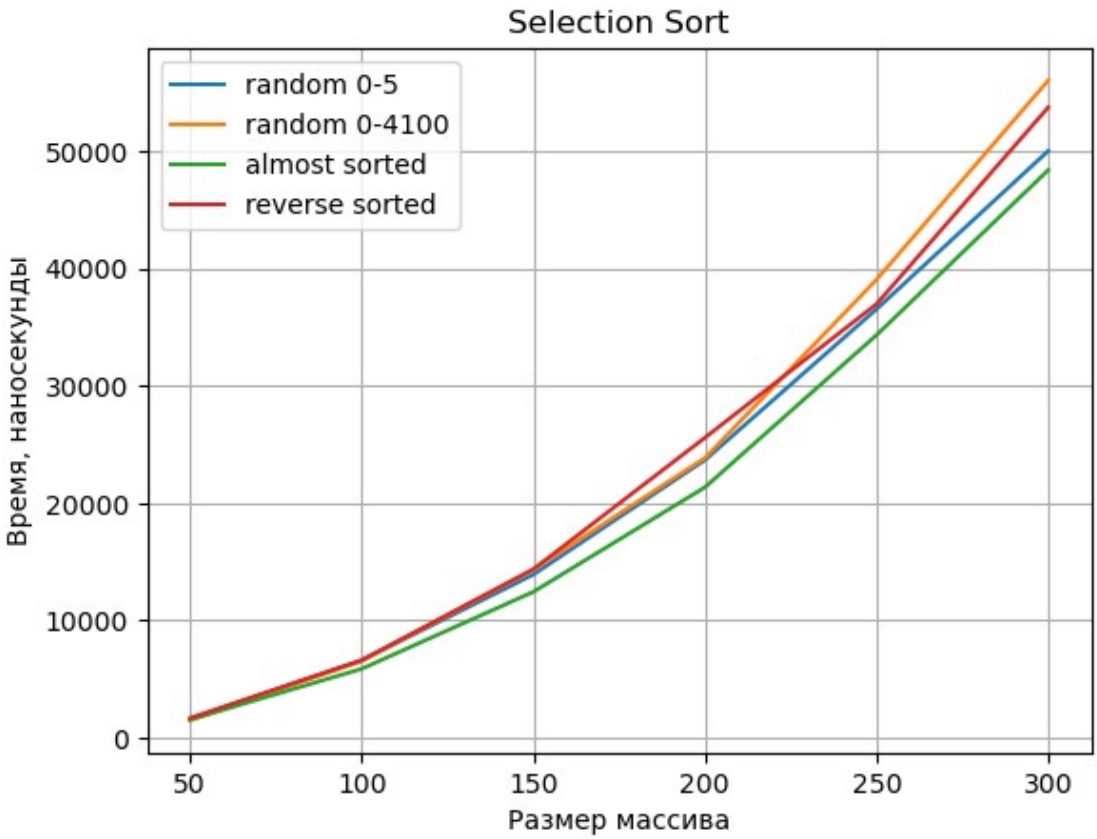
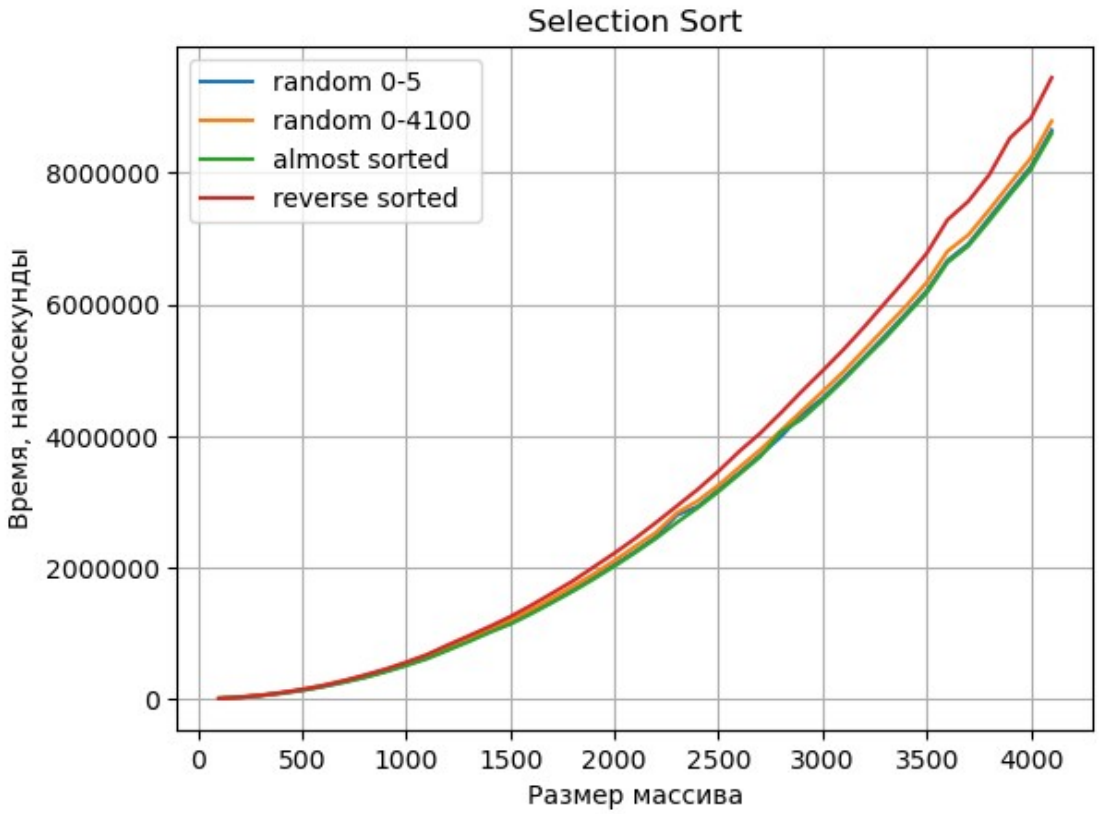
```
import matplotlib.pyplot as plt
import csv
import numpy as np

first_scale = np.arange(50, 301, 50)
second_scale = np.arange(100, 4101, 100)
first_time = open("C:\\Users\\pupki\\Desktop\\first_time.csv", "r")
reader1 = csv.reader(first_time)
selection_sort_r5 = []
selection_sort_r4000 = []
selection_sort_as = []
selection_sort_rs = []
for row in reader1:
    selection_sort_r5.append(int(row[0]))
    selection_sort_r4000.append(int(row[1]))
    selection_sort_as.append(int(row[2]))
    selection_sort_rs.append(int(row[3]))
plt.plot(first_scale, selection_sort_r5, label = 'random 0-5')
plt.plot(first_scale, selection_sort_r4000, label = 'random 0-4100')
plt.plot(first_scale, selection_sort_as, label = 'almost sorted')
plt.plot(first_scale, selection_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Selection Sort')
plt.legend(loc = 'best')
plt.show()
```




Сортировка выбором показывает квадратную сложность по времени, на небольших диапазонах разница между видами массивов не заметна

```
second_time = open("C:\\Users\\pupki\\Desktop\\second_time.csv", "r")
reader2 = csv.reader(second_time)
selection_sort_r5 = []
selection_sort_r4000 = []
selection_sort_as = []
selection_sort_rs = []
for row in reader2:
    selection_sort_r5.append(int(row[0]))
    selection_sort_r4000.append(int(row[1]))
    selection_sort_as.append(int(row[2]))
    selection_sort_rs.append(int(row[3]))
plt.plot(second_scale, selection_sort_r5, label = 'random 0-5')
plt.plot(second_scale, selection_sort_r4000, label = 'random 0-4100')
plt.plot(second_scale, selection_sort_as, label = 'almost sorted')
plt.plot(second_scale, selection_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Selection Sort')
plt.legend(loc = 'best')
```

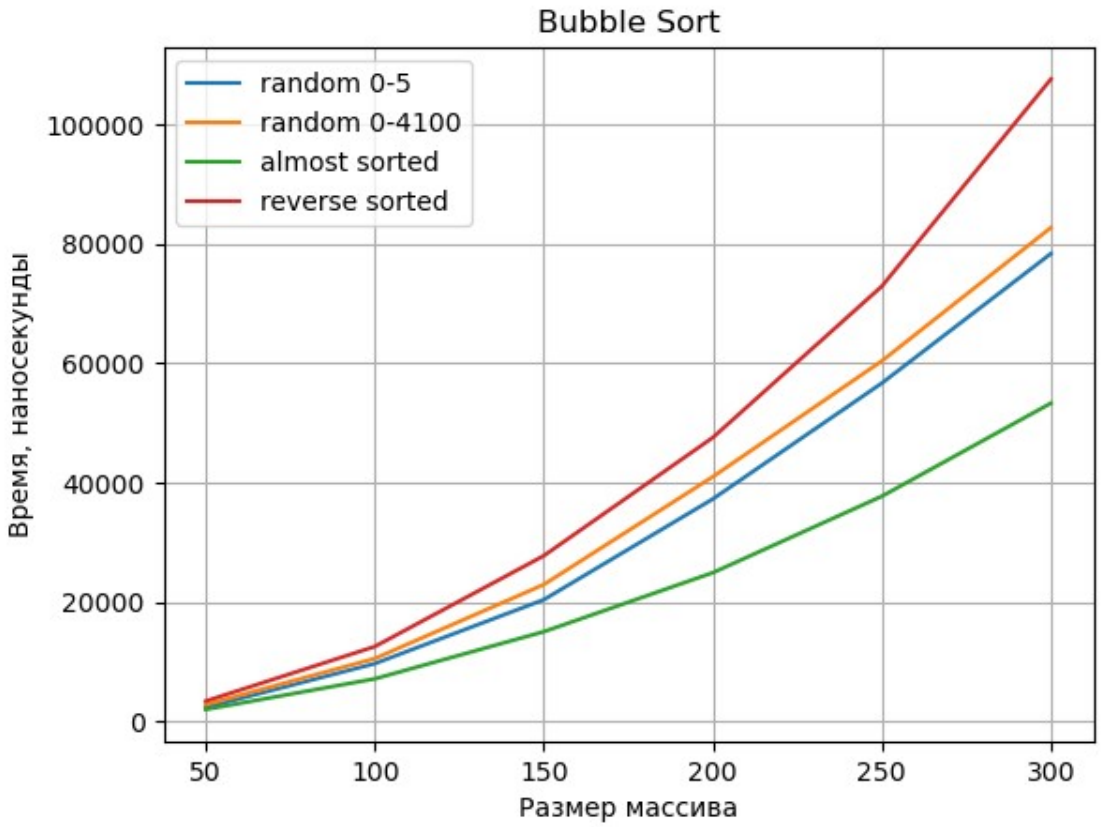
```
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




На бОльших диапазонах подтверждается квадратичная сложность, обратно отсортированный массив занимает больше времени, так как цикл каждый раз проходит по всему массиву

```
first_time = open("C:\\Users\\pupki\\Desktop\\first_time.csv", "r")
reader1 = csv.reader(first_time)
bubble_sort_r5 = []
bubble_sort_r4000 = []
bubble_sort_as = []
bubble_sort_rs = []
for row in reader1:
    bubble_sort_r5.append(int(row[4]))
    bubble_sort_r4000.append(int(row[5]))
    bubble_sort_as.append(int(row[6]))
    bubble_sort_rs.append(int(row[7]))
plt.plot(first_scale, bubble_sort_r5, label = 'random 0-5')
plt.plot(first_scale, bubble_sort_r4000, label = 'random 0-4100')
plt.plot(first_scale, bubble_sort_as, label = 'almost sorted')
plt.plot(first_scale, bubble_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
```

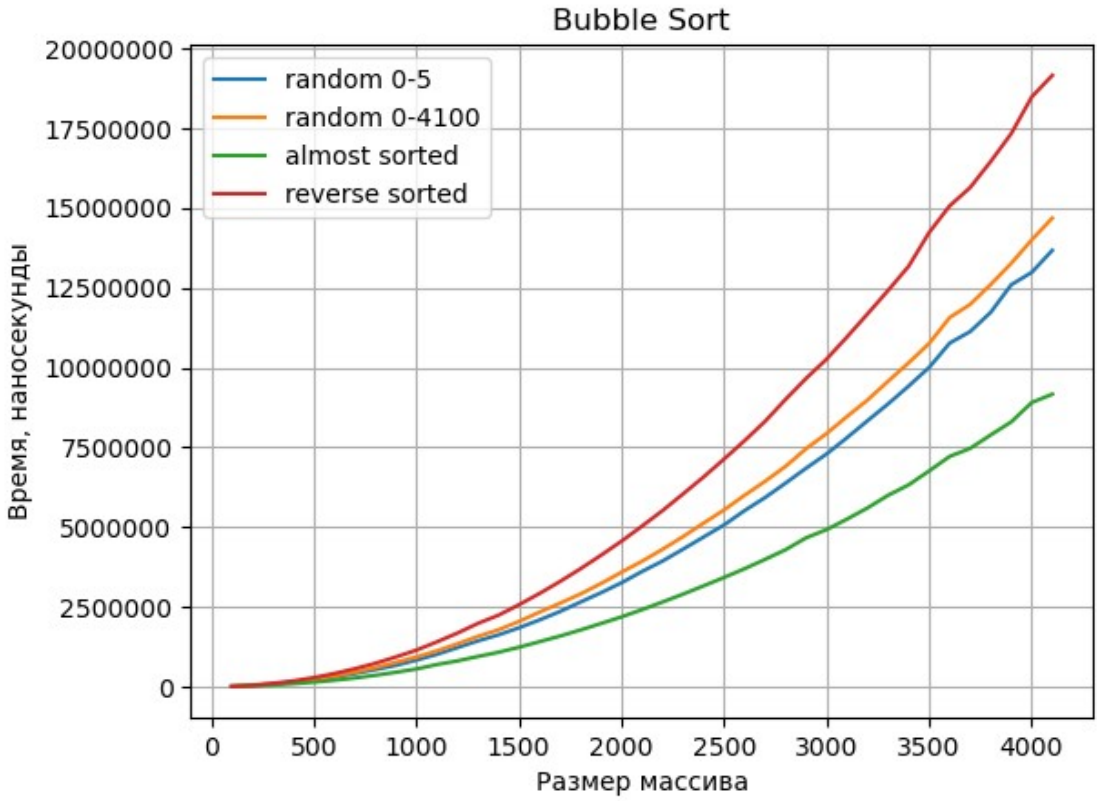
```
plt.title('Bubble Sort')
plt.legend(loc = 'best')
plt.show()
```




Пузырёк - квадратичная сложность по времени, большой цикл в обратно отсортированном массиве, рандомные примерно одинаковы, почти отсортированный работает быстрее.

