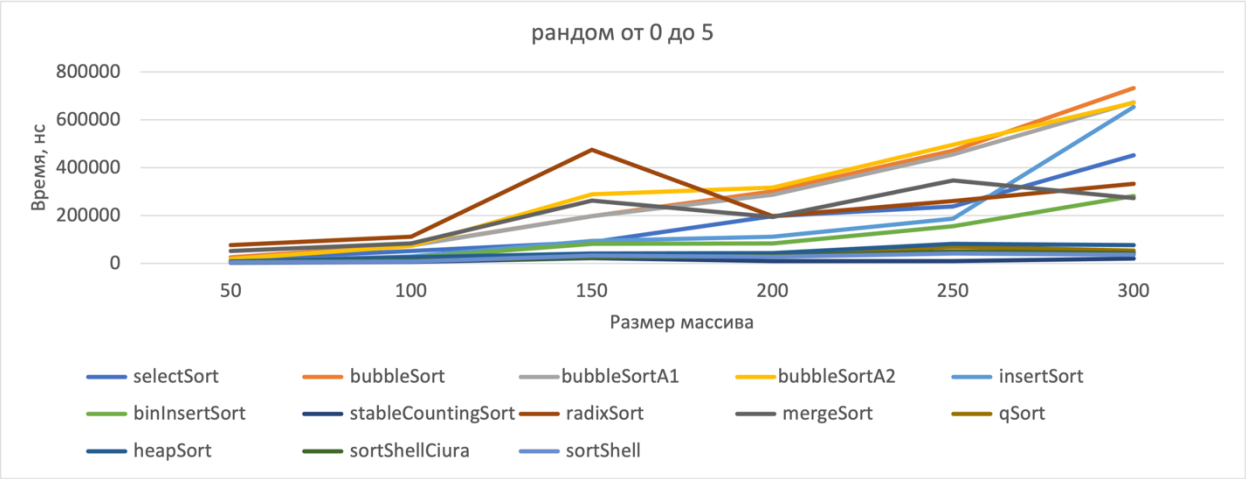
Отчет Галкиной Таисия Олеговны БПИ213

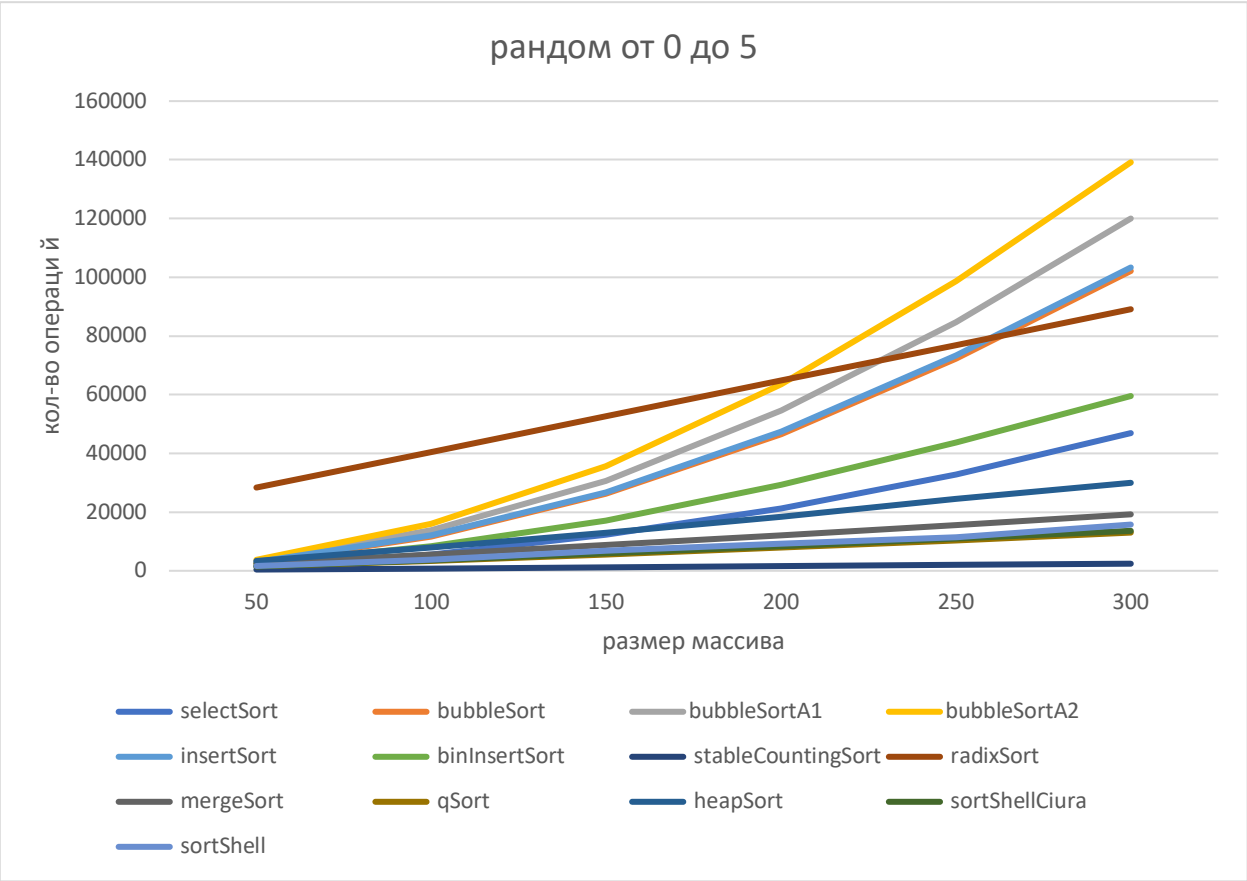
Были реализованы 13 сортировок, мы хотим их осознать и понять
Далее рассмотрит тип графиков, где все сортировки на одном графике:

1) Все сортировки на массивах размера от 50 до 300 с шагом 50 с рандомными значениями от 0 до 5
   Замеры времени с помощью наносекунд:



Заметим, что хуже всего работают, что на маленьких размерах radix , пузырек и пузырек Айверсона1, однако на больших размерах это пузырек. Также заметим, что Шелл работает лучше всего

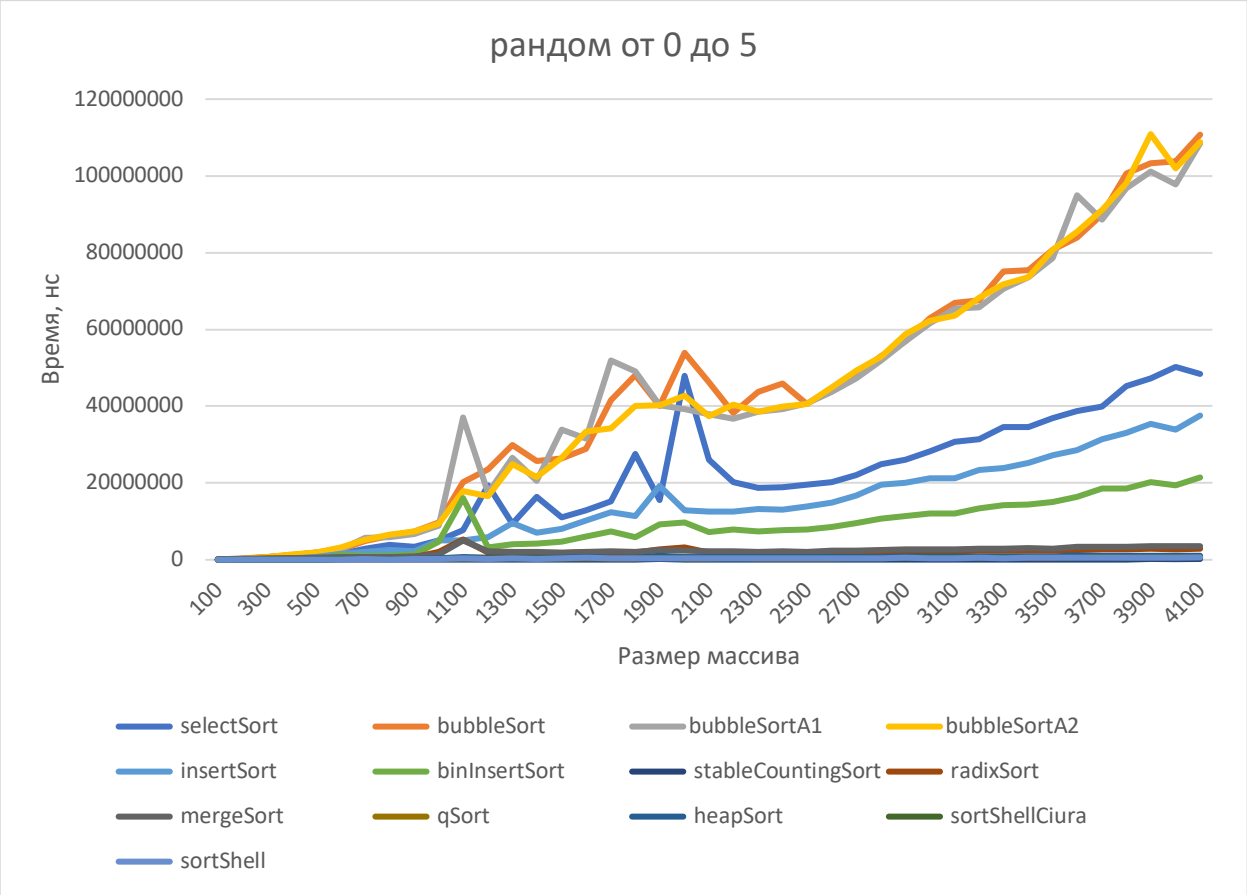
Замеры по времени с помощью элементарных операций:



Заметим, что тут хуже всего работают radix, merge и пузырек на маленьких размерах. А на больших – пузырек и сортировка вставками(примерно с 270 размера). Лучше всего работают сортировки Шелла и подсчетом.

