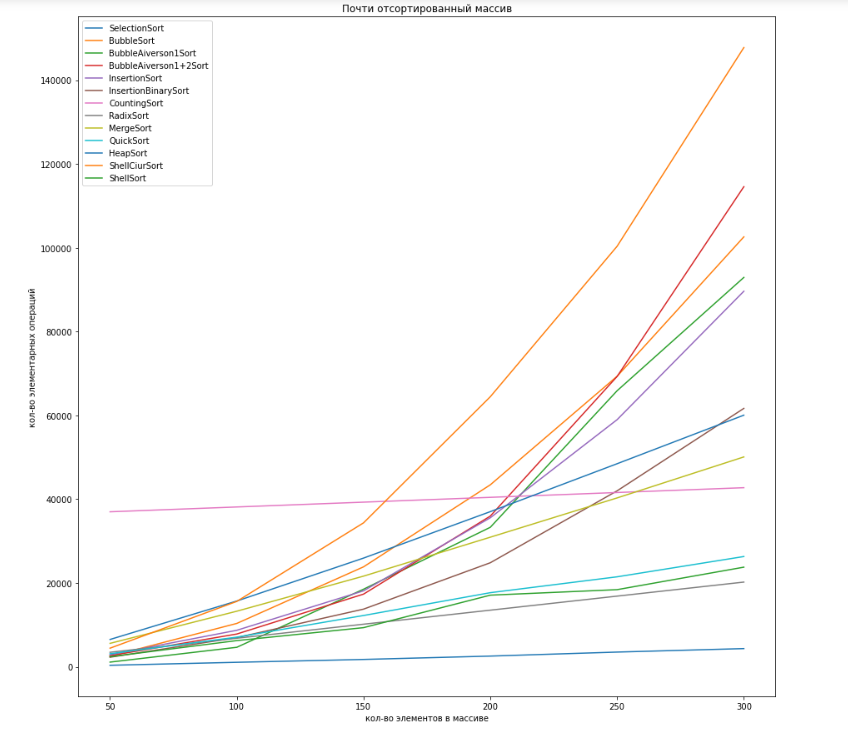
* Вывод: Теоретические данные подтвердились, сортировки, работающие асимптотически медленнее, показали более плохой результат чем те, которые должны работать быстрее