```
second_time = open("C:\\Users\\pupki\\Desktop\\second_time.csv", "r")
reader2 = csv.reader(second_time)
bubble_sort_r5 = []
bubble_sort_r4000 = []
bubble_sort_as = []
bubble_sort_rs = []
for row in reader2:
    bubble_sort_r5.append(int(row[4]))
    bubble_sort_r4000.append(int(row[5]))
    bubble_sort_as.append(int(row[6]))
    bubble_sort_rs.append(int(row[7]))
plt.plot(second_scale, bubble_sort_r5, label = 'random 0-5')
plt.plot(second_scale, bubble_sort_r4000, label = 'random 0-4100')
plt.plot(second_scale, bubble_sort_as, label = 'almost sorted')
plt.plot(second_scale, bubble_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
```

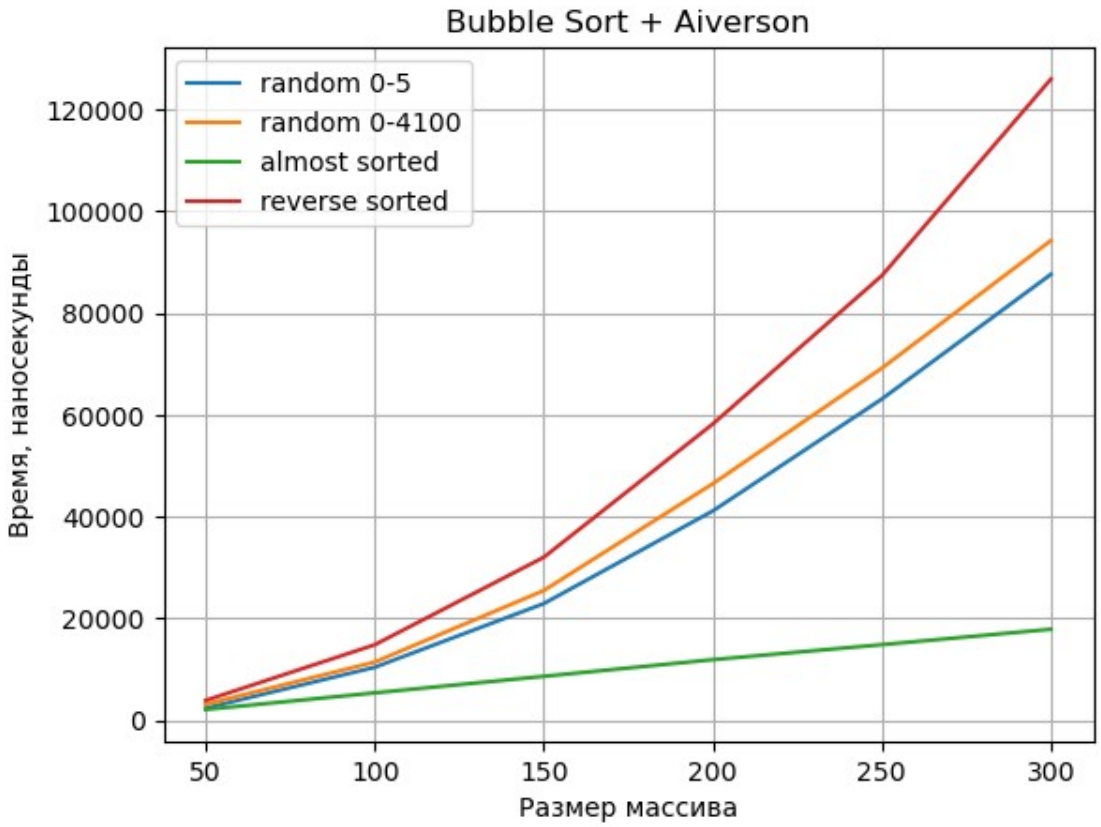
```
plt.ylabel('Время, наносекунды')
plt.title('Bubble Sort')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




На бОльших диапазонах все наблюдения подтверждаются

```
first_time = open("C:\\Users\\pupki\\Desktop\\first_time.csv", "r")
reader1 = csv.reader(first_time)
bubble_1_sort_r5 = []
bubble_1_sort_r4000 = []
bubble_1_sort_as = []
bubble_1_sort_rs = []
for row in reader1:
    bubble_1_sort_r5.append(int(row[8]))
    bubble_1_sort_r4000.append(int(row[9]))
    bubble_1_sort_as.append(int(row[10]))
    bubble_1_sort_rs.append(int(row[11]))
plt.plot(first_scale, bubble_1_sort_r5, label = 'random 0-5')
plt.plot(first_scale, bubble_1_sort_r4000, label = 'random 0-4100')
plt.plot(first_scale, bubble_1_sort_as, label = 'almost sorted')
plt.plot(first_scale, bubble_1_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
```

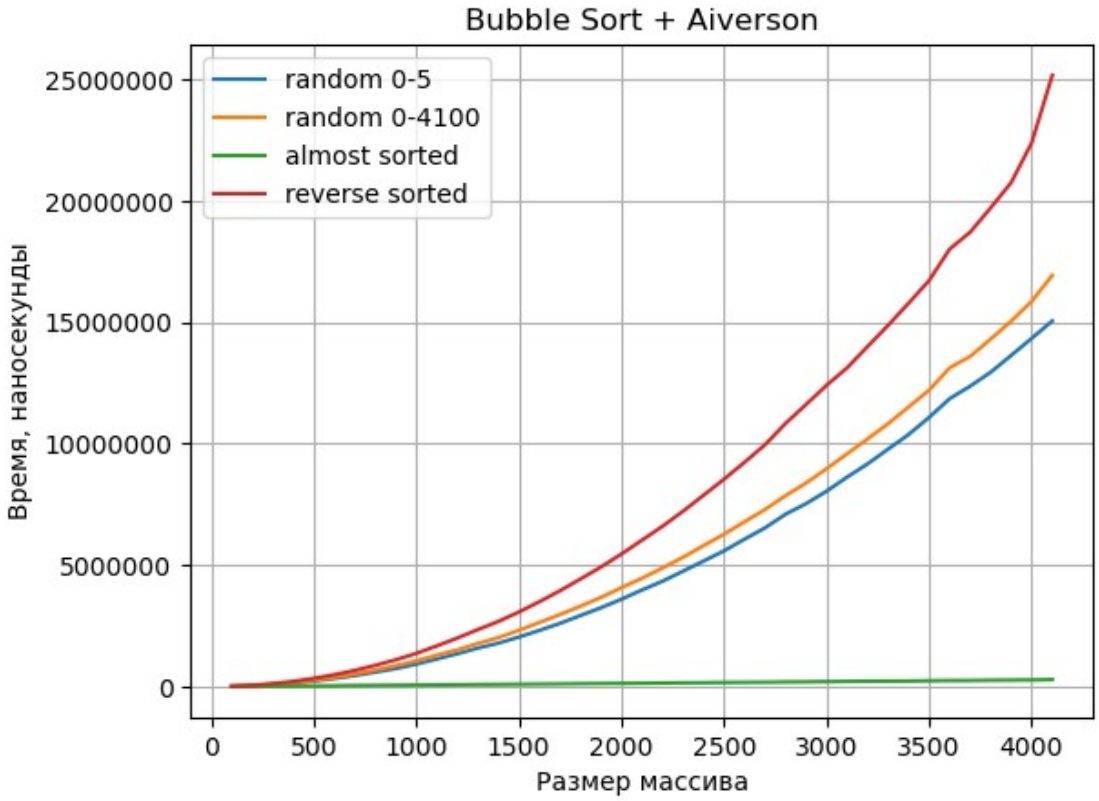
```
plt.title('Bubble Sort + Aiverson')
plt.legend(loc = 'best')
plt.show()
```




Заметное уменьшение сложности на почти отсортированном массиве благодаря первому правилу Айверсона, остальное практически без изменений

```
second_time = open("C:\\Users\\pupki\\Desktop\\second_time.csv", "r")
reader2 = csv.reader(second_time)
bubble_1_sort_r5 = []
bubble_1_sort_r4000 = []
bubble_1_sort_as = []
bubble_1_sort_rs = []
for row in reader2:
    bubble_1_sort_r5.append(int(row[8]))
    bubble_1_sort_r4000.append(int(row[9]))
    bubble_1_sort_as.append(int(row[10]))
    bubble_1_sort_rs.append(int(row[11]))
plt.plot(second_scale, bubble_1_sort_r5, label = 'random 0-5')
plt.plot(second_scale, bubble_1_sort_r4000, label = 'random 0-4100')
plt.plot(second_scale, bubble_1_sort_as, label = 'almost sorted')
plt.plot(second_scale, bubble_1_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
```

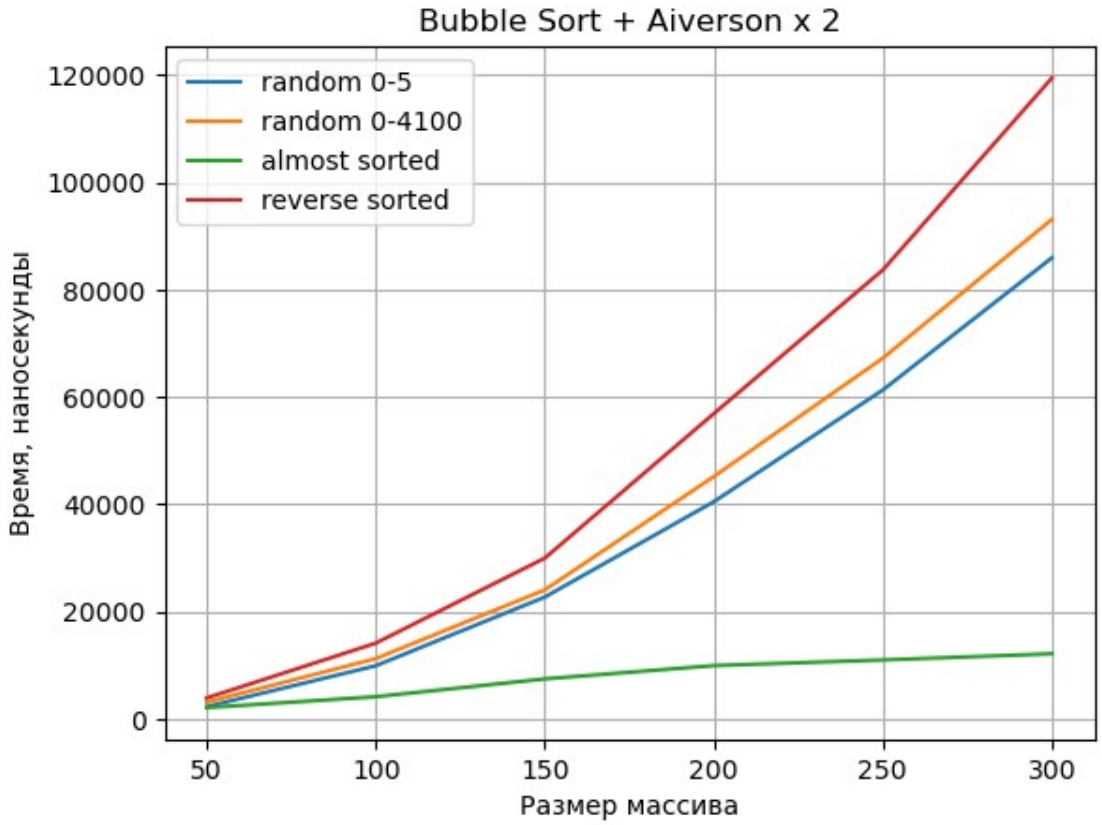
```
plt.ylabel('Время, наносекунды')
plt.title('Bubble Sort + Aiverson')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




На бОльших диапазонах почти отсортированный массив стремится к линии благодаря Айверсону

```
first_time = open("C:\\Users\\pupki\\Desktop\\first_time.csv", "r")
reader1 = csv.reader(first_time)
bubble_2_sort_r5 = []
bubble_2_sort_r4000 = []
bubble_2_sort_as = []
bubble_2_sort_rs = []
for row in reader1:
    bubble_2_sort_r5.append(int(row[12]))
    bubble_2_sort_r4000.append(int(row[13]))
    bubble_2_sort_as.append(int(row[14]))
    bubble_2_sort_rs.append(int(row[15]))
plt.plot(first_scale, bubble_2_sort_r5, label = 'random 0-5')
plt.plot(first_scale, bubble_2_sort_r4000, label = 'random 0-4100')
plt.plot(first_scale, bubble_2_sort_as, label = 'almost sorted')
plt.plot(first_scale, bubble_2_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
```

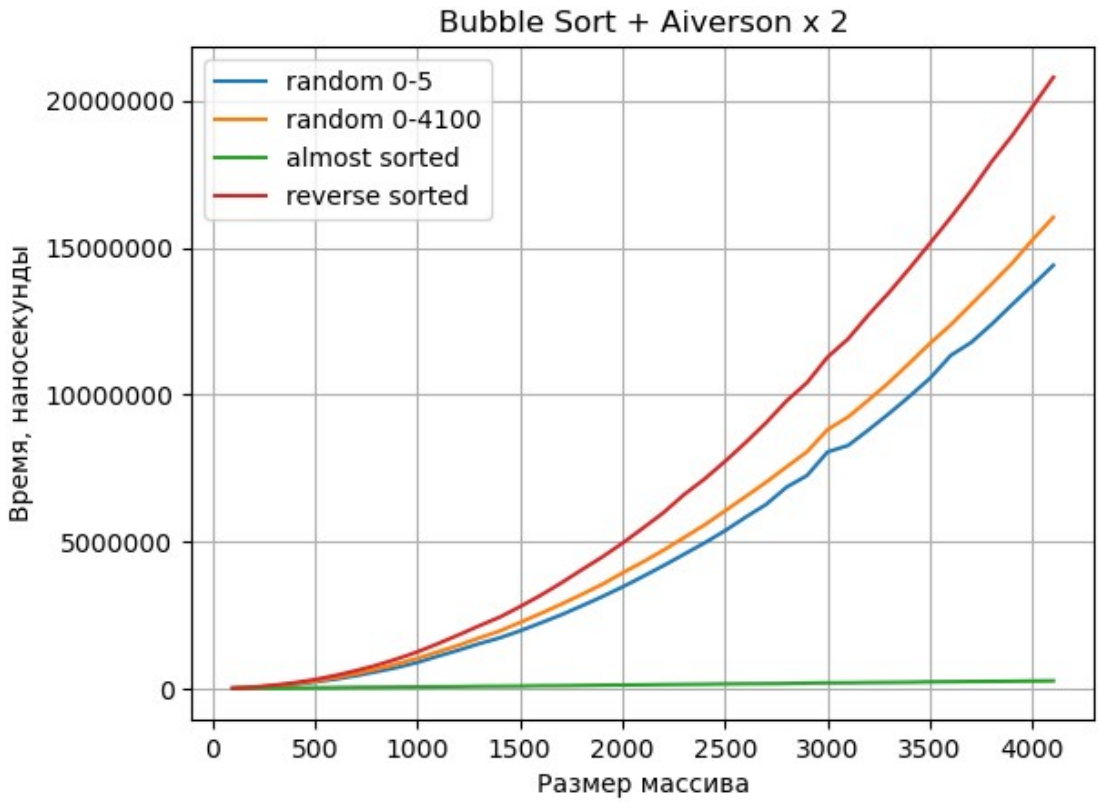
```
plt.ylabel('Время, наносекунды')
plt.title('Bubble Sort + Aiverson x 2')
plt.legend(loc = 'best')
plt.show()
```




Небольшое улучшение почти отсортированного массива