Заметим, что итоговые показатели сходятся на разных способах замера времени.

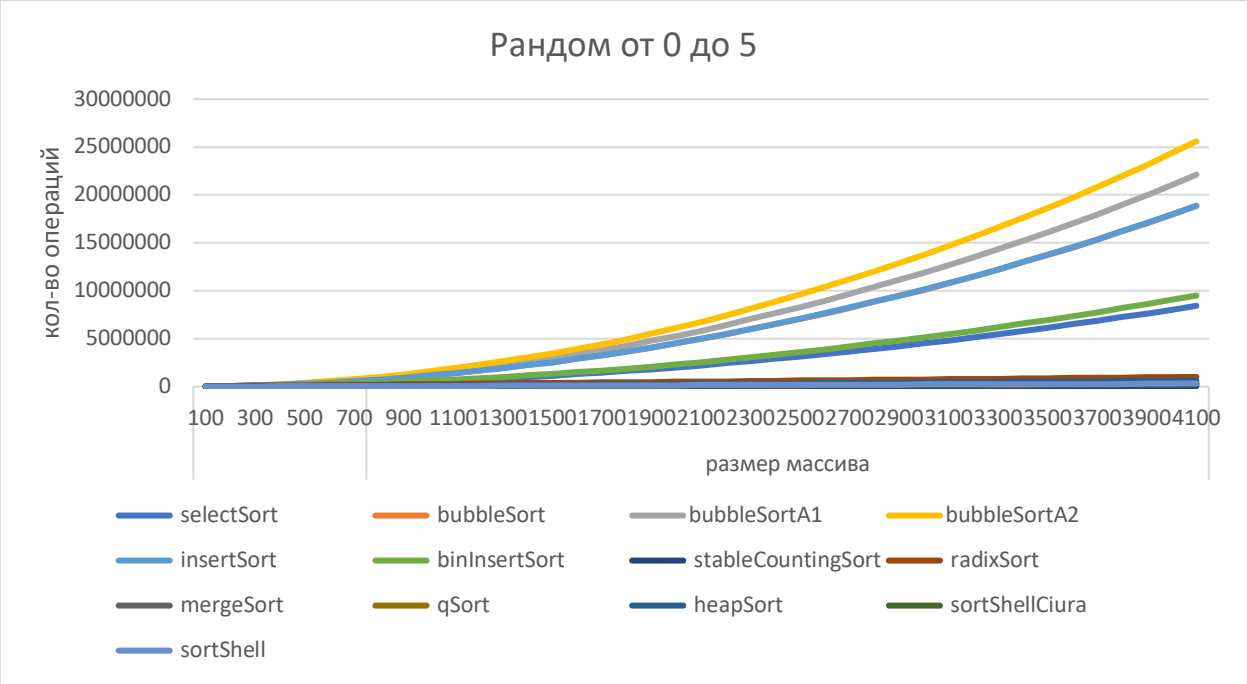
2) Все сортировки на массивах размера от 100 до 4100 с шагом 100 с рандомными значениями от 0 до 5
   Измерение в наносекундах:



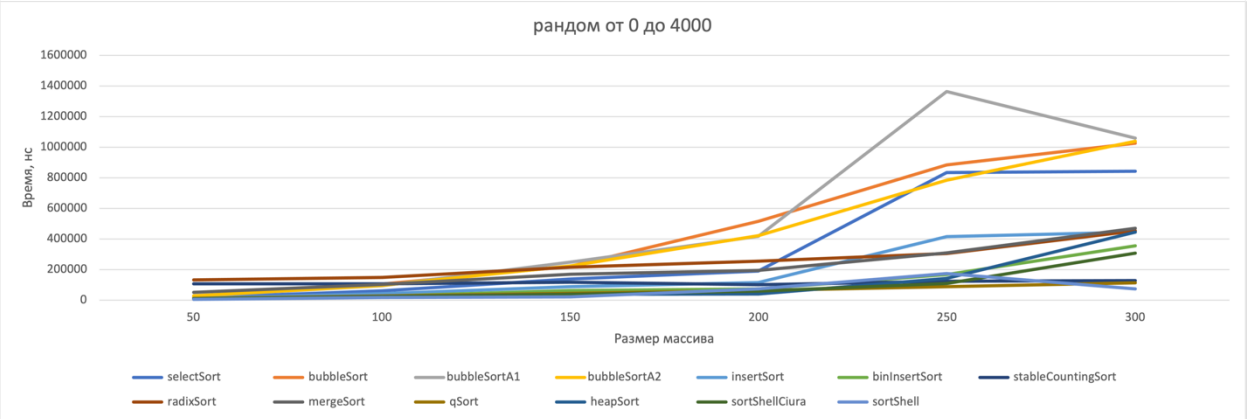
Заметим, что тут хуже всего работает пузырек Айверсона1 и 1+ 2, а также просто пузырек, потом сортировка выбором, затем вставками. Самыми лучшими же оказались сортировки Шелла и сортировка кучей (что в принципе соотносится с теоретическими данными). Также стоит отметить, что тут заметны «выбросы» - они появляются, например, когда работают другие программы. Для уменьшения выбросов специально берется среднее время, и проводится несколько экспериментов. Однако полностью от них избавиться нельзя.

Теперь рассмотрим замеры с помощью элементарных позиций:
Заметим, что хуже всего работают сортировка выбором, пузырьком, потом вставками. Лучше всего сортировки Шелла и пирамидальная.

Заметим, что тут тоже результаты схожи, однако нет выбросов(потому что никак не зависит от того, что там с компьютером происходит), что хорошо.



3) Все сортировки размера от 50 до 300 с шагом 50 с рандомными значениями от 0 до 4000
   Замер в наносекундах

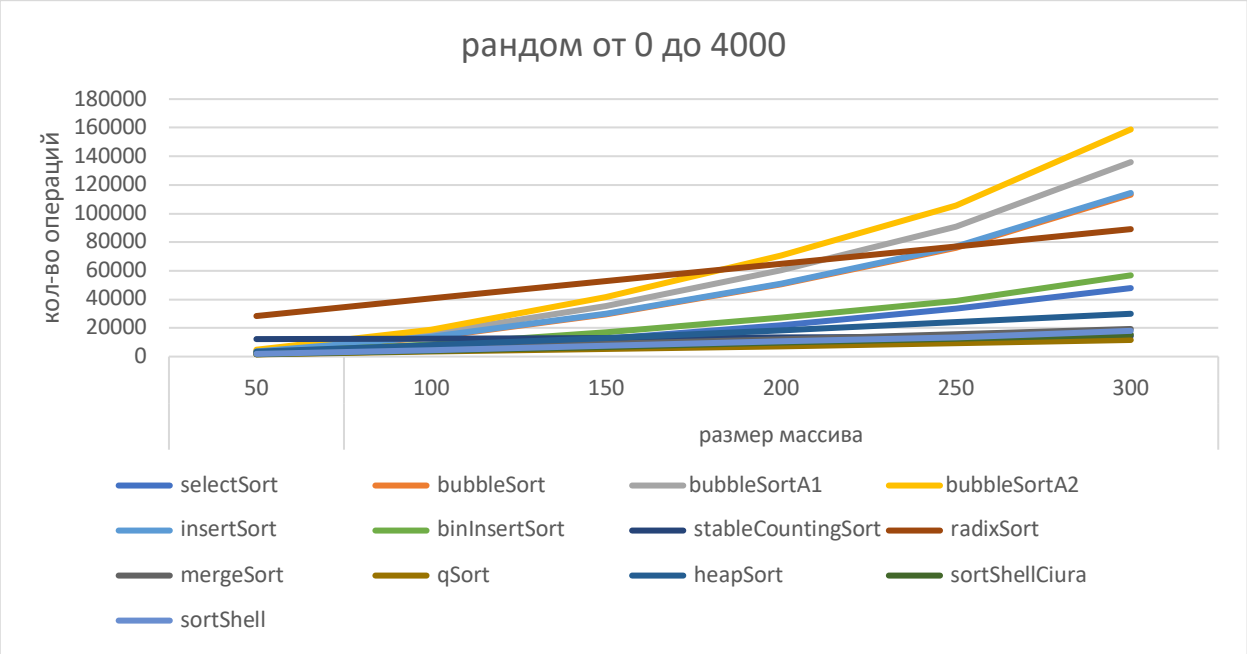


Заметим, что тут тоже на маленьких размерах хуже всего работают пузырек(обычный, 1, 1 + 2), потом сортировка выбором. Лучше всего работают сортировки Шелла и быстрая сортировка