**50-300, тип 2**

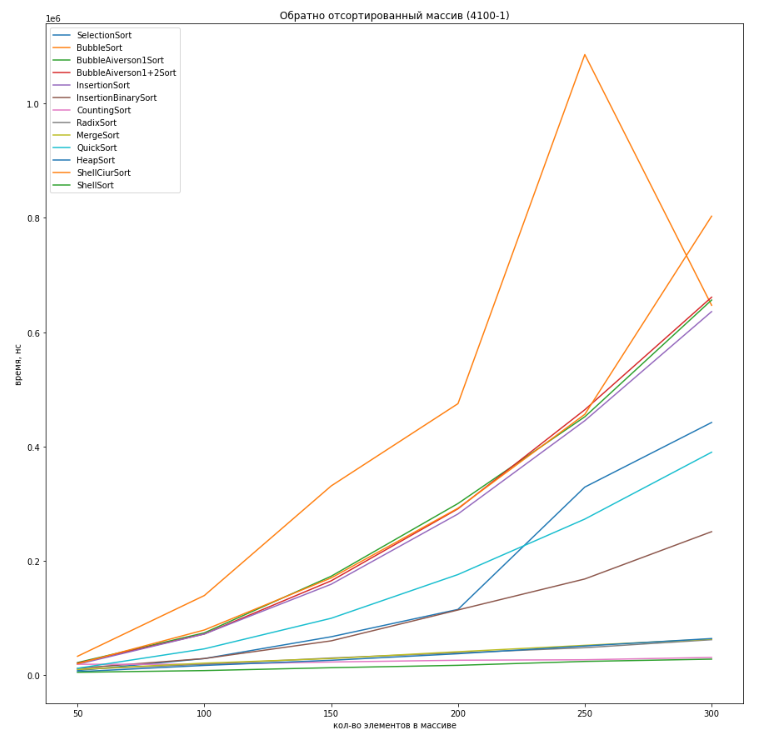
****

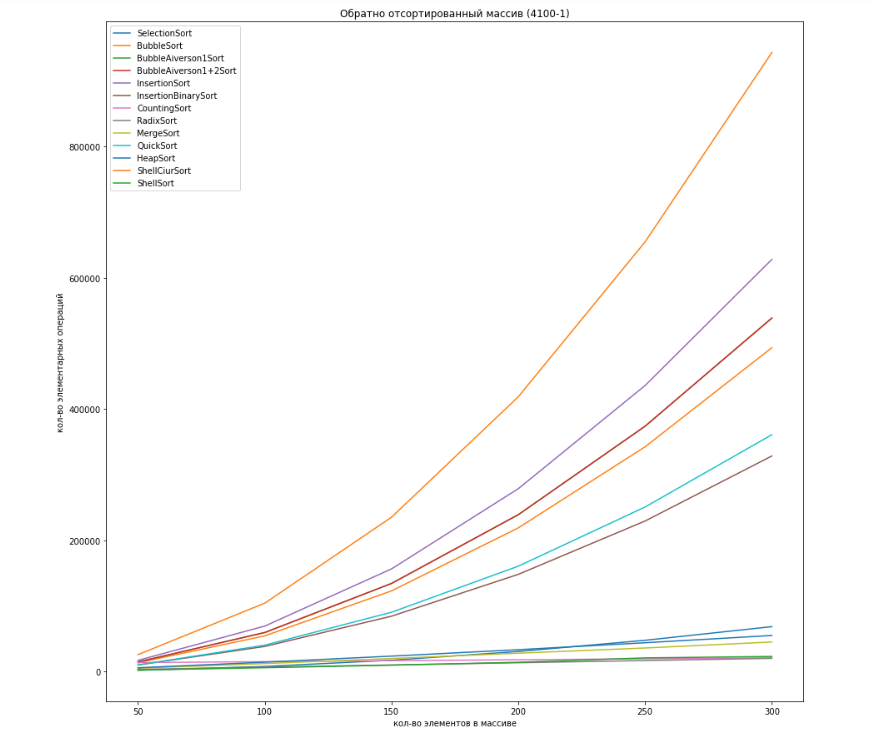
* Вывод: Теоретические данные подтвердились, сортировки, работающие асимптотически медленнее, показали более плохой результат чем те, которые должны работать быстрее. Скачки могут быть обусловлены погрешностью от измерения операций, а также возможной работы посторонних программ компьютера

**50-300, тип 3**

****

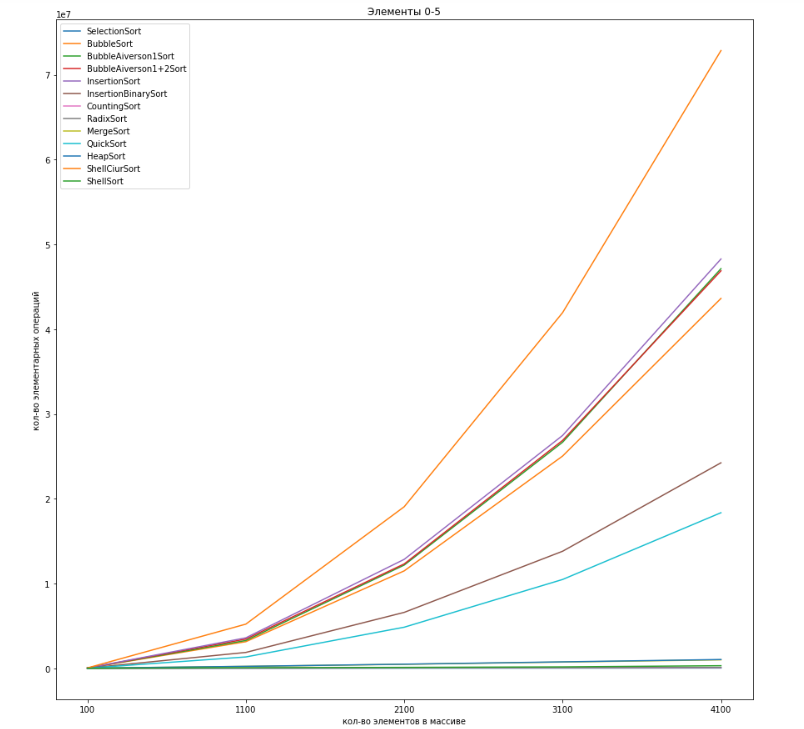
* Вывод: Теоретические данные подтвердились, сортировки, работающие асимптотически медленнее, показали более плохой результат чем те, которые должны работать быстрее. Скачки могут быть обусловлены погрешностью от измерения операций, а также возможной работы посторонних программ компьютера. В среднем сортировки на почти отсортированных данных работают лучше