```
second_time = open("C:\\Users\\pupki\\Desktop\\second_time.csv", "r")
reader2 = csv.reader(second_time)
bubble_1_sort_r5 = []
bubble_1_sort_r4000 = []
bubble_1_sort_as = []
bubble_1_sort_rs = []
for row in reader2:
    bubble_1_sort_r5.append(int(row[12]))
    bubble_1_sort_r4000.append(int(row[13]))
    bubble_1_sort_as.append(int(row[14]))
    bubble_1_sort_rs.append(int(row[15]))
plt.plot(second_scale, bubble_1_sort_r5, label = 'random 0-5')
plt.plot(second_scale, bubble_1_sort_r4000, label = 'random 0-4100')
plt.plot(second_scale, bubble_1_sort_as, label = 'almost sorted')
plt.plot(second_scale, bubble_1_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
```

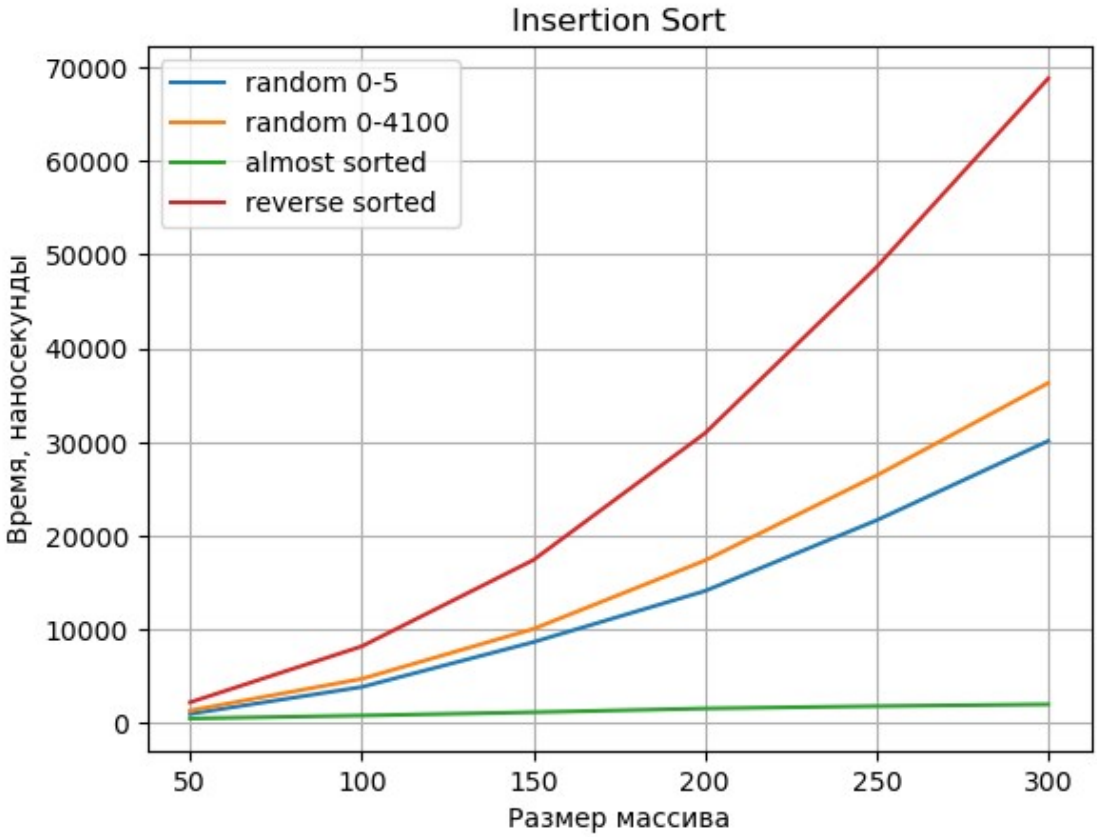
```
plt.title('Bubble Sort + Aiverson x 2')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




Заметное улучшение практически всех видов массивов

```
first_time = open("C:\\Users\\pupki\\Desktop\\first_time.csv", "r")
reader1 = csv.reader(first_time)
insertion_sort_r5 = []
insertion_sort_r4000 = []
insertion_sort_as = []
insertion_sort_rs = []
for row in reader1:
    insertion_sort_r5.append(int(row[16]))
    insertion_sort_r4000.append(int(row[17]))
    insertion_sort_as.append(int(row[18]))
    insertion_sort_rs.append(int(row[19]))
plt.plot(first_scale, insertion_sort_r5, label = 'random 0-5')
plt.plot(first_scale, insertion_sort_r4000, label = 'random 0-4100')
plt.plot(first_scale, insertion_sort_as, label = 'almost sorted')
plt.plot(first_scale, insertion_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Insertion Sort')
```

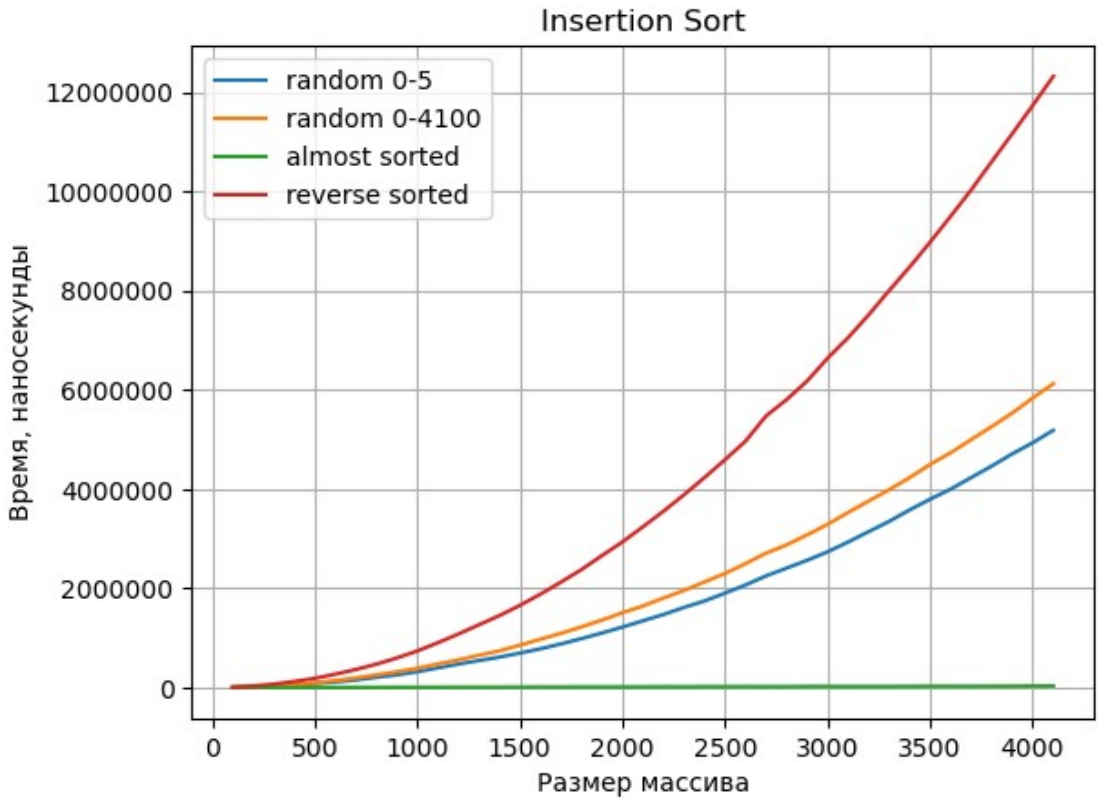
```
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




Отличное время для почти отсортированного массива, рандомные массивы похожи на линию, квадратичная сложность обратного массива из-за прохода циклом по всему массиву

```
second_time = open("C:\\Users\\pupki\\Desktop\\second_time.csv", "r")
reader2 = csv.reader(second_time)
insertion_sort_r5 = []
insertion_sort_r4000 = []
insertion_sort_as = []
insertion_sort_rs = []
for row in reader2:
    insertion_sort_r5.append(int(row[16]))
    insertion_sort_r4000.append(int(row[17]))
    insertion_sort_as.append(int(row[18]))
    insertion_sort_rs.append(int(row[19]))
plt.plot(second_scale, insertion_sort_r5, label = 'random 0-5')
plt.plot(second_scale, insertion_sort_r4000, label = 'random 0-4100')
plt.plot(second_scale, insertion_sort_as, label = 'almost sorted')
plt.plot(second_scale, insertion_sort_rs, label = 'reverse sorted')
plt.grid(True)
```

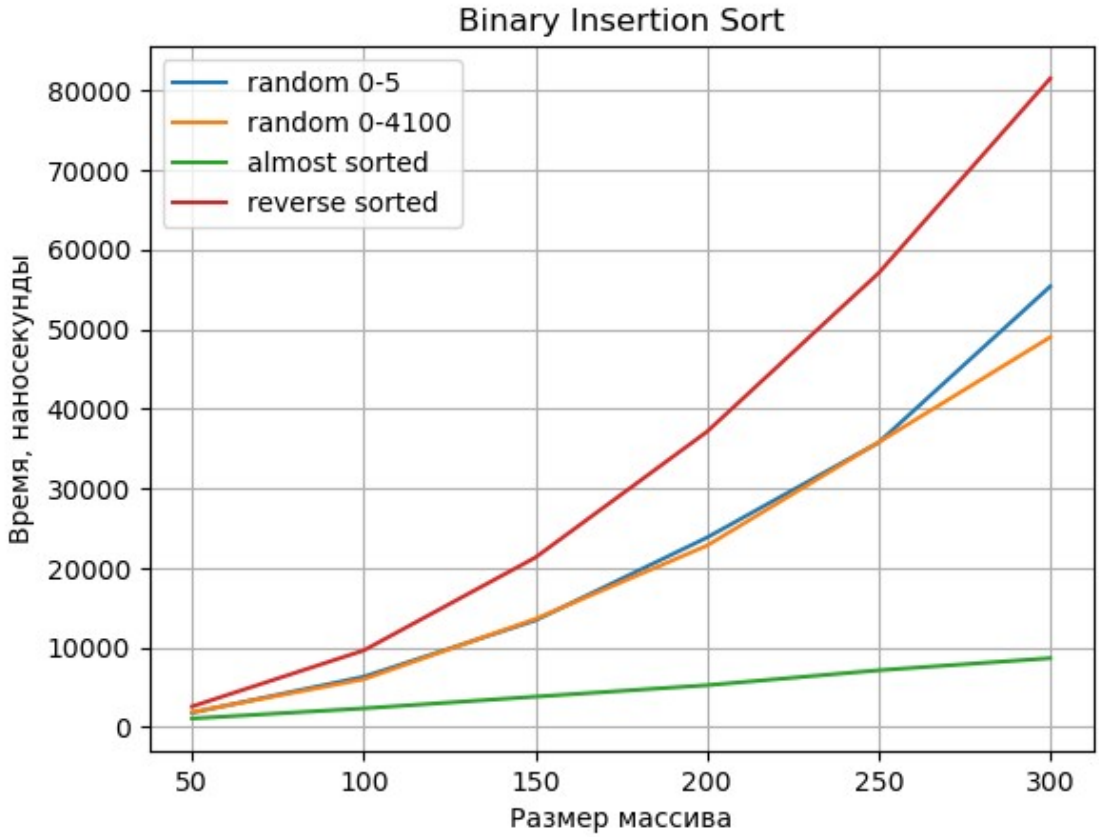
```
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Insertion Sort')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




Похожая на константу сложность при почти отсортированном, квадратичная для обратного и квадрат с меньшей константой для рандомных

```
first_time = open("C:\\Users\\pupki\\Desktop\\first_time.csv", "r")
reader1 = csv.reader(first_time)
binary_insertion_sort_r5 = []
binary_insertion_sort_r4000 = []
binary_insertion_sort_as = []
binary_insertion_sort_rs = []
for row in reader1:
    binary_insertion_sort_r5.append(int(row[20]))
    binary_insertion_sort_r4000.append(int(row[21]))
    binary_insertion_sort_as.append(int(row[22]))
    binary_insertion_sort_rs.append(int(row[23]))
plt.plot(first_scale, binary_insertion_sort_r5, label = 'random 0-5')
plt.plot(first_scale, binary_insertion_sort_r4000, label = 'random 0-
4100')
plt.plot(first_scale, binary_insertion_sort_as, label = 'almost
```

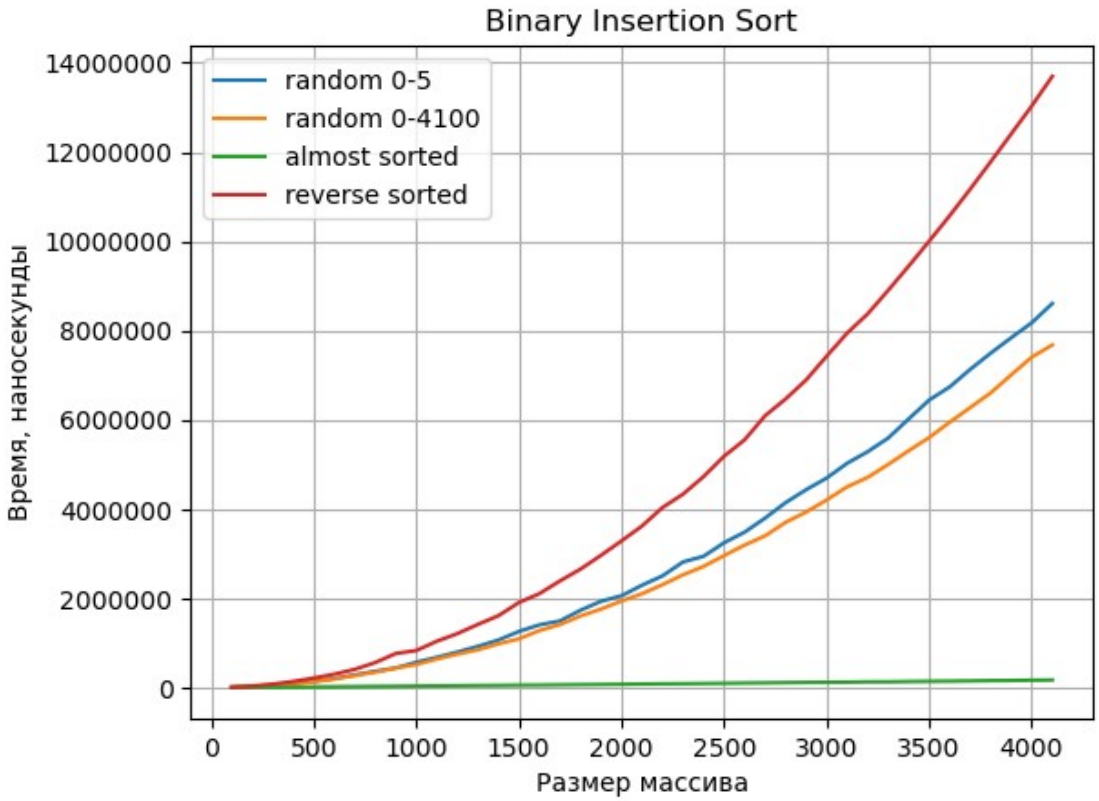
```
sorted')
plt.plot(first_scale, binary_insertion_sort_rs, label = 'reverse
sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Binary Insertion Sort')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




Ухудшение времени по сравнению с обычной вставкой