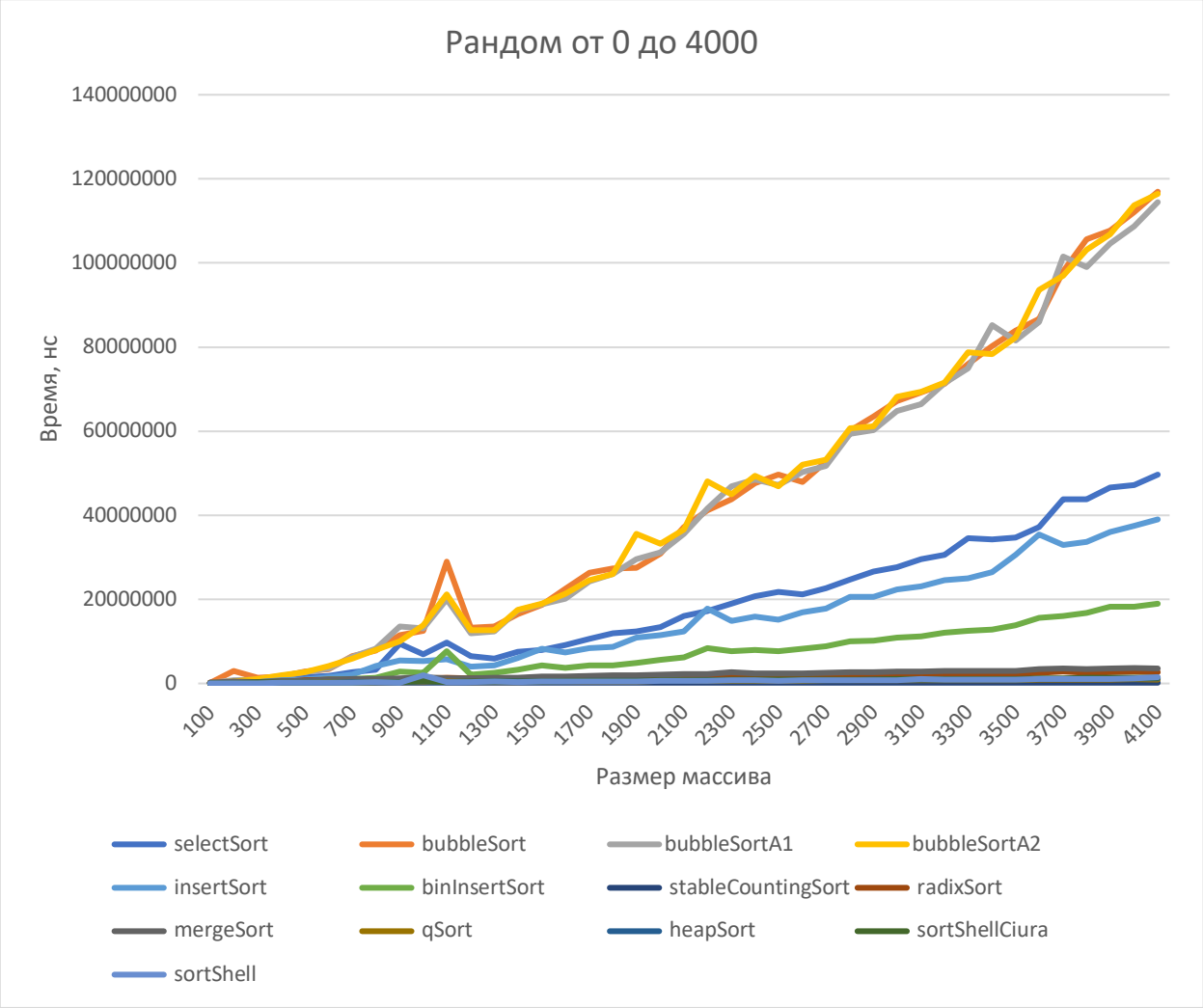
Замер с помощью элементарных операций:

Заметим, что тут хуже всего работают radix(но до размера ~180), сортировка вставками и пузырьками. Что напоминает показатели и на другом способе измерения. А лучше всего работают быстрая сортировка и сортировки Шелла.

Заметим, что итоги тоже совпадают.

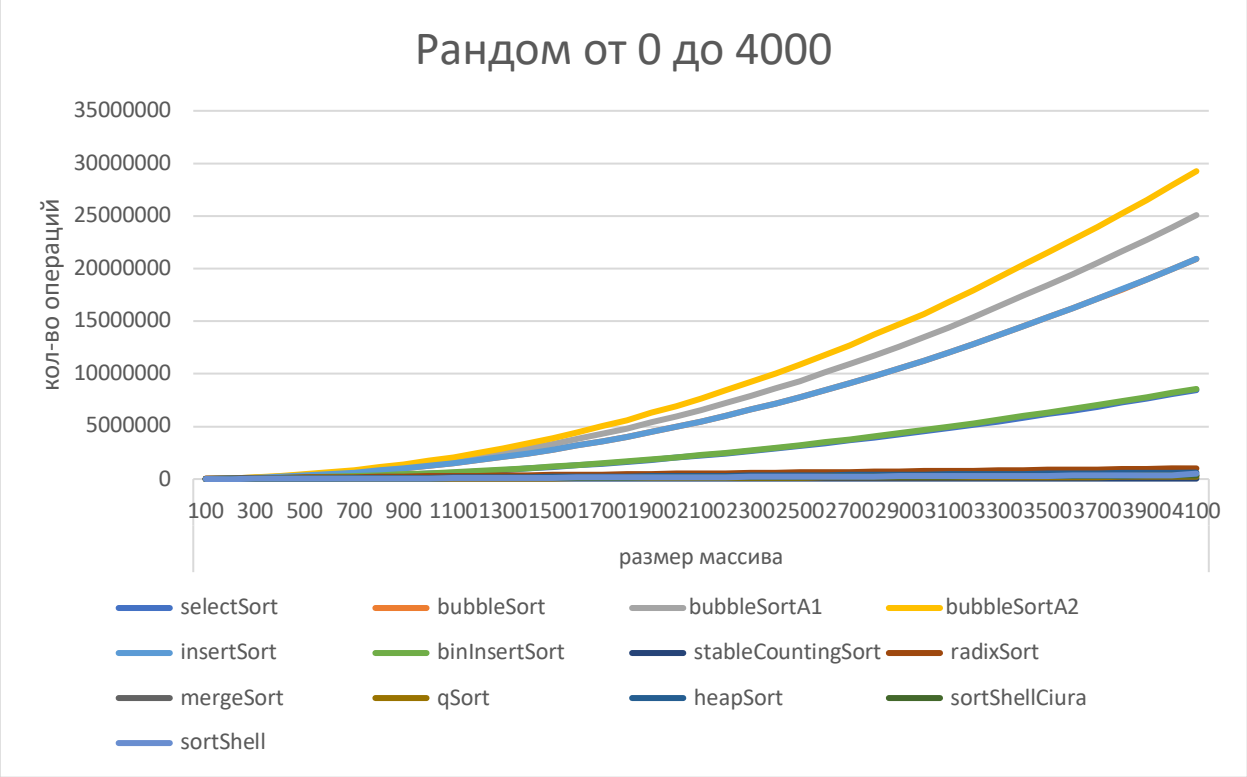


4) Все сортировки размера от 100 до 4100 с шагом 100 с рандомными значениями от 0 до 4000



Заметим, что хуже всего работают все пузырки, сортировка выбором и потом уже сортировка вставками. Заметим, что также эти сортировки еще медленнее работают на больших массивах. Лучшие же показатели у сортировок Шелла и пирамидальной.