**50-300, тип 4**

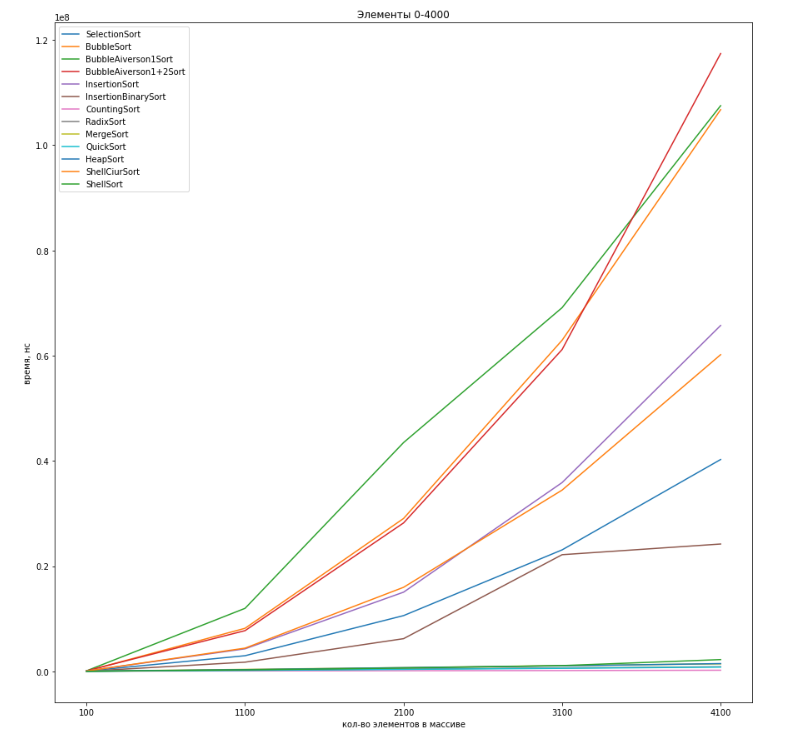
****

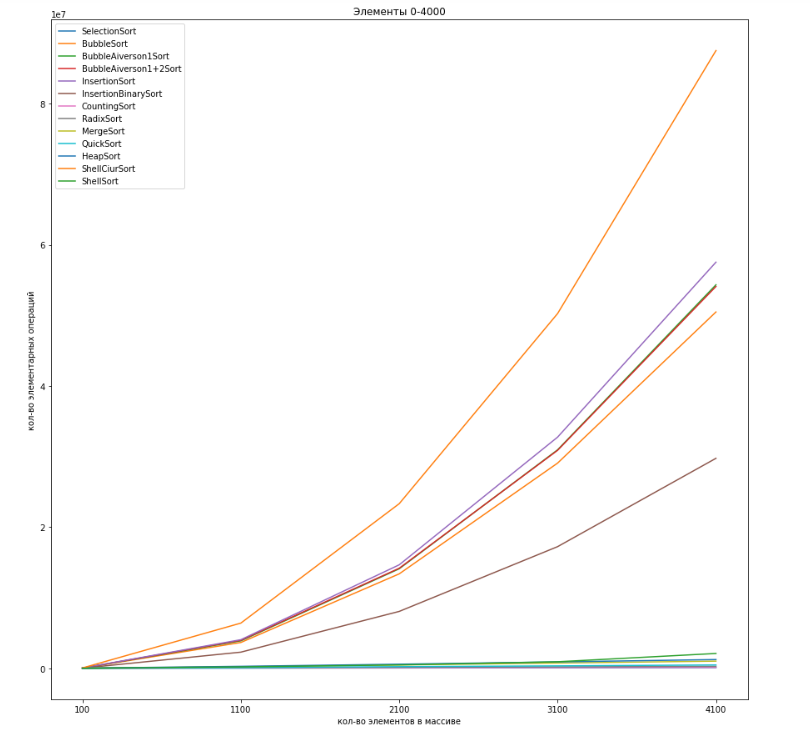
* Вывод: Теоретические данные подтвердились, сортировки, работающие асимптотически медленнее, показали более плохой результат чем те, которые должны работать быстрее. Скачки могут быть обусловлены погрешностью от измерения операций, а также возможной работы посторонних программ компьютера. В среднем сортировки на почти отсортированных данных работают хуже