```
second_time = open("C:\\Users\\pupki\\Desktop\\second_time.csv", "r")
reader2 = csv.reader(second_time)
binary_insertion_sort_r5 = []
binary_insertion_sort_r4000 = []
binary_insertion_sort_as = []
binary_insertion_sort_rs = []
for row in reader2:
    binary_insertion_sort_r5.append(int(row[20]))
    binary_insertion_sort_r4000.append(int(row[21]))
    binary_insertion_sort_as.append(int(row[22]))
    binary_insertion_sort_rs.append(int(row[23]))
plt.plot(second_scale, binary_insertion_sort_r5, label = 'random 0-5')
```

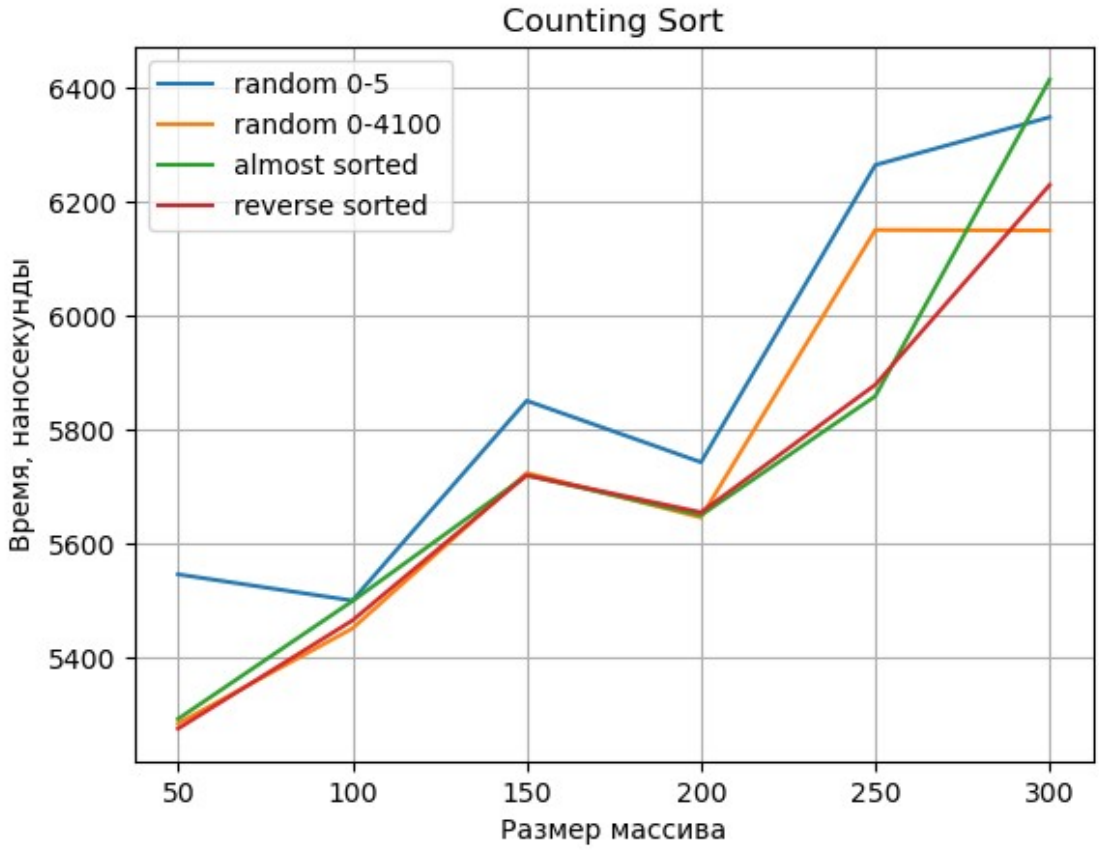
```
plt.plot(second_scale, binary_insertion_sort_r4000, label = 'random 0-
4100')
plt.plot(second_scale, binary_insertion_sort_as, label = 'almost
sorted')
plt.plot(second_scale, binary_insertion_sort_rs, label = 'reverse
sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Binary Insertion Sort')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




Ещё большее ухудшение времени

```
first_time = open("C:\\Users\\pupki\\Desktop\\first_time.csv", "r")
reader1 = csv.reader(first_time)
counting_sort_r5 = []
counting_sort_r4000 = []
counting_sort_as = []
counting_sort_rs = []
for row in reader1:
    counting_sort_r5.append(int(row[24]))
    counting_sort_r4000.append(int(row[25]))
    counting_sort_as.append(int(row[26]))
```

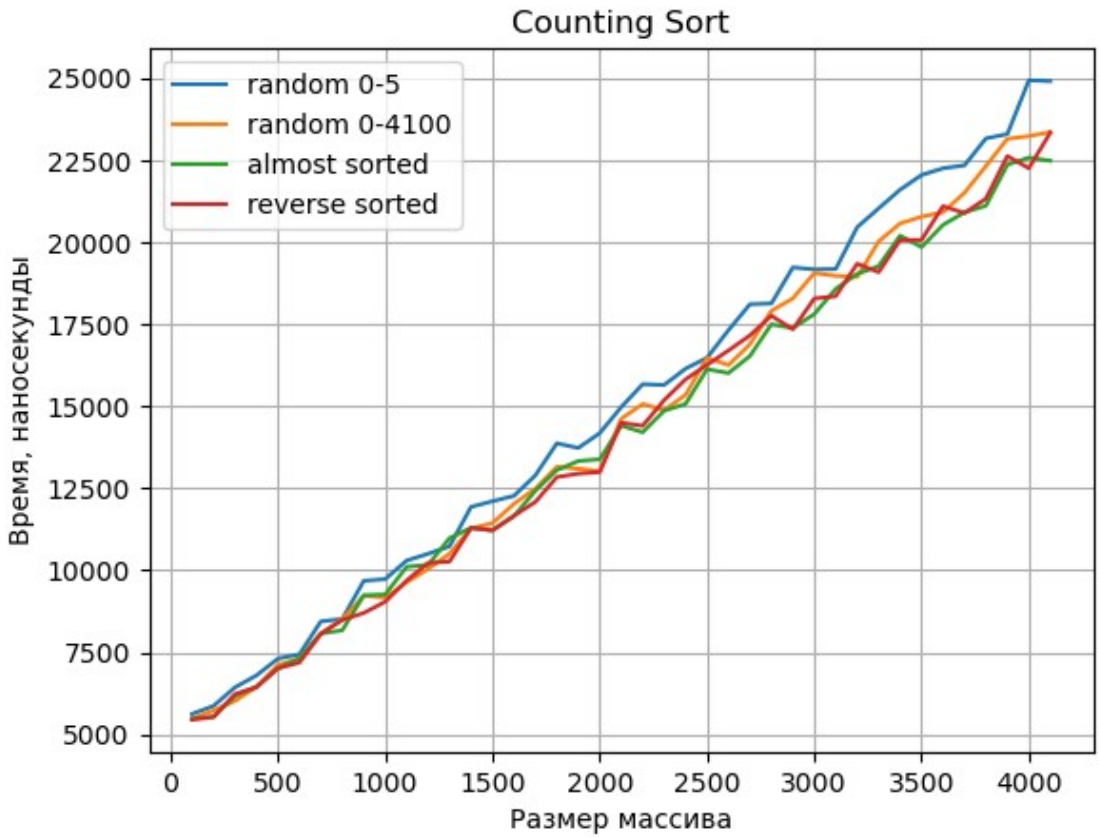
```
    counting_sort_rs.append(int(row[27]))
plt.plot(first_scale, counting_sort_r5, label = 'random 0-5')
plt.plot(first_scale, counting_sort_r4000, label = 'random 0-4100')
plt.plot(first_scale, counting_sort_as, label = 'almost sorted')
plt.plot(first_scale, counting_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Counting Sort')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




Все графики стремятся к линии, вид массива не играет роли

```
second_time = open("C:\\Users\\pupki\\Desktop\\second_time.csv", "r")
reader2 = csv.reader(second_time)
counting_sort_r5 = []
counting_sort_r4000 = []
counting_sort_as = []
counting_sort_rs = []
for row in reader2:
    counting_sort_r5.append(int(row[24]))
    counting_sort_r4000.append(int(row[25]))
```

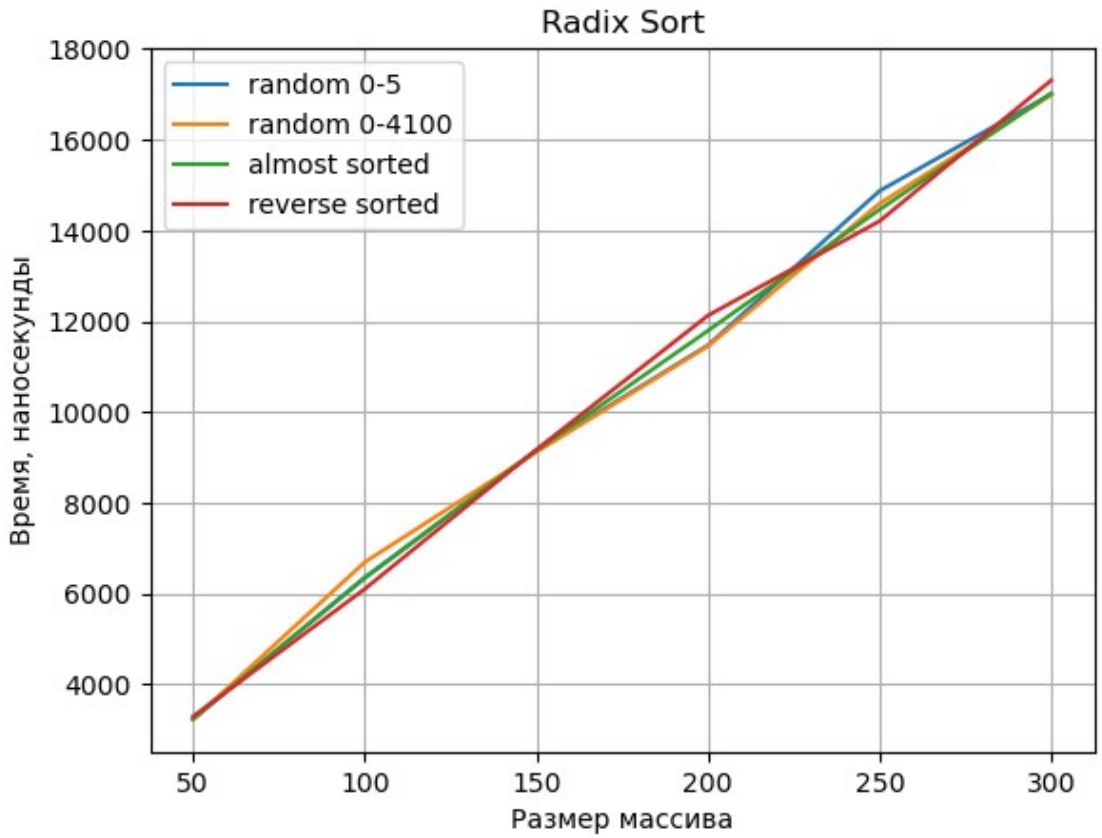
```
    counting_sort_as.append(int(row[26]))
    counting_sort_rs.append(int(row[27]))
plt.plot(second_scale, counting_sort_r5, label = 'random 0-5')
plt.plot(second_scale, counting_sort_r4000, label = 'random 0-4100')
plt.plot(second_scale, counting_sort_as, label = 'almost sorted')
plt.plot(second_scale, counting_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Counting Sort')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




Графики ещё больше походят на линию

```
first_time = open("C:\\Users\\pupki\\Desktop\\first_time.csv", "r")
reader1 = csv.reader(first_time)
radix_sort_r5 = []
radix_sort_r4000 = []
radix_sort_as = []
radix_sort_rs = []
for row in reader1:
    radix_sort_r5.append(int(row[28]))
    radix_sort_r4000.append(int(row[29]))
```

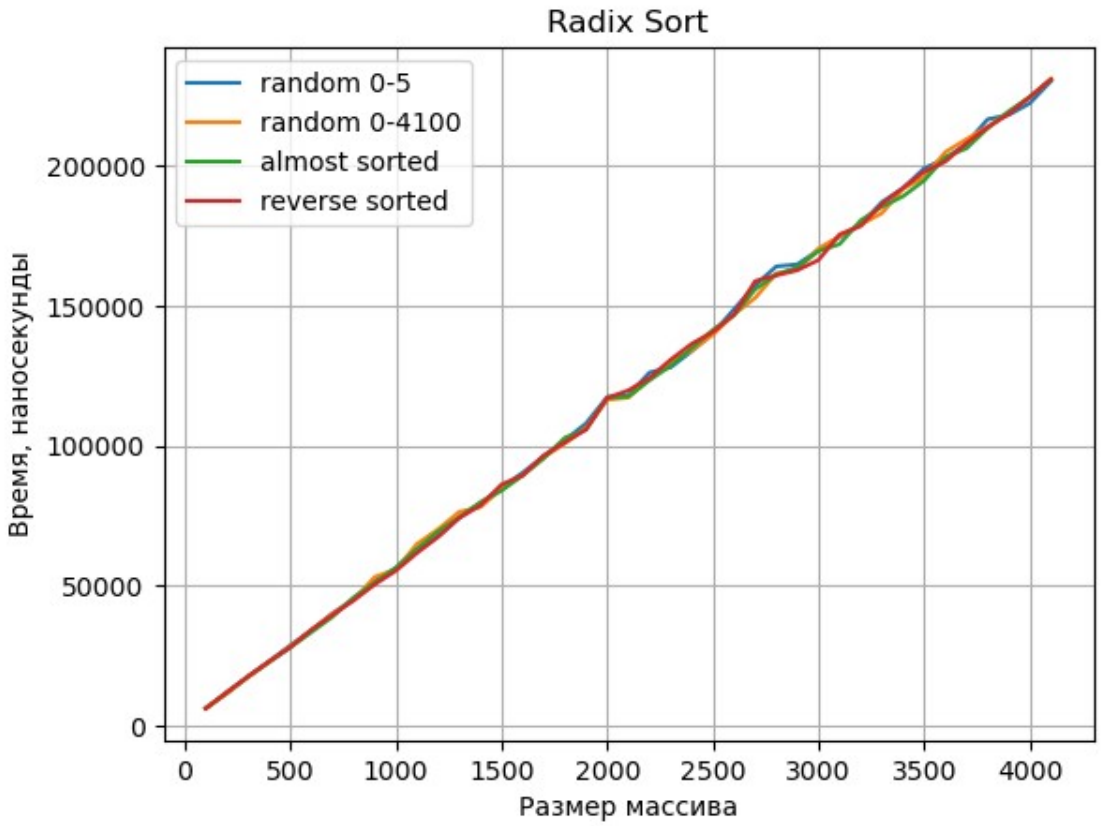
```
    radix_sort_as.append(int(row[30]))
    radix_sort_rs.append(int(row[31]))
plt.plot(first_scale, radix_sort_r5, label = 'random 0-5')
plt.plot(first_scale, radix_sort_r4000, label = 'random 0-4100')
plt.plot(first_scale, radix_sort_as, label = 'almost sorted')
plt.plot(first_scale, radix_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Radix Sort')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




Линейная сложность, вид массива не играет роли

```
second_time = open("C:\\Users\\pupki\\Desktop\\second_time.csv", "r")
reader2 = csv.reader(second_time)
radix_sort_r5 = []
radix_sort_r4000 = []
radix_sort_as = []
radix_sort_rs = []
for row in reader2:
    radix_sort_r5.append(int(row[28]))
    radix_sort_r4000.append(int(row[29]))
```

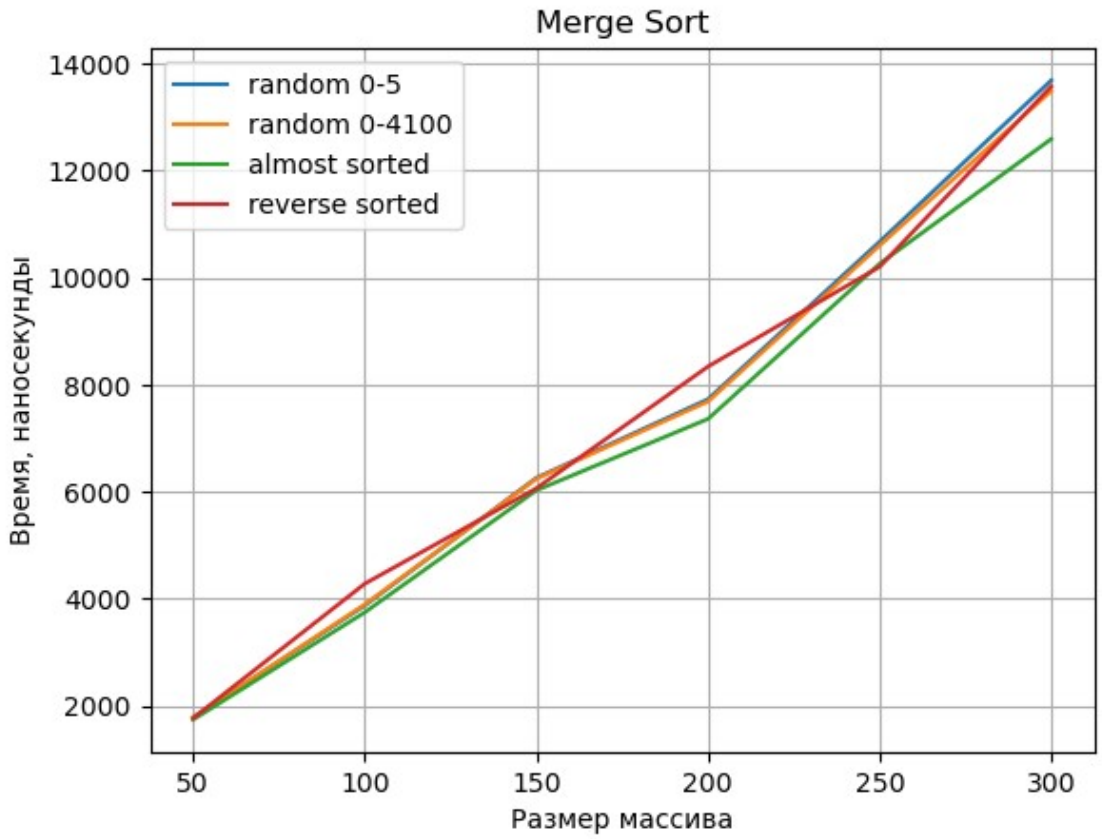
```
    radix_sort_as.append(int(row[30]))
    radix_sort_rs.append(int(row[31]))
plt.plot(second_scale, radix_sort_r5, label = 'random 0-5')
plt.plot(second_scale, radix_sort_r4000, label = 'random 0-4100')
plt.plot(second_scale, radix_sort_as, label = 'almost sorted')
plt.plot(second_scale, radix_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Radix Sort')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