Замер с помощью элементарных операций:

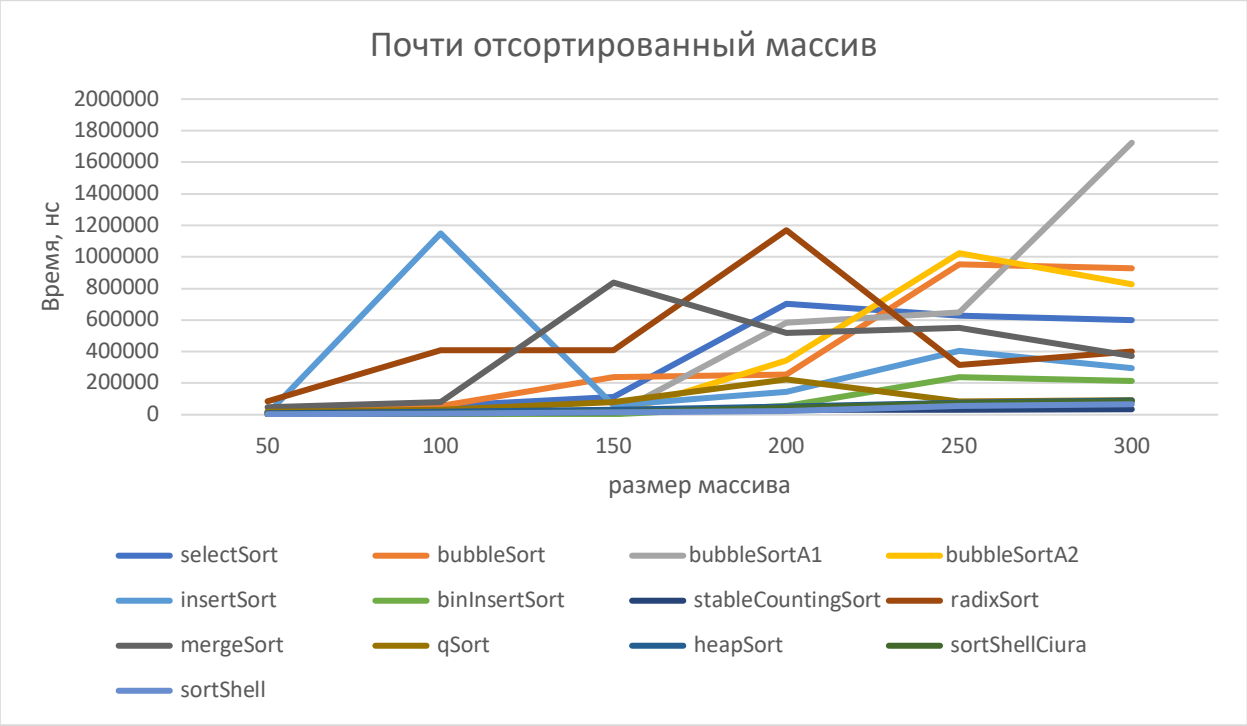


Заметим, что хуже всего работают пузырьки, потом вставками и бинарными вставками. Лучше всего работает сортировка подсчетом и сортировки Шелла.

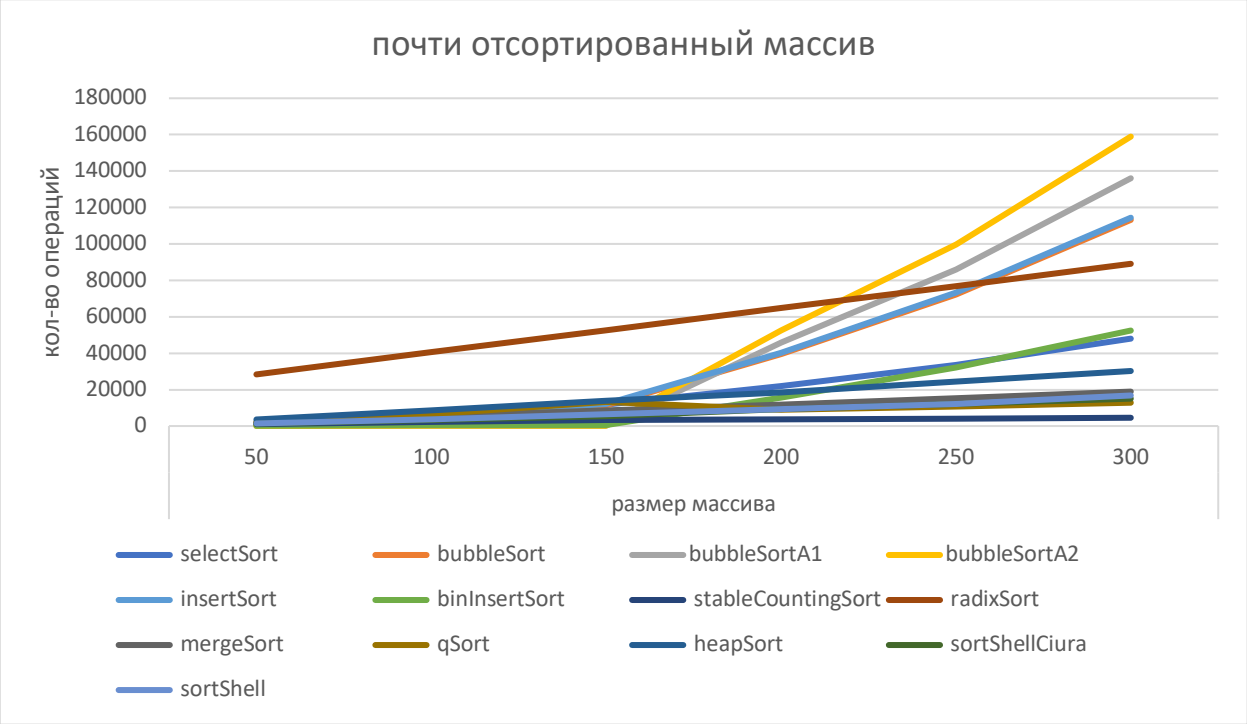
Итог почти одинаковый.

5) Почти отсортированный массив размера от 50 до 300 с шагом 50
   Замеры в наносекундах:

Заметим, что пузырьки стабильно работают плохо. Также медленно работают сортировки radix, merge. Но начиная с ~длины массива 270 медленнее всего работает пузырек. Лучше всего работают пирамидальная сортировка и сортировки Шелла.



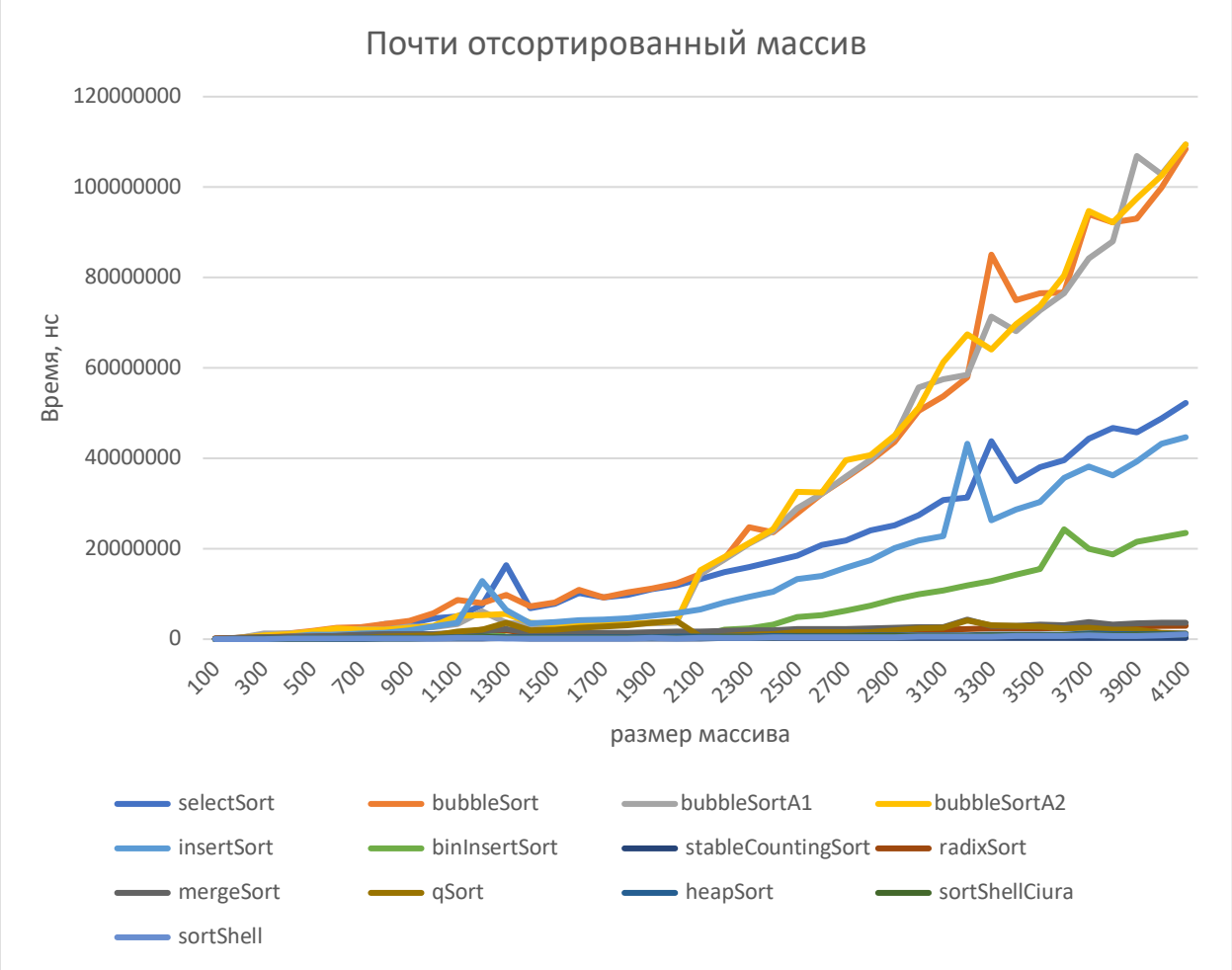
Кол-во элементарных операций:



Заметим, что хуже всего работают сортировки. А где-то с ~270 размера медленнее всего работают пузырек и сортировка вставками. Лучше же работают сортировка подсчетом, сортировки Шелла и сортировка подсчетом. И заметим, что у нас изначальный массив до 150 отсортирован => там все равно что да как. Но потом уже видно что идет как при рандоме.