**100-4100, тип 1**

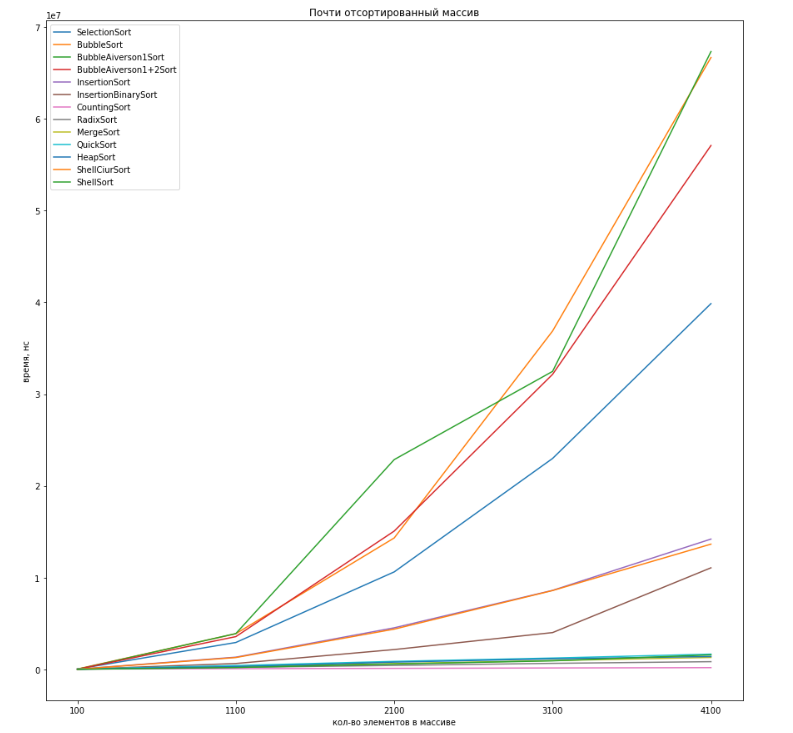
****

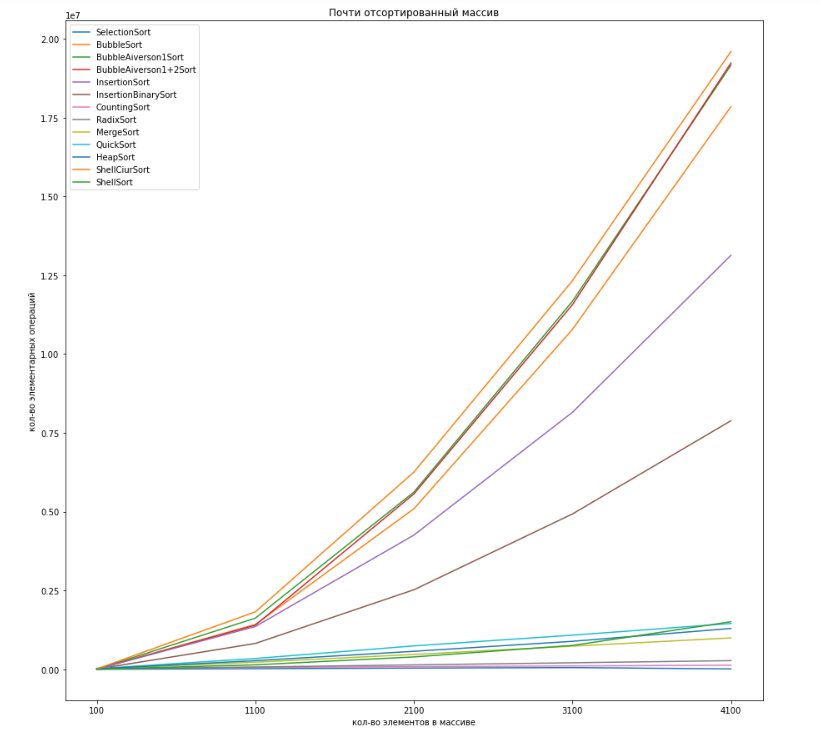
* Вывод: Теоретические данные подтвердились, сортировки, работающие асимптотически медленнее, показали более плохой результат чем те, которые должны работать быстрее