Графики практически идентичны из-за устройства сортировки и показывают линейную сложность

```
first_time = open("C:\\Users\\pupki\\Desktop\\first_time.csv", "r")
reader1 = csv.reader(first_time)
merge_sort_r5 = []
merge_sort_r4000 = []
merge_sort_as = []
merge_sort_rs = []
for row in reader1:
    merge_sort_r5.append(int(row[32]))
```

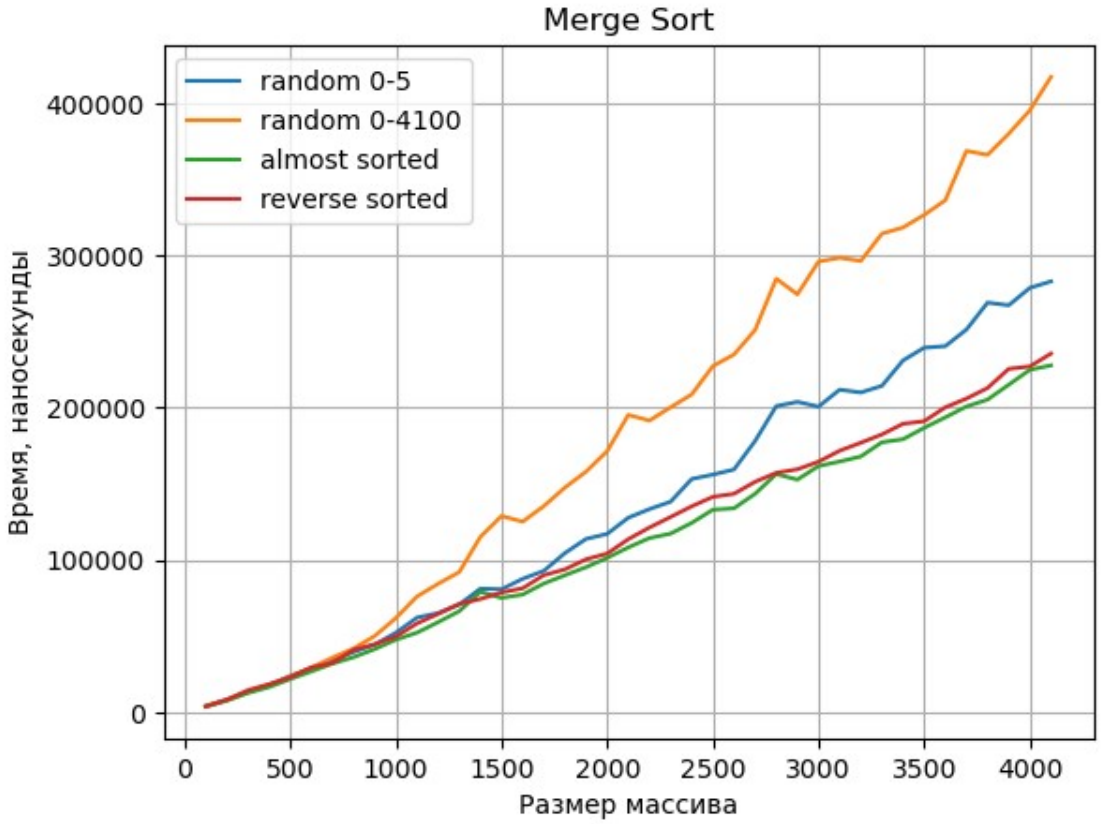
```
    merge_sort_r4000.append(int(row[33]))
    merge_sort_as.append(int(row[34]))
    merge_sort_rs.append(int(row[35]))
plt.plot(first_scale, merge_sort_r5, label = 'random 0-5')
plt.plot(first_scale, merge_sort_r4000, label = 'random 0-4100')
plt.plot(first_scale, merge_sort_as, label = 'almost sorted')
plt.plot(first_scale, merge_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Merge Sort')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




Линейная сложность, небольшие диапазоны не показывают различия в массивах

```
second_time = open("C:\\Users\\pupki\\Desktop\\second_time.csv", "r")
reader2 = csv.reader(second_time)
merge_sort_r5 = []
merge_sort_r4000 = []
merge_sort_as = []
merge_sort_rs = []
for row in reader2:
```

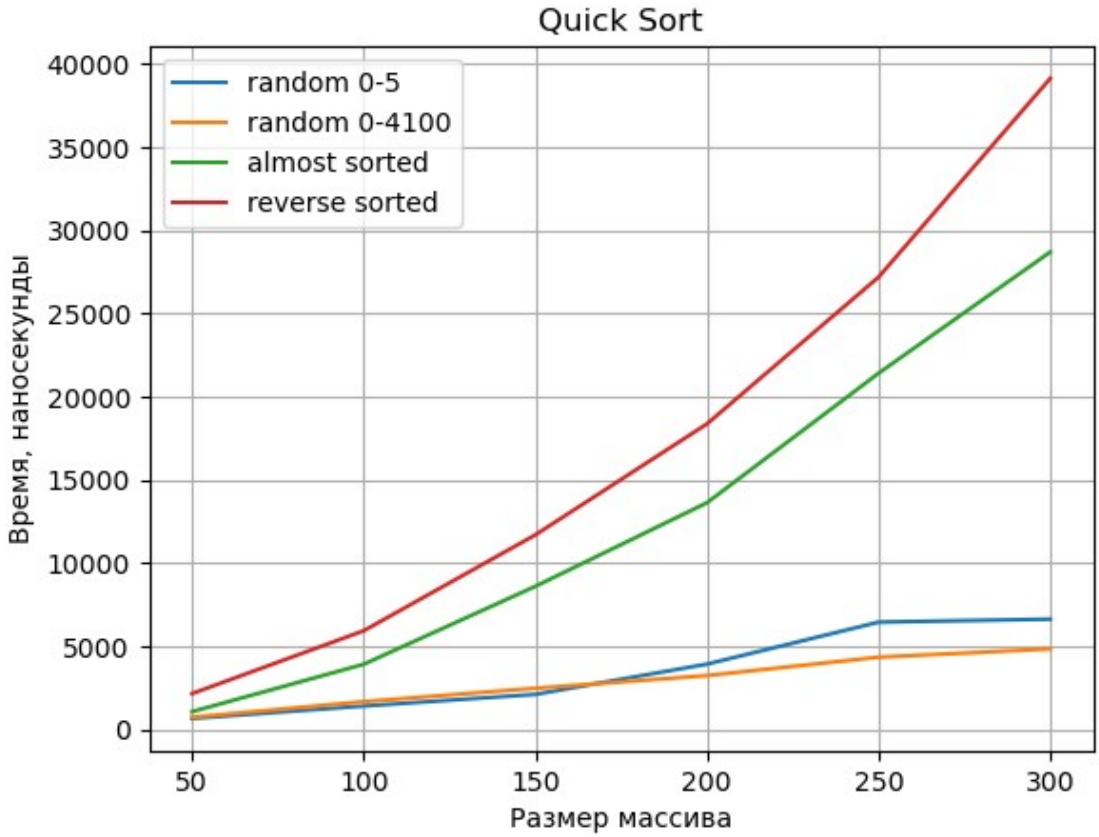
```
    merge_sort_r5.append(int(row[32]))
    merge_sort_r4000.append(int(row[33]))
    merge_sort_as.append(int(row[34]))
    merge_sort_rs.append(int(row[35]))
plt.plot(second_scale, merge_sort_r5, label = 'random 0-5')
plt.plot(second_scale, merge_sort_r4000, label = 'random 0-4100')
plt.plot(second_scale, merge_sort_as, label = 'almost sorted')
plt.plot(second_scale, merge_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Merge Sort')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




Рандомные массивы показали бОльшее время из-за устройства сортировки

```
first_time = open("C:\\Users\\pupki\\Desktop\\first_time.csv", "r")
reader1 = csv.reader(first_time)
quick_sort_r5 = []
quick_sort_r4000 = []
quick_sort_as = []
quick_sort_rs = []
```

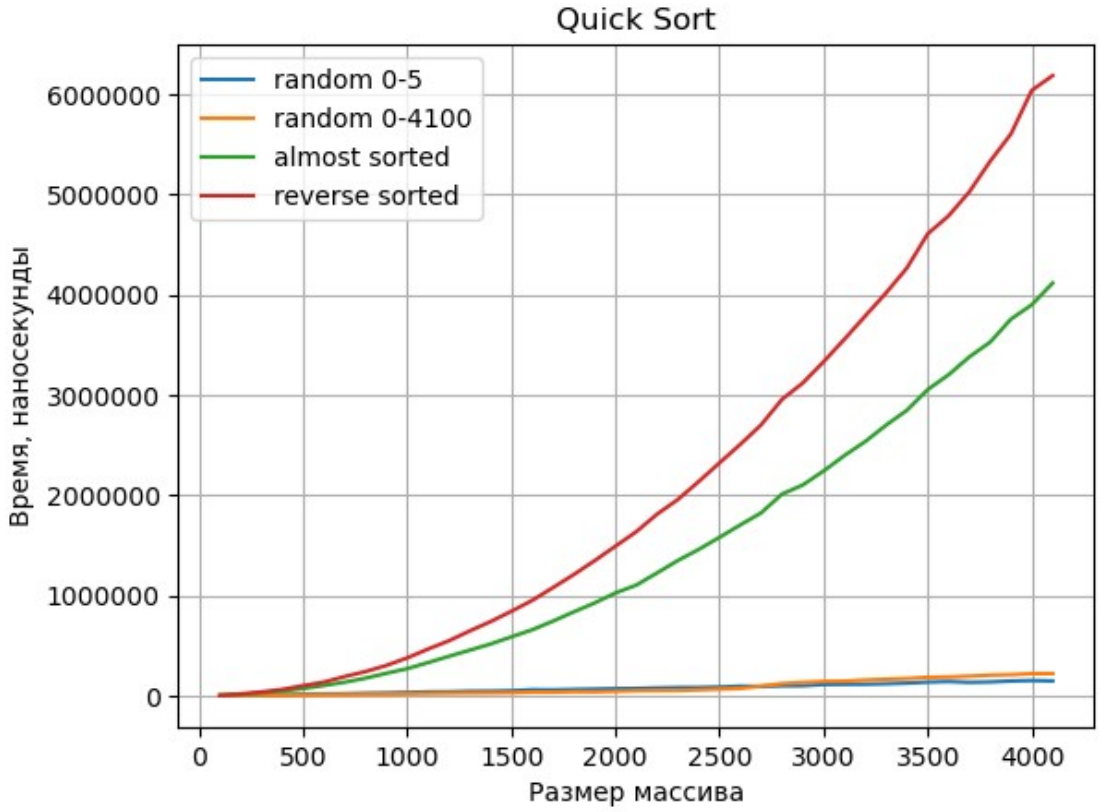
```
for row in reader1:
    quick_sort_r5.append(int(row[36]))
    quick_sort_r4000.append(int(row[37]))
    quick_sort_as.append(int(row[38]))
    quick_sort_rs.append(int(row[39]))
plt.plot(first_scale, quick_sort_r5, label = 'random 0-5')
plt.plot(first_scale, quick_sort_r4000, label = 'random 0-4100')
plt.plot(first_scale, quick_sort_as, label = 'almost sorted')
plt.plot(first_scale, quick_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Quick Sort')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




Из-за реализация сортировки с опорным первым элементом, обратно и частично отсортированные массивы показали квадратную сложность, так как в этом случае быстрая сортировка похожа на сортировку вставками

```
second_time = open("C:\\Users\\pupki\\Desktop\\second_time.csv", "r")
reader2 = csv.reader(second_time)
quick_sort_r5 = []
```

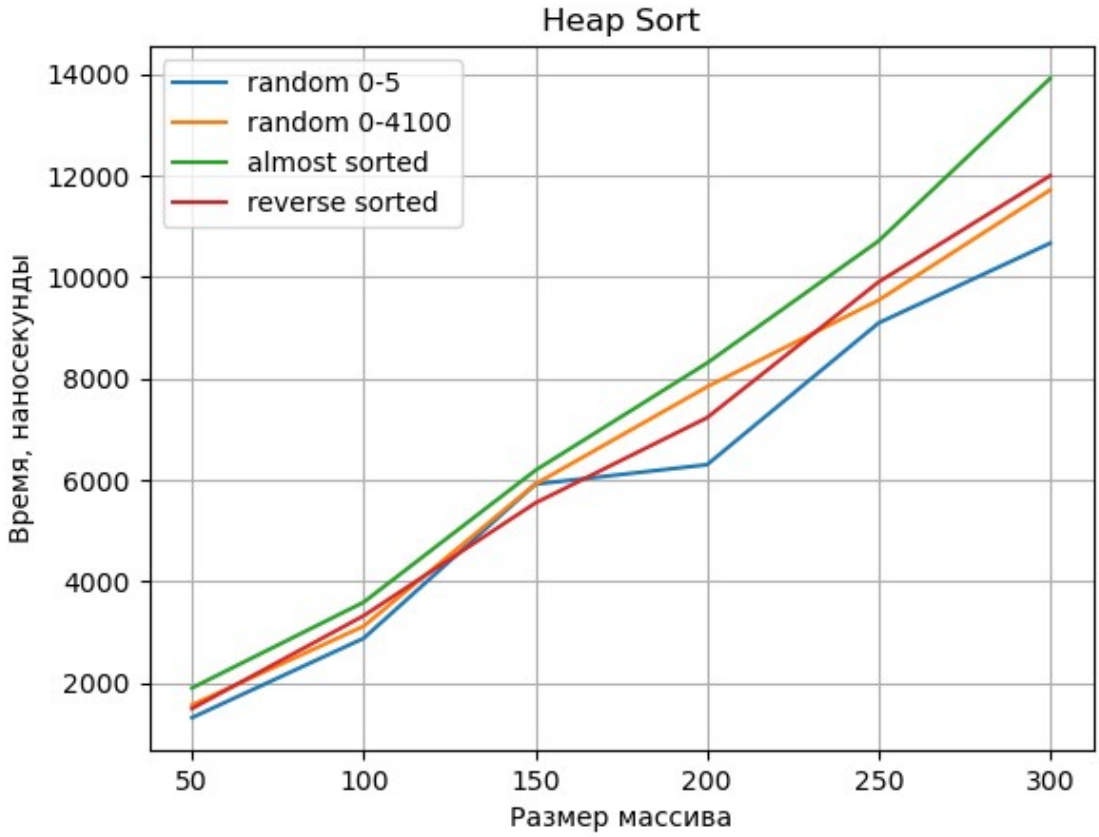
```
quick_sort_r4000 = []
quick_sort_as = []
quick_sort_rs = []
for row in reader2:
    quick_sort_r5.append(int(row[36]))
    quick_sort_r4000.append(int(row[37]))
    quick_sort_as.append(int(row[38]))
    quick_sort_rs.append(int(row[39]))
plt.plot(second_scale, quick_sort_r5, label = 'random 0-5')
plt.plot(second_scale, quick_sort_r4000, label = 'random 0-4100')
plt.plot(second_scale, quick_sort_as, label = 'almost sorted')
plt.plot(second_scale, quick_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Quick Sort')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