Заметим, что графики не отличаются глобально.

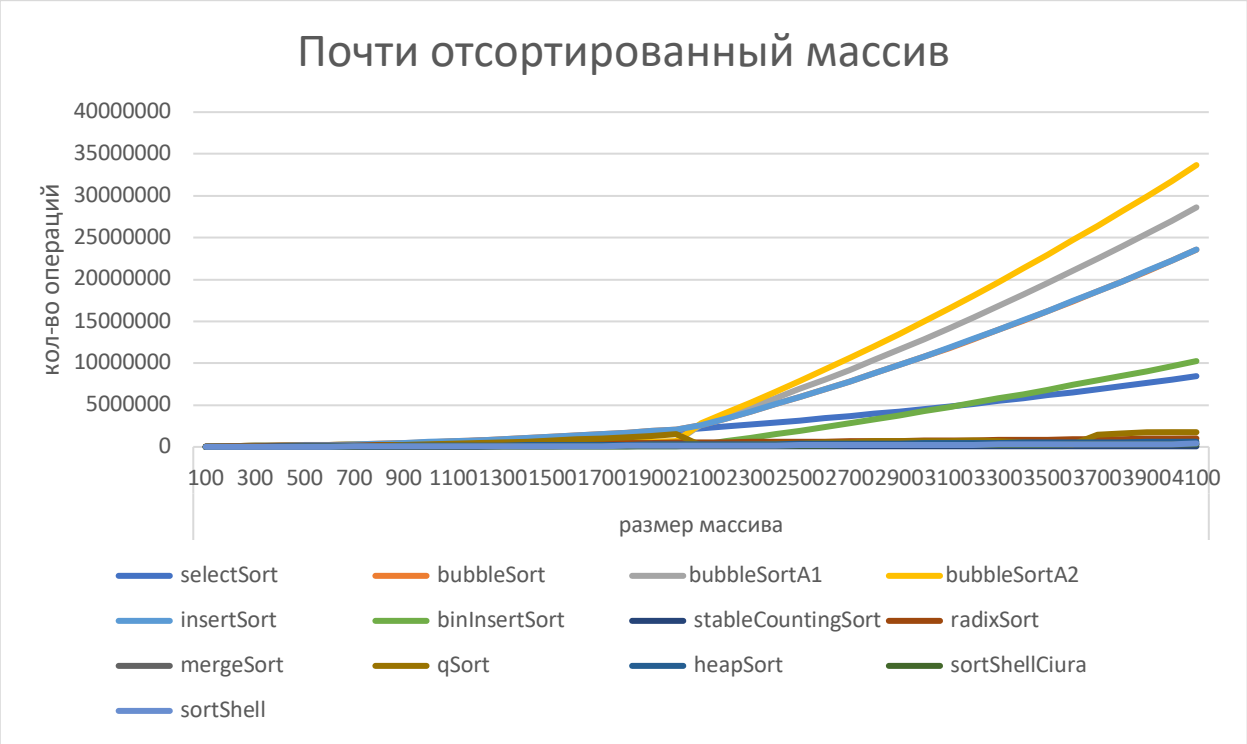
6) Почти отсортированный массив размера от 100 до 4100 с шагом 100
   Замеры в наносекундах:



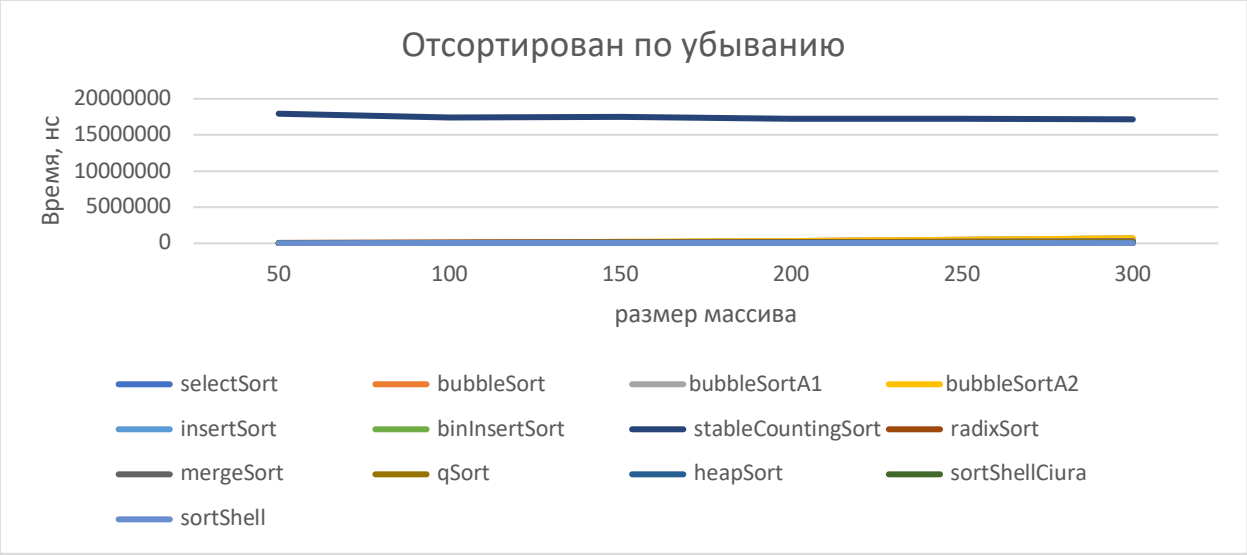
Здесь все аналогично - пузырьки стабильно работают плохо. Также медлено работают сортировки выбором и merge. Лучше всего работают пирмидальная сортировка и сортировки Шелла и быстрая сортировка.

Кол-во элементарных операций:

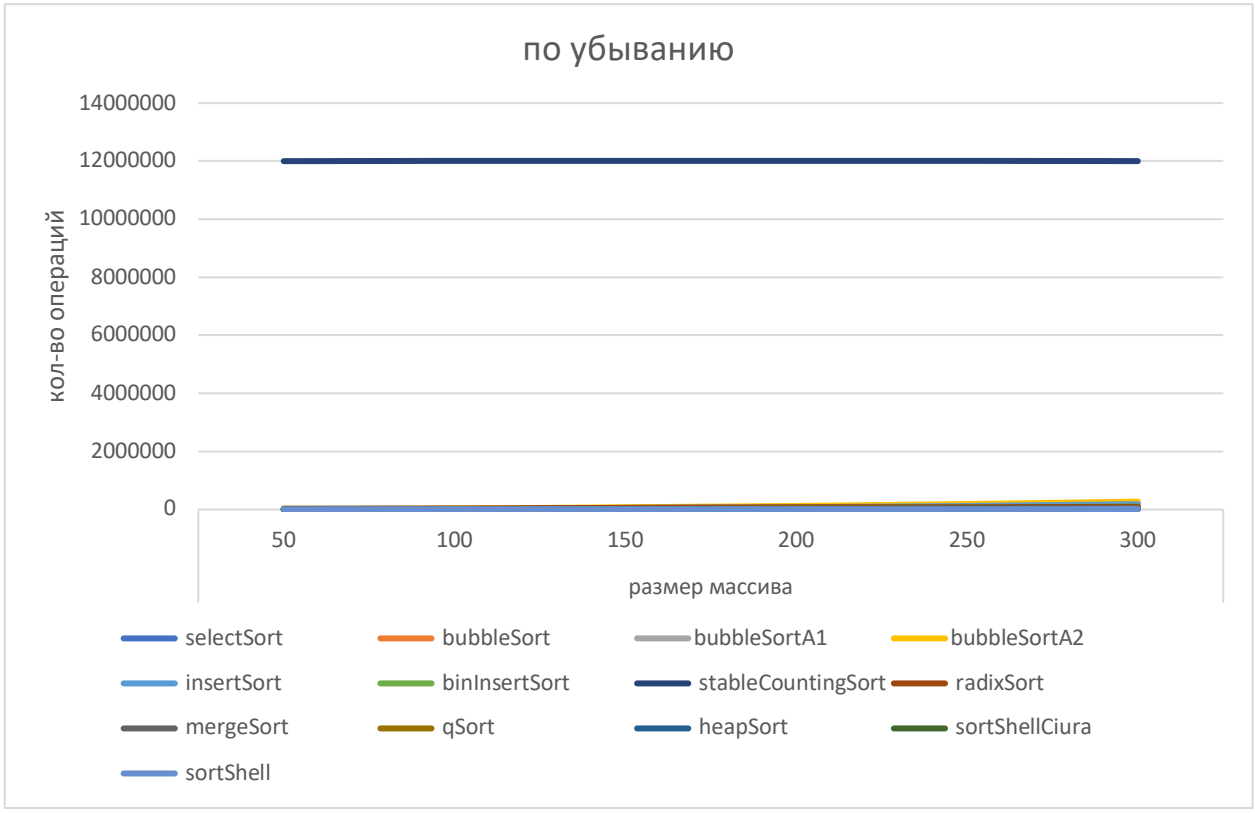
И заметим, что у нас изначальный массив до 2050(на размерах меньше нужно смотреть предыдущий пункт) отсортирован => там все равно что да как. Но потом уже видно что идет как при рандоме. И Результат там точно такой же