**100-4100, тип 2**

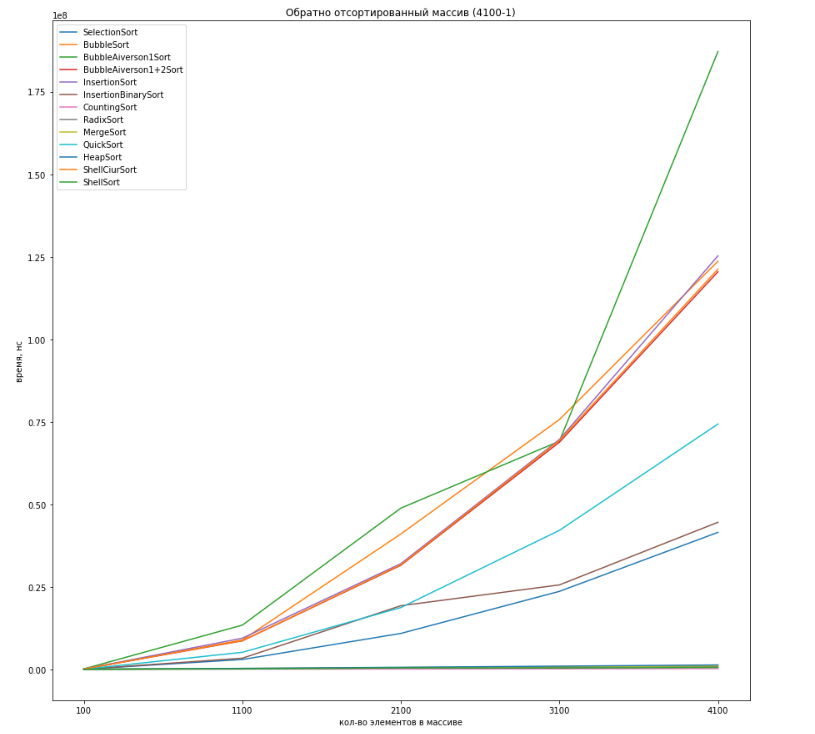
****

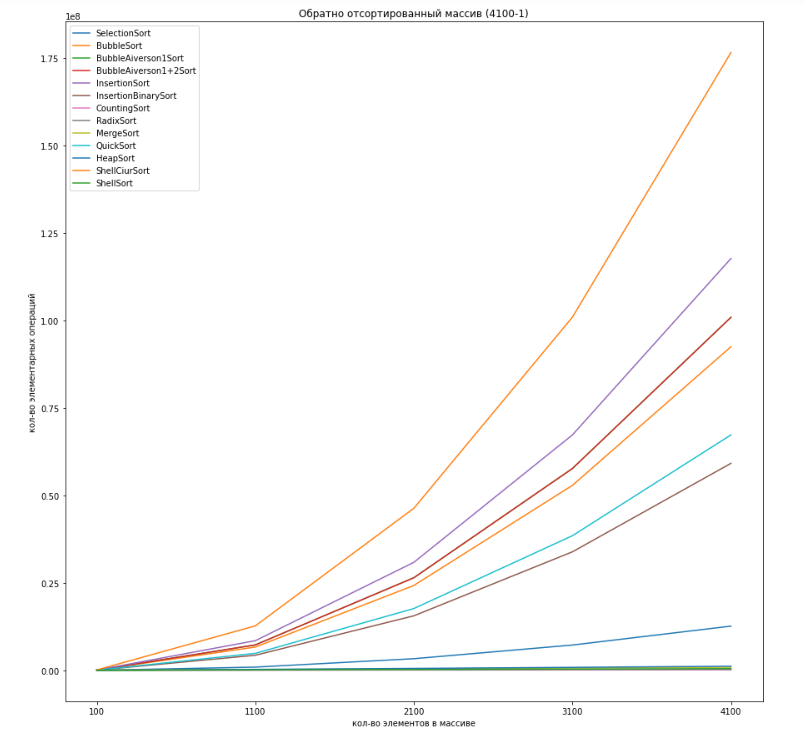
* Вывод: Теоретические данные подтвердились, сортировки, работающие асимптотически медленнее, показали более плохой результат чем те, которые должны работать быстрее. Скачки могут быть обусловлены погрешностью от измерения операций, а также возможной работы посторонних программ компьютера

**100-4100, тип 3**

****

* Вывод: Теоретические данные подтвердились, сортировки, работающие асимптотически медленнее, показали более плохой результат чем те, которые должны работать быстрее. Скачки могут быть обусловлены погрешностью от измерения операций, а также возможной работы посторонних программ компьютера. В среднем сортировки на почти отсортированных данных работают лучше

**100-4100, тип 4**

****

* Вывод: Теоретические данные подтвердились, сортировки, работающие асимптотически медленнее, показали более плохой результат чем те, которые должны работать быстрее. Скачки могут быть обусловлены погрешностью от измерения операций, а также возможной работы посторонних программ компьютера. В среднем сортировки на почти отсортированных данных работают хуже