Ещё бОльший уход в квадрат для не рандомных массивов, для рандомных массивов быстрая сортировка подтверждает своё название

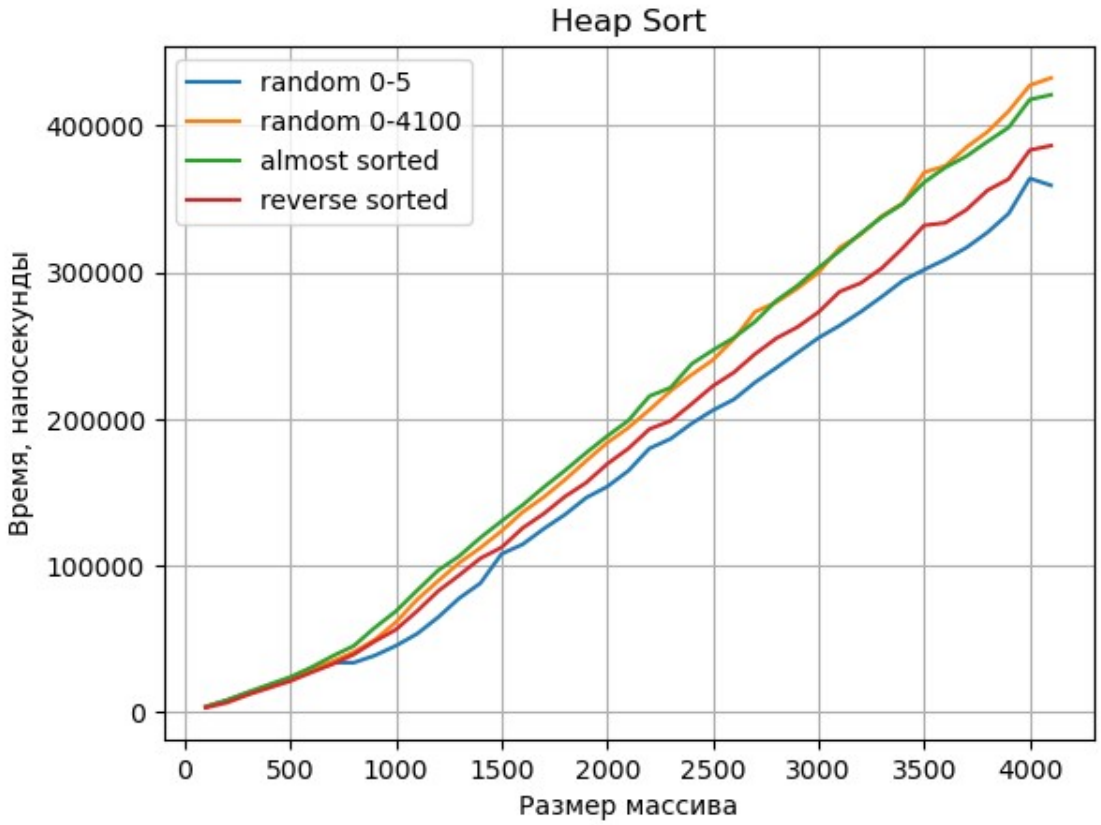
```
first_time = open("C:\\Users\\pupki\\Desktop\\first_time.csv", "r")
reader1 = csv.reader(first_time)
```

```
heap_sort_r5 = []
heap_sort_r4000 = []
heap_sort_as = []
heap_sort_rs = []
for row in reader1:
    heap_sort_r5.append(int(row[40]))
    heap_sort_r4000.append(int(row[41]))
    heap_sort_as.append(int(row[42]))
    heap_sort_rs.append(int(row[43]))
plt.plot(first_scale, heap_sort_r5, label = 'random 0-5')
plt.plot(first_scale, heap_sort_r4000, label = 'random 0-4100')
plt.plot(first_scale, heap_sort_as, label = 'almost sorted')
plt.plot(first_scale, heap_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Heap Sort')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```



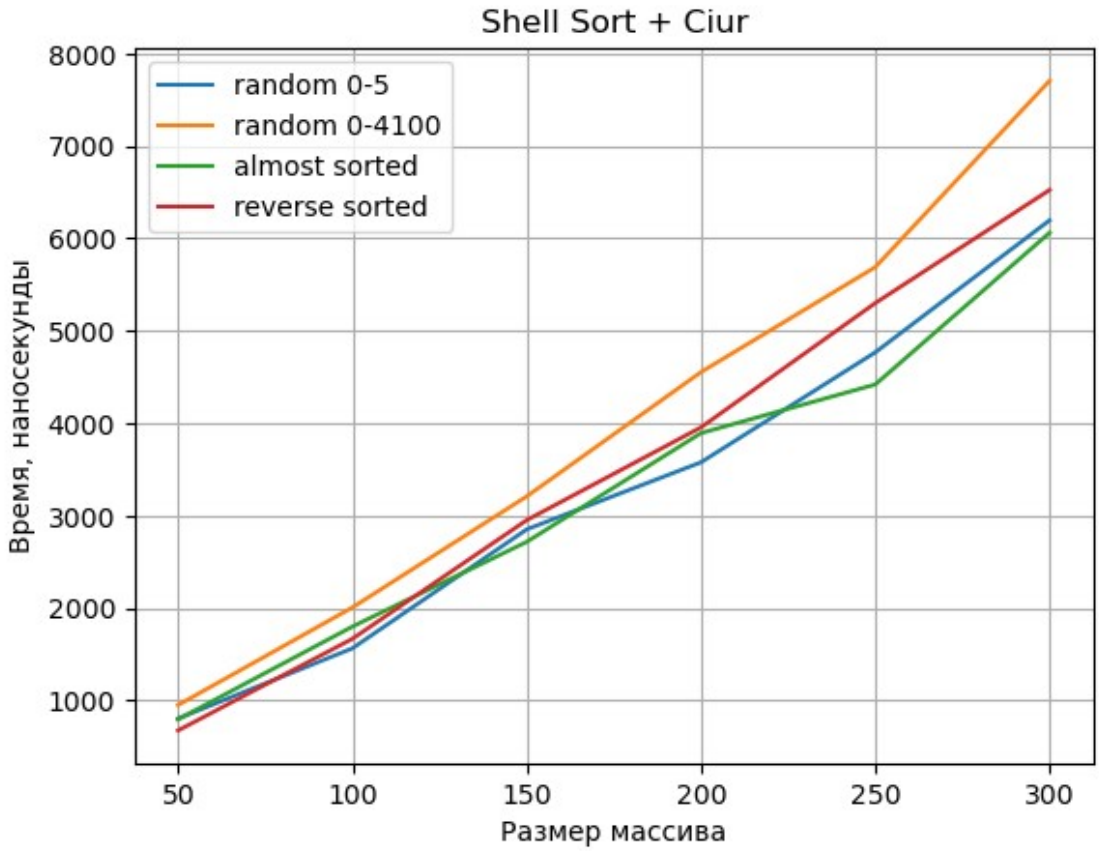

Схожее время всех массивов, похожее на nlogn из-за внутреннего устройства сортировки

```
second_time = open("C:\\Users\\pupki\\Desktop\\second_time.csv", "r")
reader2 = csv.reader(second_time)
heap_sort_r5 = []
heap_sort_r4000 = []
heap_sort_as = []
heap_sort_rs = []
for row in reader2:
    heap_sort_r5.append(int(row[40]))
    heap_sort_r4000.append(int(row[41]))
    heap_sort_as.append(int(row[42]))
    heap_sort_rs.append(int(row[43]))
plt.plot(second_scale, heap_sort_r5, label = 'random 0-5')
plt.plot(second_scale, heap_sort_r4000, label = 'random 0-4100')
plt.plot(second_scale, heap_sort_as, label = 'almost sorted')
plt.plot(second_scale, heap_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Heap Sort')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```



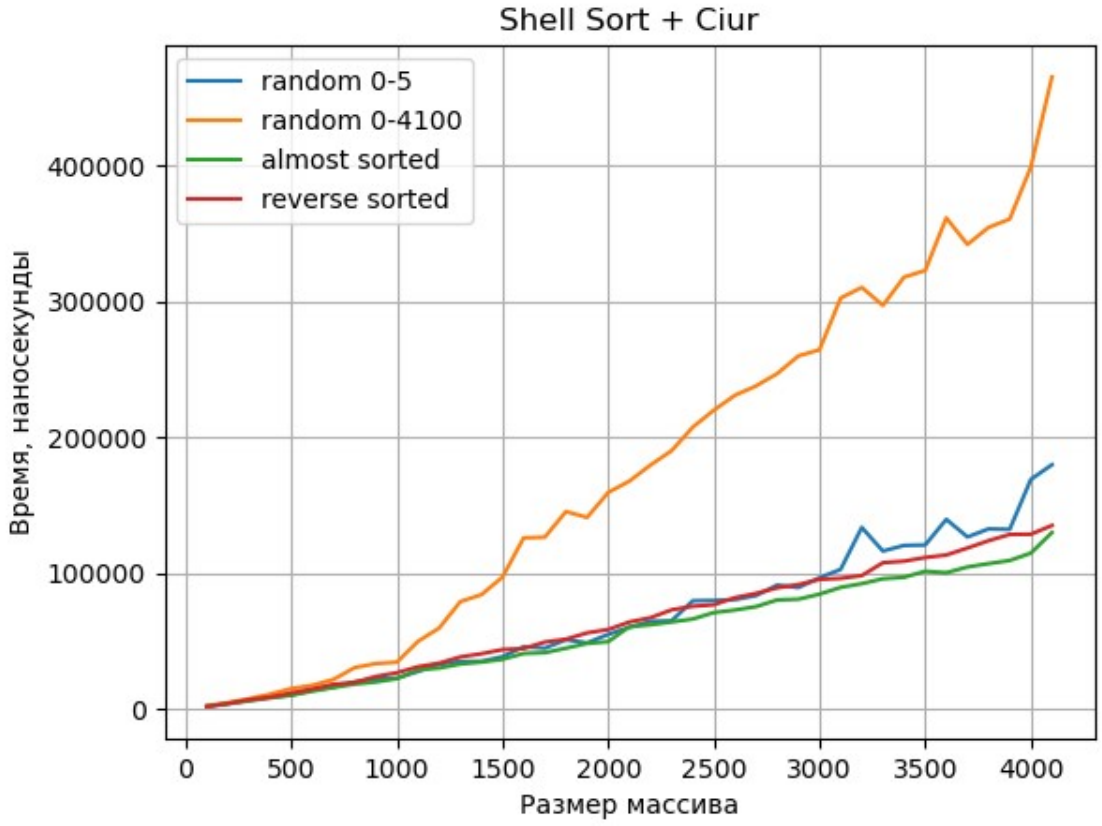

БОльшие диапазоны подтверждают догадки, так как вне зависимости от массива производится одинаковое количество вызовов makeHeap

```
first_time = open("C:\\Users\\pupki\\Desktop\\first_time.csv", "r")
reader1 = csv.reader(first_time)
shell1_sort_r5 = []
shell1_sort_r4000 = []
shell1_sort_as = []
shell1_sort_rs = []
for row in reader1:
    shell1_sort_r5.append(int(row[44]))
    shell1_sort_r4000.append(int(row[45]))
    shell1_sort_as.append(int(row[46]))
    shell1_sort_rs.append(int(row[47]))
plt.plot(first_scale, shell1_sort_r5, label = 'random 0-5')
plt.plot(first_scale, shell1_sort_r4000, label = 'random 0-4100')
plt.plot(first_scale, shell1_sort_as, label = 'almost sorted')
plt.plot(first_scale, shell1_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Shell Sort + Ciur')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```



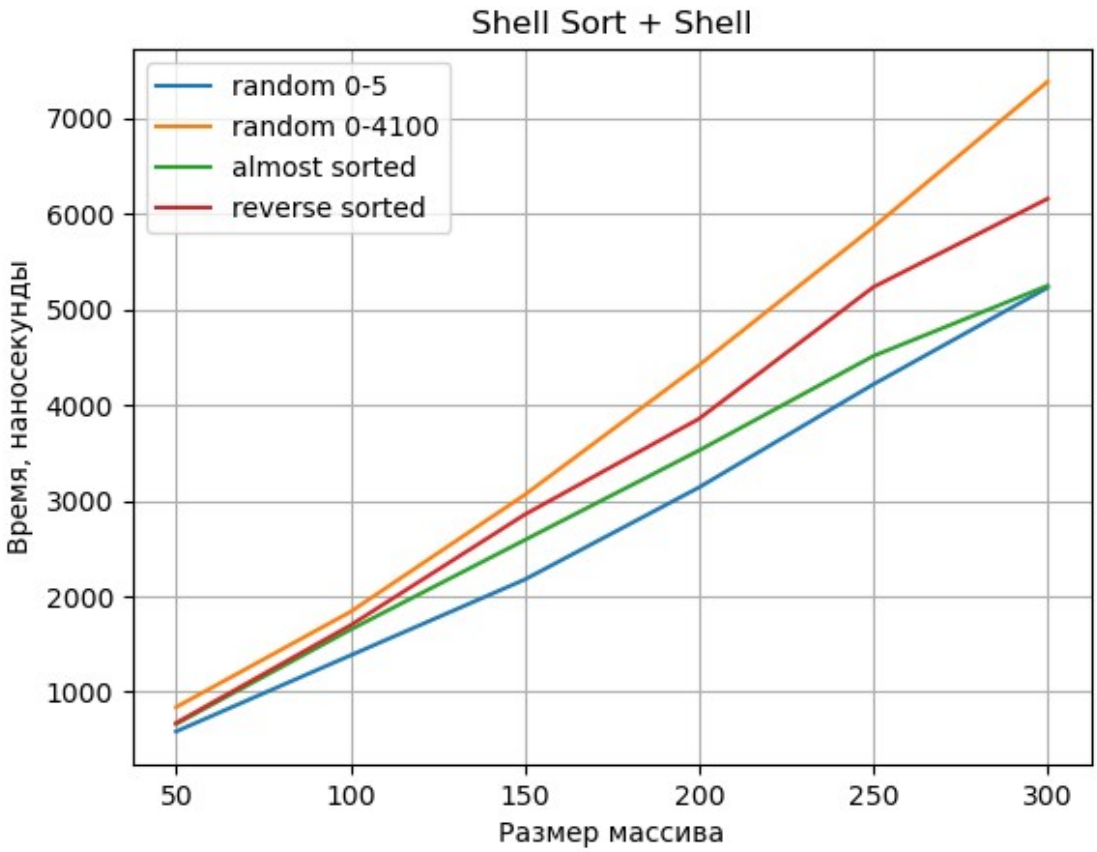
Заметное увеличение времени для больших рандомных чисел даже на небольших диапазонах

```
second_time = open("C:\\Users\\pupki\\Desktop\\second_time.csv", "r")
reader2 = csv.reader(second_time)
shell1_sort_r5 = []
shell1_sort_r4000 = []
shell1_sort_as = []
shell1_sort_rs = []
for row in reader2:
    shell1_sort_r5.append(int(row[44]))
    shell1_sort_r4000.append(int(row[45]))
    shell1_sort_as.append(int(row[46]))
    shell1_sort_rs.append(int(row[47]))
plt.plot(second_scale, shell1_sort_r5, label = 'random 0-5')
plt.plot(second_scale, shell1_sort_r4000, label = 'random 0-4100')
plt.plot(second_scale, shell1_sort_as, label = 'almost sorted')
plt.plot(second_scale, shell1_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Shell Sort + Ciur')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```