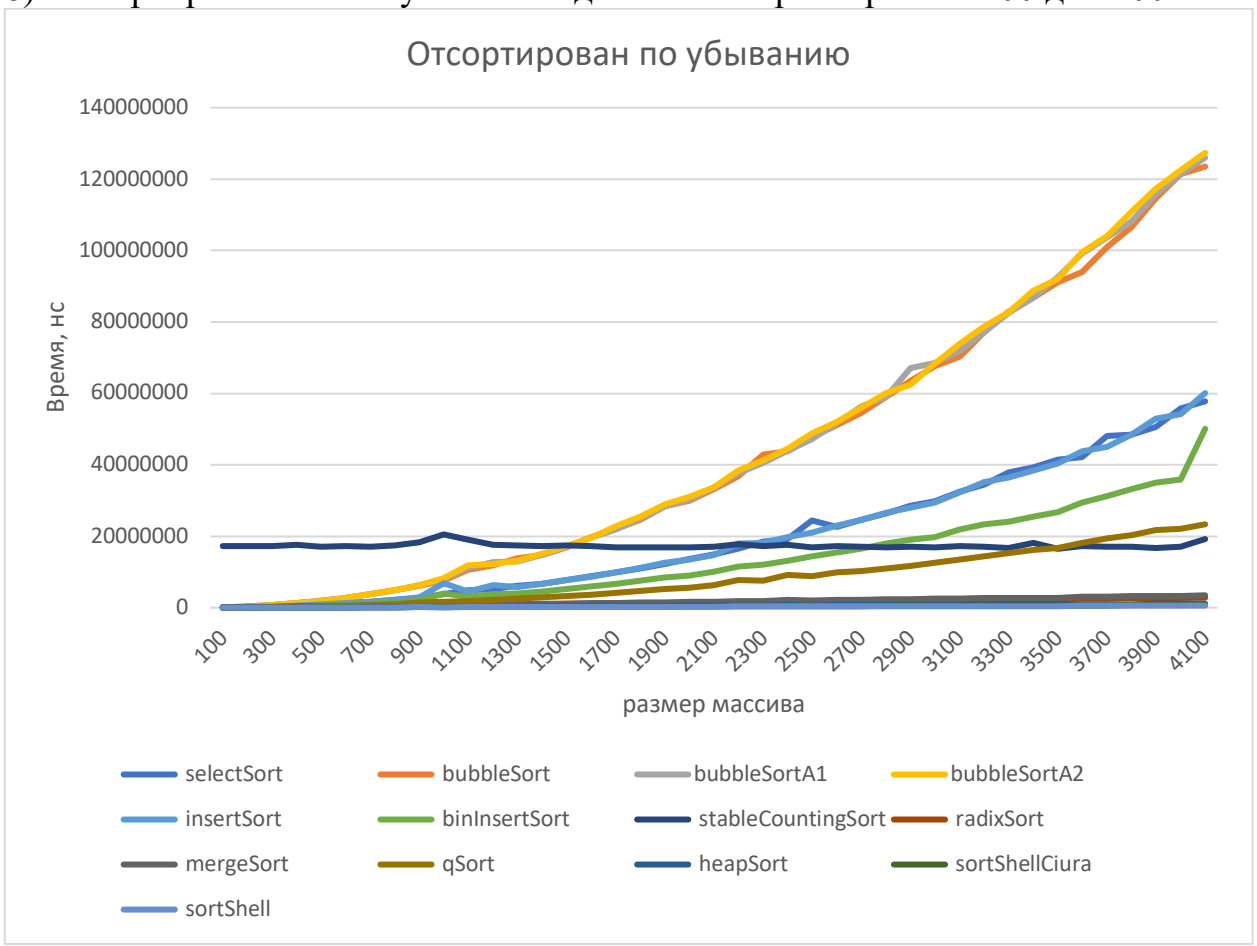
7) Отсортированный по убыванию для массива размером от 50 до 300
Сортировка подсчетом дала тут самый ужасный результат, после нее идут пузырьки. Самая лучшая сортировка – сортировки Шелла.



По кол-ву элементарных операций:
Заметим, что тут аналогичная ситуация, что и в прошлом пункте. То есть сортировка подсчетом ооооочень плохо работает. А потом идут пузырьки, сортировка вставками. А лучше всего – сортировки Шелла.

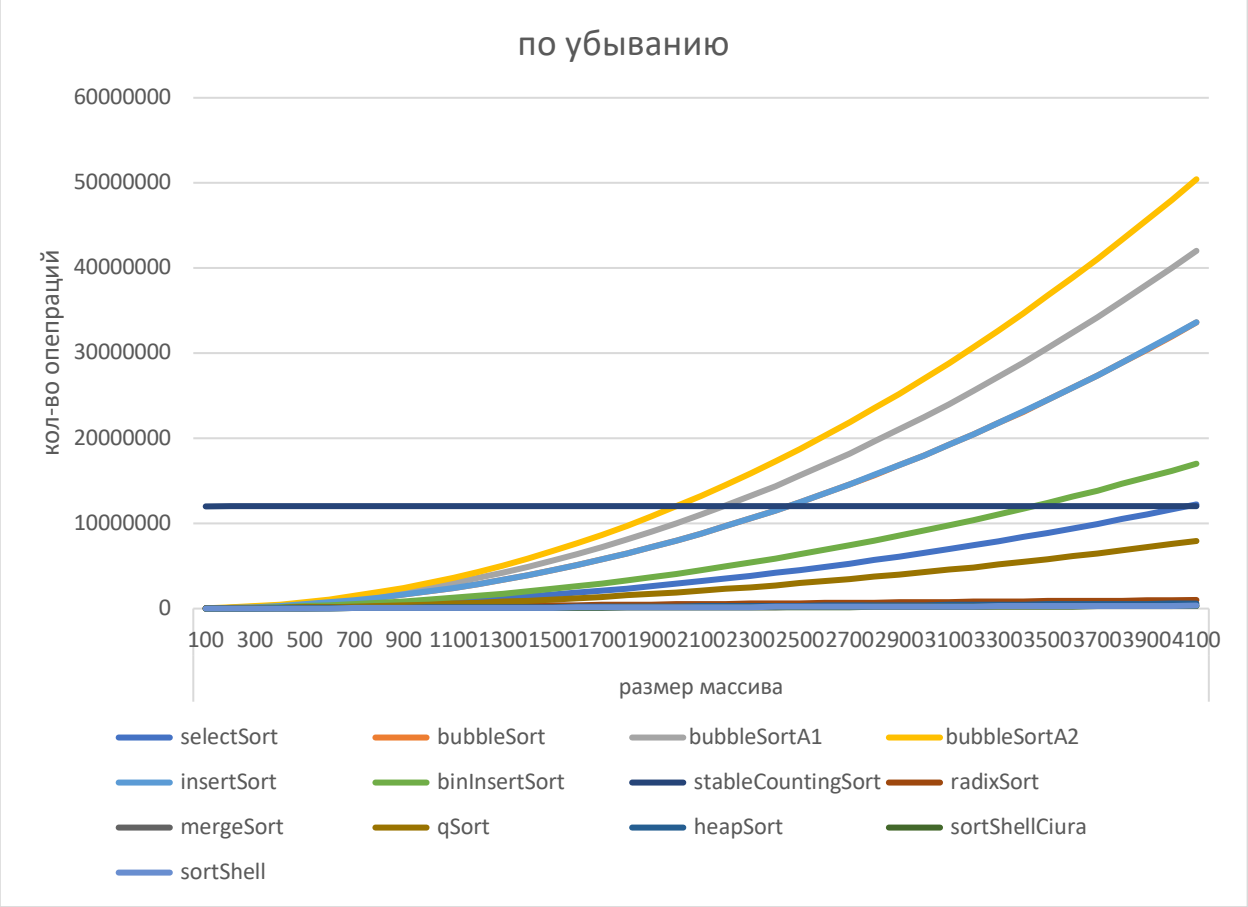


8) Отсортированный по убыванию для массива размером от 100 до 4100



Хуже всего работают пузырьки, потом идут сортировки выбором и вставками. Заметим, что сортировка подсчетом наоборот в данном случае становится быстрее с увеличением размера массива. Самые лучшие – сортировки Шелла.

По кол-ву элементарных оперций:

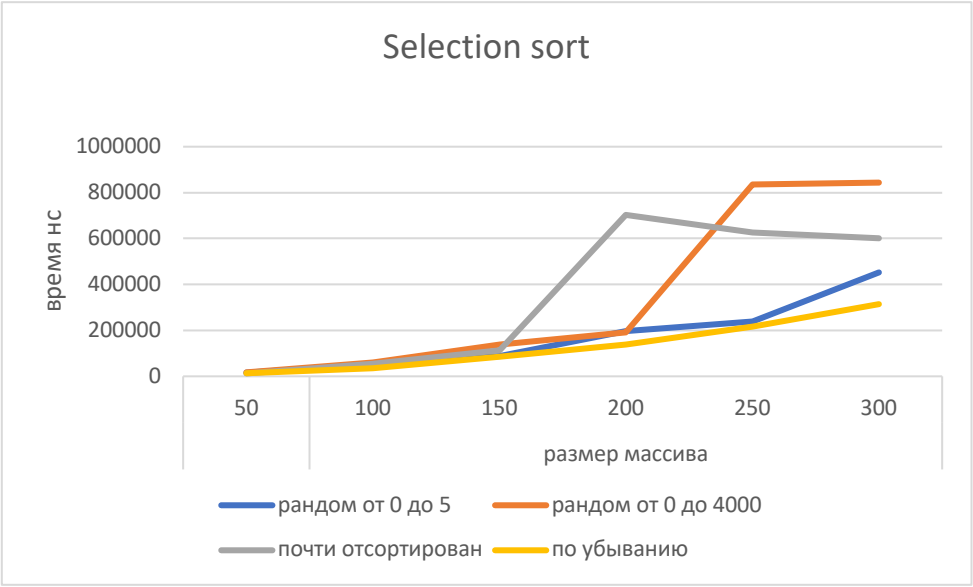


Хуже всего работает сортировка подсчетом. Но это до раз0мера 2200 примерно. Потом идут сортировки пузырьком, вставками. Лучше всего работают сортировки Шелла и пирамидальная.

То есть видим, что все совпадает.

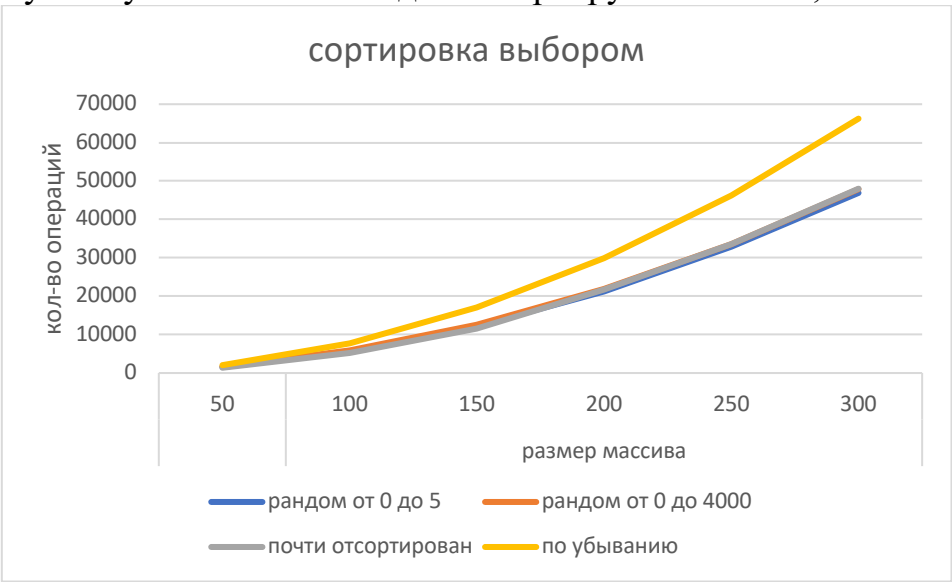
Сортировка выбором:

Видим, что до размера 150 на всех массивах одинаково работает сортировка, однако потом хуже всего рандом от 0 до 4000, а лучше всего – по убыванию.



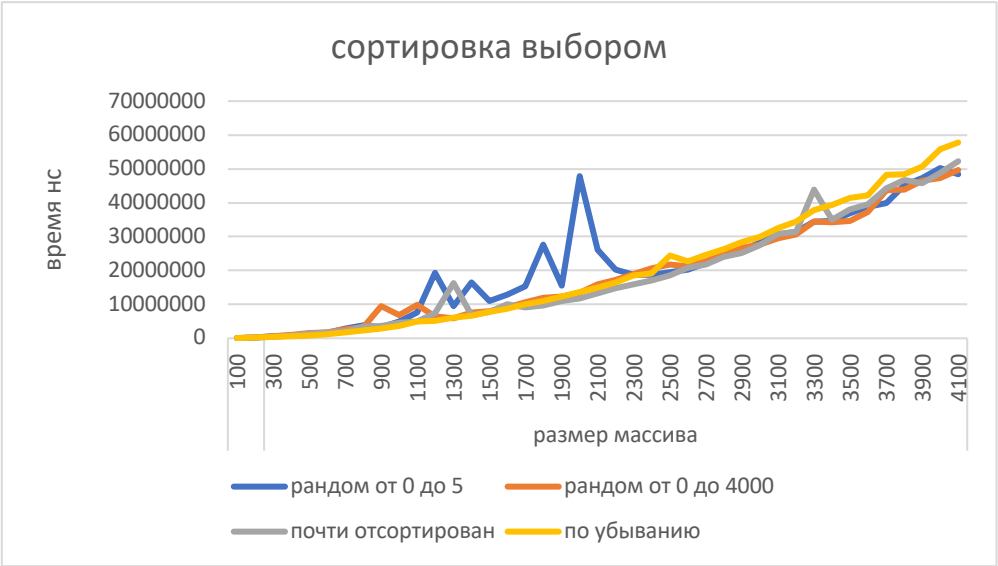
Для маленьких массивов – время

Тут По убыванию очень долго сортируется массив, все остаоьное также



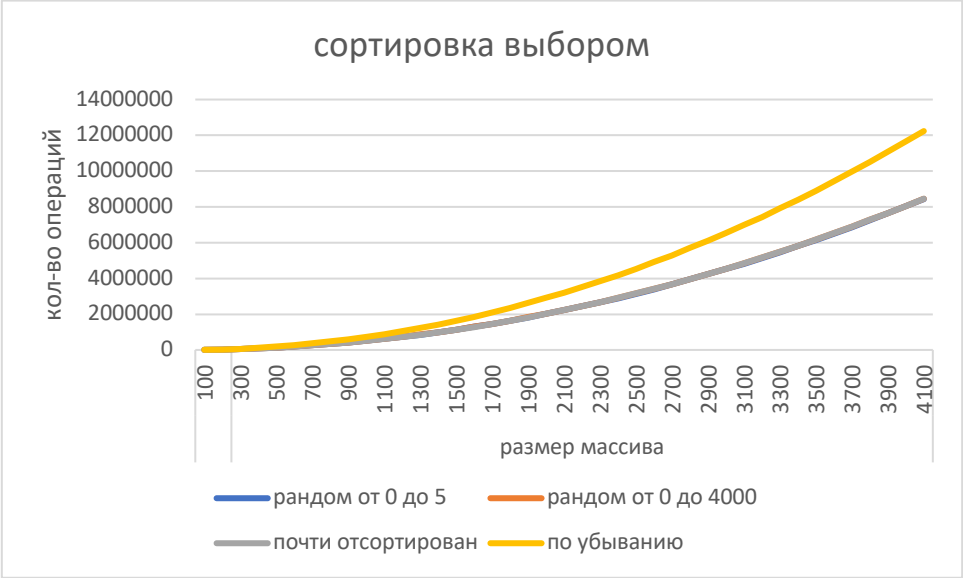
Для маленьких – кол-во операций

Тут явно можно видеть выбросы, но это лучшее из того, что было. Рандом гуже всех тут получается



Для больших – время

Тут просто в обрат ном порядке хуже всех

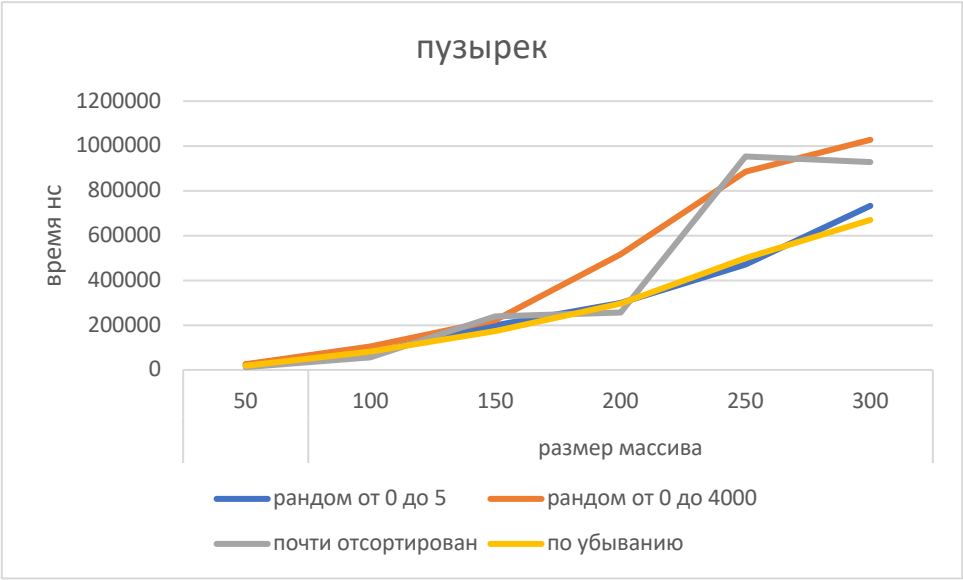


Для больших – кол-во операций

Заметим, что худший результат всегда для массива по убыванию. А остальные варианты примерно равноценны.