Заметный уход в квадрат для больших рандомных чисел, остальные массивы стремятся к линии

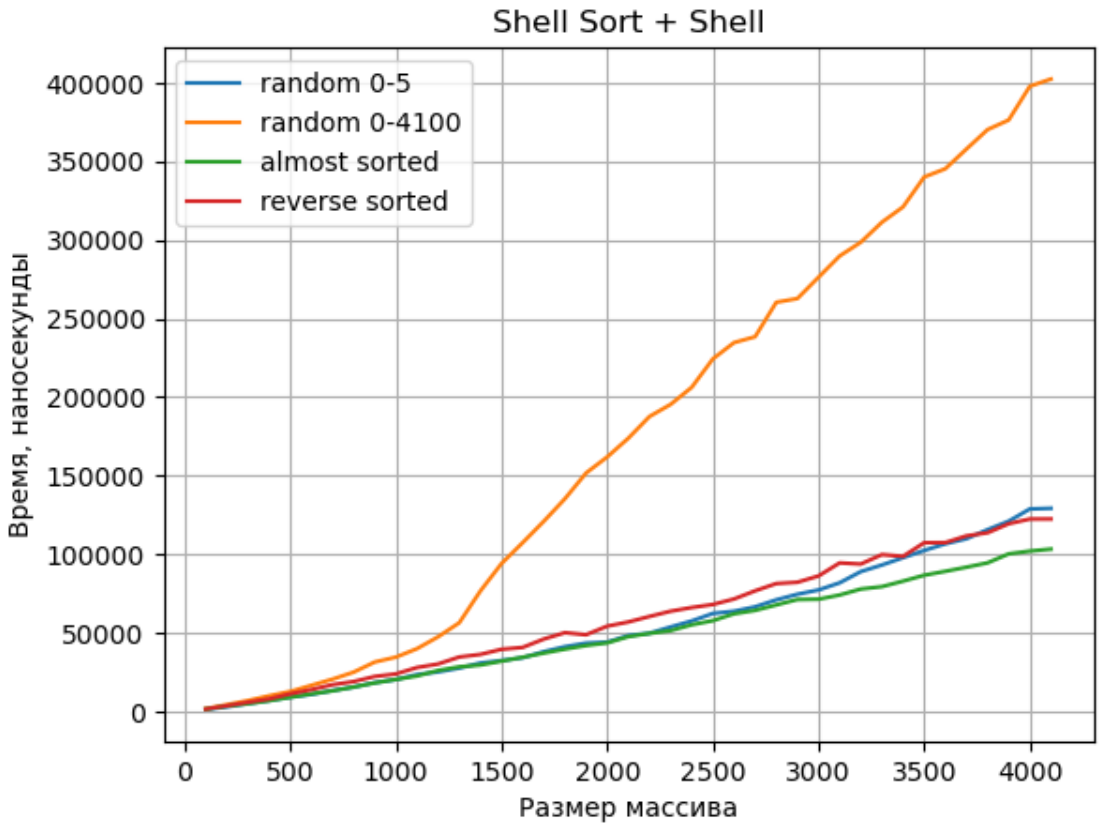
```
first_time = open("C:\\Users\\pupki\\Desktop\\first_time.csv", "r")
reader1 = csv.reader(first_time)
shell2_sort_r5 = []
shell2_sort_r4000 = []
shell2_sort_as = []
shell2_sort_rs = []
for row in reader1:
    shell2_sort_r5.append(int(row[48]))
    shell2_sort_r4000.append(int(row[49]))
    shell2_sort_as.append(int(row[50]))
    shell2_sort_rs.append(int(row[51]))
plt.plot(first_scale, shell2_sort_r5, label = 'random 0-5')
plt.plot(first_scale, shell2_sort_r4000, label = 'random 0-4100')
plt.plot(first_scale, shell2_sort_as, label = 'almost sorted')
plt.plot(first_scale, shell2_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Shell Sort + Shell')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




Похожая картина на последовательность Циура, однако время немного уменьшилось на всех видах массивов

```
second_time = open("C:\\Users\\pupki\\Desktop\\second_time.csv", "r")
reader2 = csv.reader(second_time)
shell2_sort_r5 = []
shell2_sort_r4000 = []
shell2_sort_as = []
shell2_sort_rs = []
for row in reader2:
    shell2_sort_r5.append(int(row[48]))
    shell2_sort_r4000.append(int(row[49]))
    shell2_sort_as.append(int(row[50]))
    shell2_sort_rs.append(int(row[51]))
plt.plot(second_scale, shell2_sort_r5, label = 'random 0-5')
plt.plot(second_scale, shell2_sort_r4000, label = 'random 0-4100')
plt.plot(second_scale, shell2_sort_as, label = 'almost sorted')
plt.plot(second_scale, shell2_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Shell Sort + Shell')
plt.legend(loc = 'best')
```

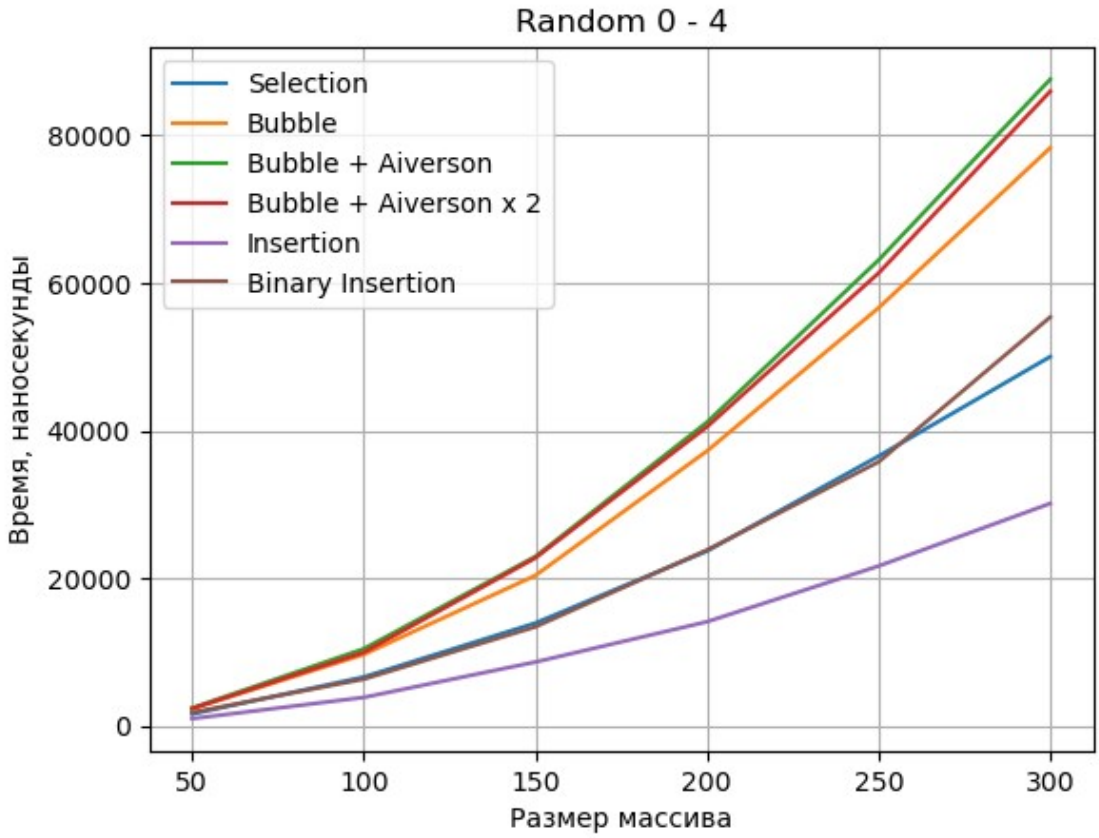
```
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




Заметное снижение времени по сравнению с последовательностью Циура, поэтому на практике более простая последовательность Шелла показывает лучшее время

```
first_time = open("C:\\Users\\pupki\\Desktop\\first_time.csv", "r")
reader1 = csv.reader(first_time)
selection_sort = []
bubble_sort = []
bubble_1_sort = []
bubble_2_sort = []
insertion_sort = []
binary_insertion_sort = []
for row in reader1:
    selection_sort.append(int(row[0]))
    bubble_sort.append(int(row[4]))
    bubble_1_sort.append(int(row[8]))
    bubble_2_sort.append(int(row[12]))
    insertion_sort.append(int(row[16]))
    binary_insertion_sort.append(int(row[20]))
plt.plot(first_scale, selection_sort, label = 'Selection')
plt.plot(first_scale, bubble_sort, label = 'Bubble')
plt.plot(first_scale, bubble_1_sort, label = 'Bubble + Aiverson')
```

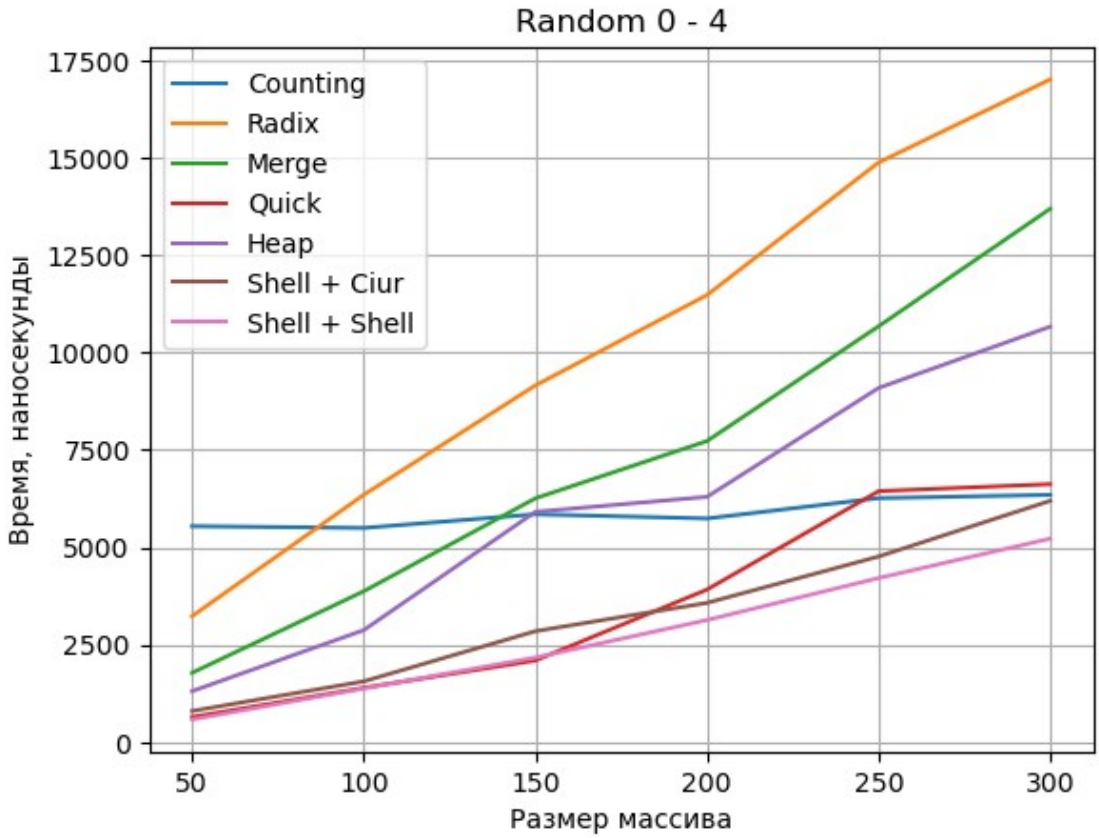
```
plt.plot(first_scale, bubble_2_sort, label = 'Bubble + Aiverson x 2')
plt.plot(first_scale, insertion_sort, label = 'Insertion')
plt.plot(first_scale, binary_insertion_sort, label = 'Binary
Insertion')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Random 0 - 4')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




Группа сортировок, стремящихся к квадрату, на небольших диапазонах вставка показывает наилучшее время, пузырёк наихудшее, причём условия Айверсона только ухудшают его

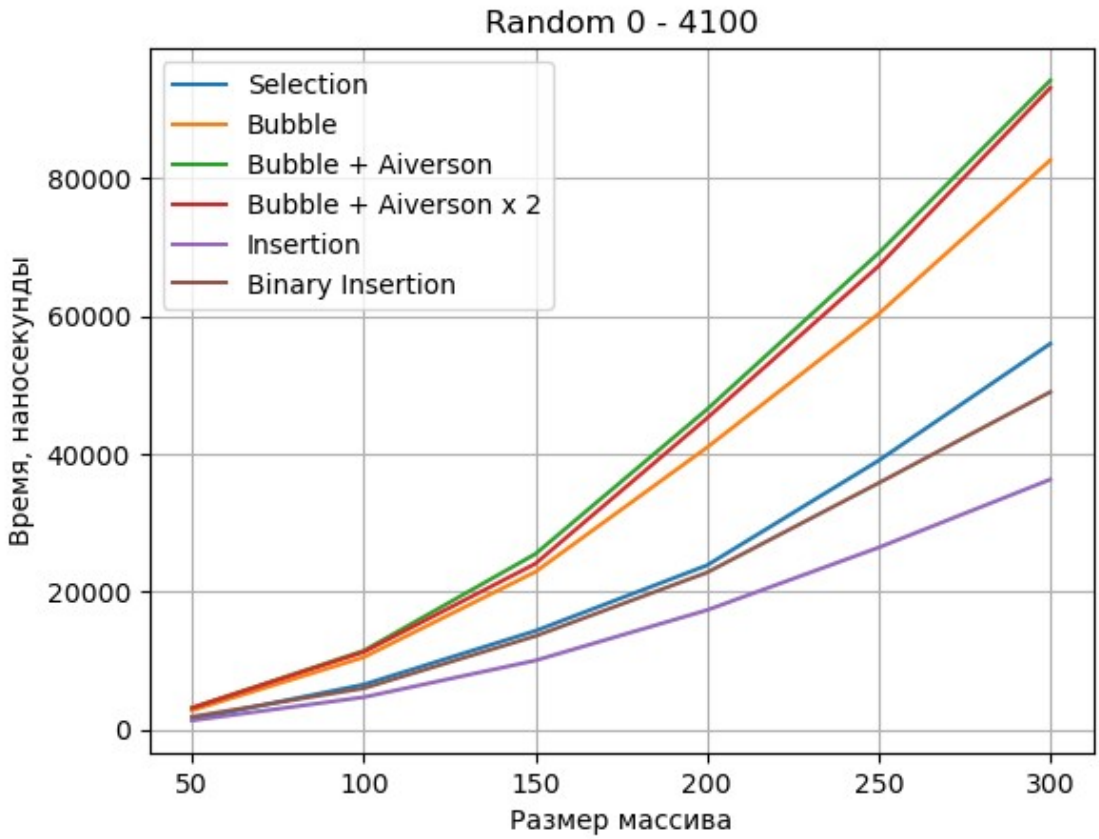
```
first_time = open("C:\\Users\\pupki\\Desktop\\first_time.csv", "r")
reader1 = csv.reader(first_time)
counting_sort = []
radix_sort = []
merge_sort = []
quick_sort = []
heap_sort = []
shell_1_sort = []
```

```
shell_2_sort = []
for row in reader1:
    counting_sort.append(int(row[24]))
    radix_sort.append(int(row[28]))
    merge_sort.append(int(row[32]))
    quick_sort.append(int(row[36]))
    heap_sort.append(int(row[40]))
    shell_1_sort.append(int(row[44]))
    shell_2_sort.append(int(row[48]))
plt.plot(first_scale, counting_sort, label = 'Counting')
plt.plot(first_scale, radix_sort, label = 'Radix')
plt.plot(first_scale, merge_sort, label = 'Merge')
plt.plot(first_scale, quick_sort, label = 'Quick')
plt.plot(first_scale, heap_sort, label = 'Heap')
plt.plot(first_scale, shell_1_sort, label = 'Shell + Ciur')
plt.plot(first_scale, shell_2_sort, label = 'Shell + Shell')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Random 0 - 4')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```