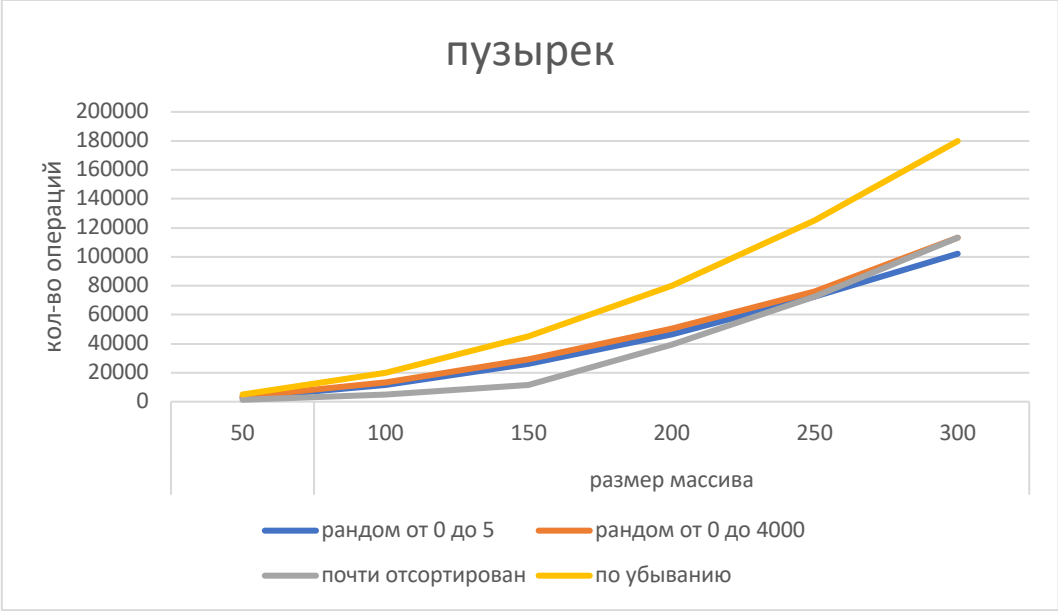
Сортировка пузырьком:

Тут видим, что рандом стабильно плох, однако и почти отсортированный себя проявляет в плохом смысле этого слова. Лучше всех – по убыванию



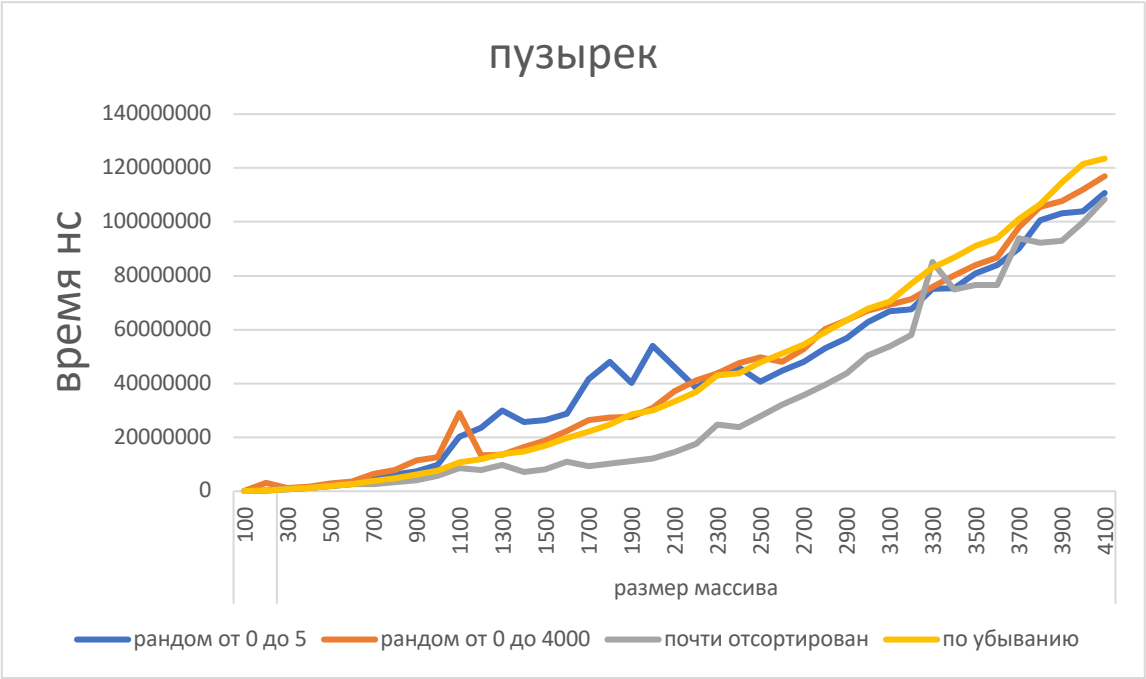
Для маленьких массивов – время

Тут хуже всех по убыванию, лучше – почти отсортированный



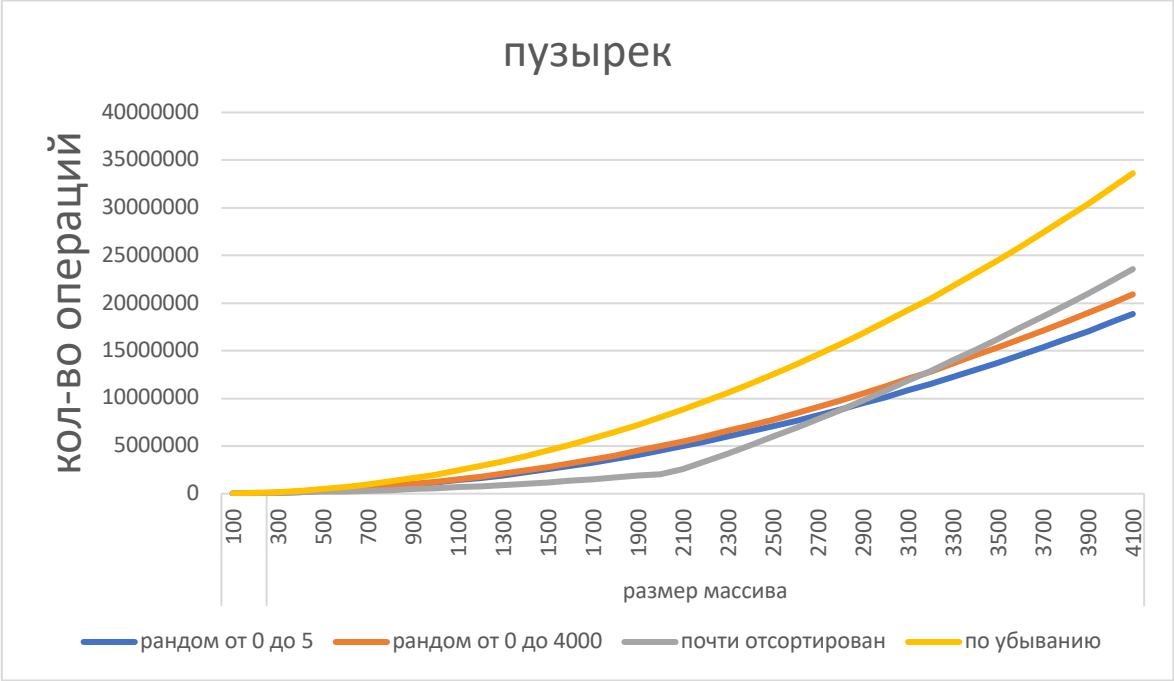
Для маленьких – кол-во операций

Тут хуже всех по рандом от 0 до 5, лучше – почти отсортированный



Для больших – время

Тут хуже всех по убыванию, лучше – почти отсортированный, но с размера примерно 2700 – рандом от 0 до 5



Для больших – кол-во операций

Итог с самопознанием

У меня получилось, что общий вывод при одном графике при подсчете обычным временим и элементарными операциями совпадают. Однако если понимать, что идеальных условий нет, то мне кажется, что лучше считать время выполнения через элементарные операции, поскольку при их подсчете не важно – открыты ли у нас другие приложения, есть ли подключение к интернету и тд. А эти факторы влияют на подсчет времени через реальное время. Но если представить, что мы можем создать условия так, чтобы выбросов не было – то конечно я выбираю реальное время.

Рандомные познания

Заметим, что быстрая сортировка, реализованная рекурсивным методом, работает быстрее, чем сортировка простым выбором при данных размерах.

Заметим, что сортировка Шелла требует значительно меньше времени для сортировки массива (исключая случай отсортированного массива), нежели сортировка пузырьком на экспериментальных данных при данных размерах.

Спасибо за внимание и прочтение!

Хорошего дня!