Группа не квадратных сортировок, все стремятся к линии, сортировки Шелла показывают наилучшее время благодаря маленькому диапазону чисел, сортировка подсчётом практически константа

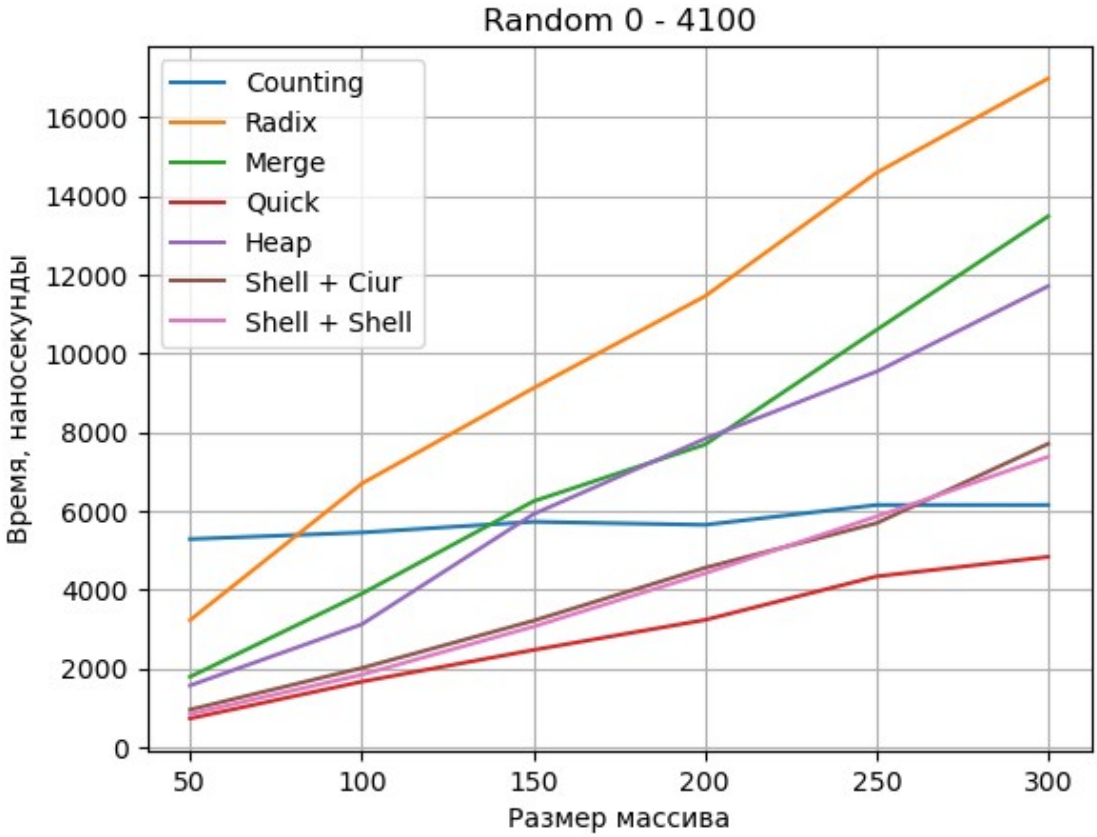
```
first_time = open("C:\\Users\\pupki\\Desktop\\first_time.csv", "r")
reader1 = csv.reader(first_time)
selection_sort = []
bubble_sort = []
bubble_1_sort = []
bubble_2_sort = []
insertion_sort = []
binary_insertion_sort = []
for row in reader1:
    selection_sort.append(int(row[1]))
    bubble_sort.append(int(row[5]))
    bubble_1_sort.append(int(row[9]))
    bubble_2_sort.append(int(row[13]))
    insertion_sort.append(int(row[17]))
    binary_insertion_sort.append(int(row[21]))
plt.plot(first_scale, selection_sort, label = 'Selection')
plt.plot(first_scale, bubble_sort, label = 'Bubble')
plt.plot(first_scale, bubble_1_sort, label = 'Bubble + Aiverson')
plt.plot(first_scale, bubble_2_sort, label = 'Bubble + Aiverson x 2')
plt.plot(first_scale, insertion_sort, label = 'Insertion')
plt.plot(first_scale, binary_insertion_sort, label = 'Binary
Insertion')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Random 0 - 4100')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




Отсутствие изменений по сравнению с меньшим диапазоном чисел

```
first_time = open("C:\\Users\\pupki\\Desktop\\first_time.csv", "r")
reader1 = csv.reader(first_time)
counting_sort = []
radix_sort = []
merge_sort = []
quick_sort = []
heap_sort = []
shell_1_sort = []
shell_2_sort = []
for row in reader1:
    counting_sort.append(int(row[25]))
    radix_sort.append(int(row[29]))
    merge_sort.append(int(row[33]))
    quick_sort.append(int(row[37]))
    heap_sort.append(int(row[41]))
    shell_1_sort.append(int(row[45]))
    shell_2_sort.append(int(row[49]))
plt.plot(first_scale, counting_sort, label = 'Counting')
plt.plot(first_scale, radix_sort, label = 'Radix')
plt.plot(first_scale, merge_sort, label = 'Merge')
plt.plot(first_scale, quick_sort, label = 'Quick')
plt.plot(first_scale, heap_sort, label = 'Heap')
```

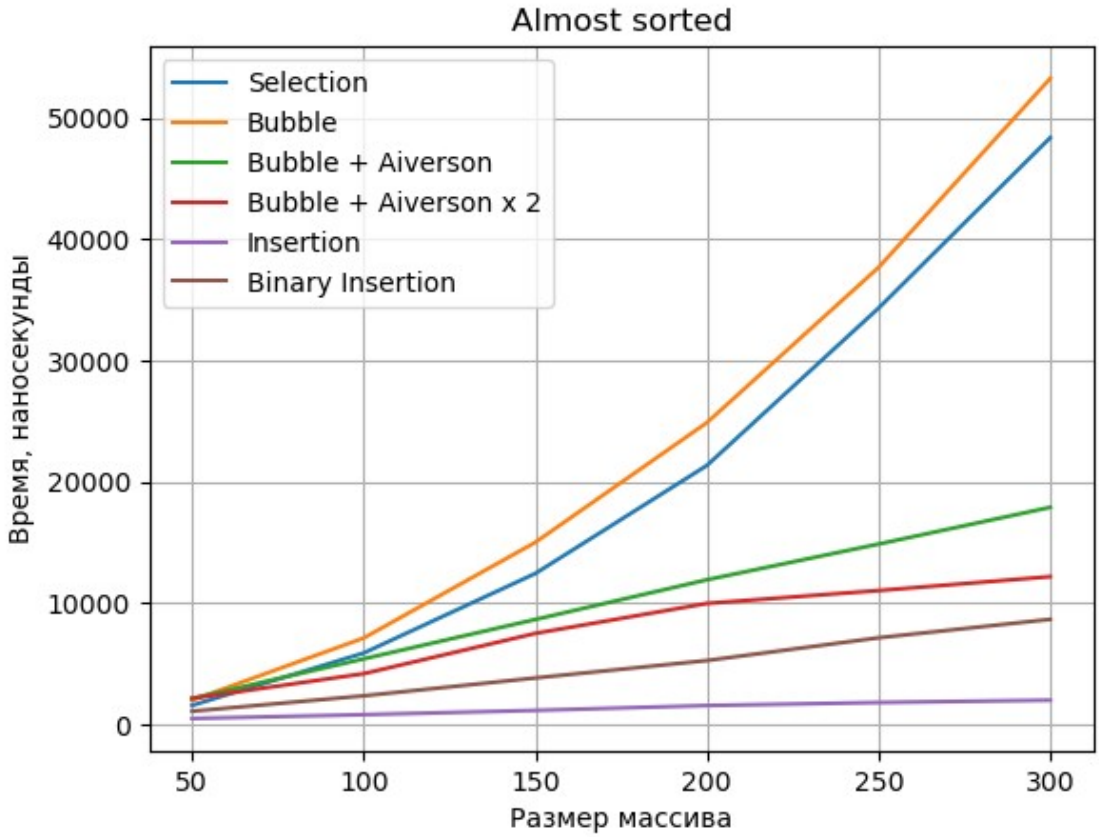
```
plt.plot(first_scale, shell_1_sort, label = 'Shell + Ciur')
plt.plot(first_scale, shell_2_sort, label = 'Shell + Shell')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Random 0 - 4100')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




Отсутствие изменений по сравнению с меньшим диапазоном чисел

```
first_time = open("C:\\Users\\pupki\\Desktop\\first_time.csv", "r")
reader1 = csv.reader(first_time)
selection_sort = []
bubble_sort = []
bubble_1_sort = []
bubble_2_sort = []
insertion_sort = []
binary_insertion_sort = []
for row in reader1:
    selection_sort.append(int(row[2]))
    bubble_sort.append(int(row[6]))
    bubble_1_sort.append(int(row[10]))
    bubble_2_sort.append(int(row[14]))
```

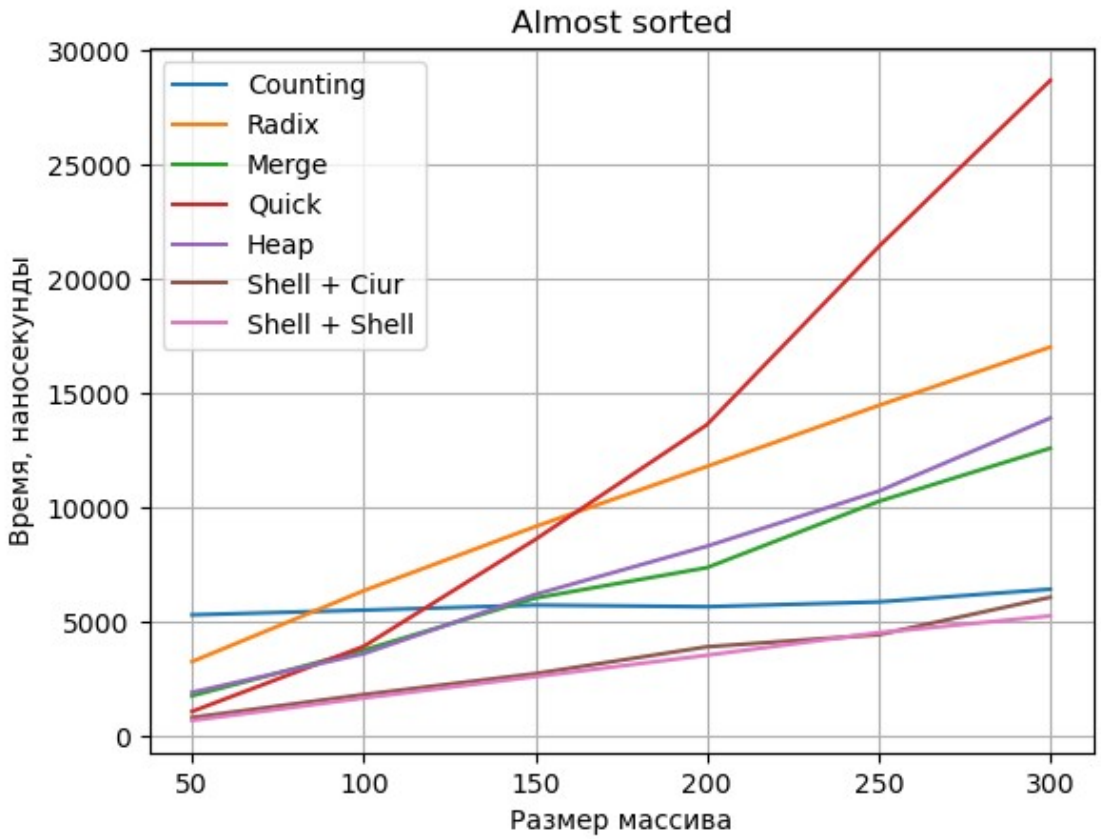
```
    insertion_sort.append(int(row[18]))
    binary_insertion_sort.append(int(row[22]))
plt.plot(first_scale, selection_sort, label = 'Selection')
plt.plot(first_scale, bubble_sort, label = 'Bubble')
plt.plot(first_scale, bubble_1_sort, label = 'Bubble + Aiverson')
plt.plot(first_scale, bubble_2_sort, label = 'Bubble + Aiverson x 2')
plt.plot(first_scale, insertion_sort, label = 'Insertion')
plt.plot(first_scale, binary_insertion_sort, label = 'Binary
Insertion')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Almost sorted')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




Выбор показывает плохое время из-за проходов почти по всему массиву, правила Айверсона заметно уменьшили время пузырька, вставка и бинарная вставка стремятся к линии

```
first_time = open("C:\\Users\\pupki\\Desktop\\first_time.csv", "r")
reader1 = csv.reader(first_time)
counting_sort = []
```

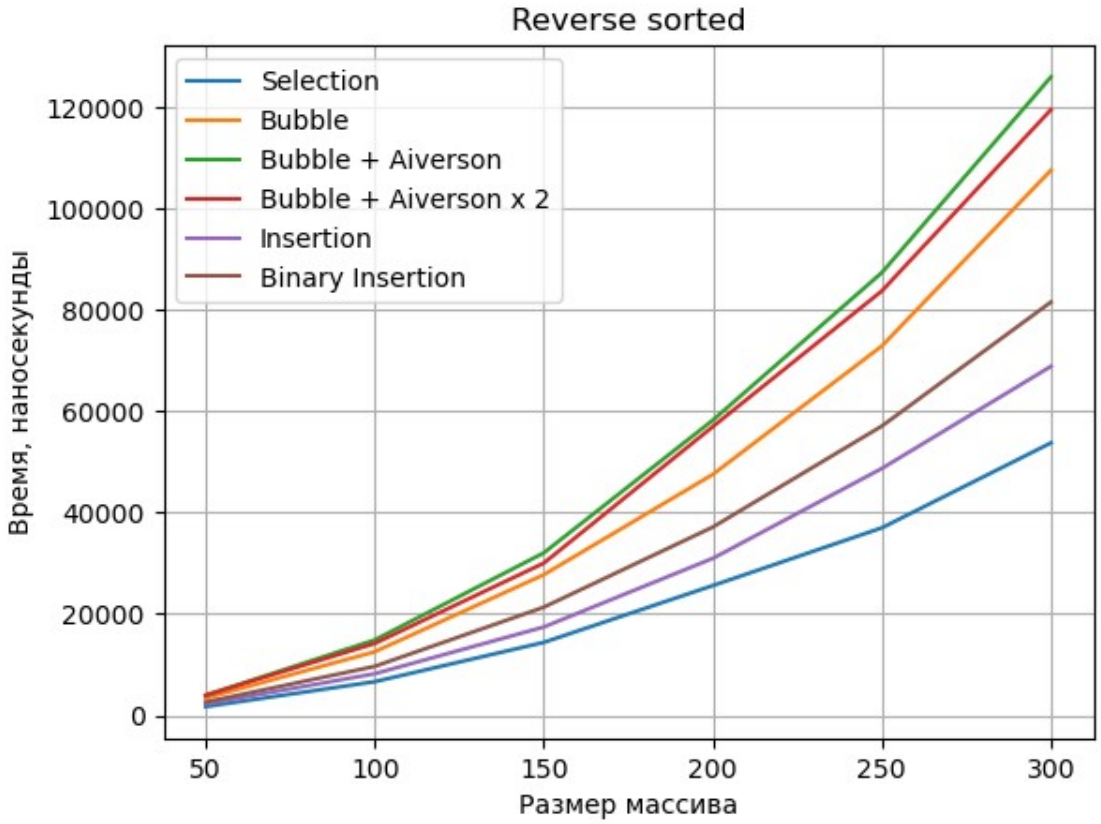
```
radix_sort = []
merge_sort = []
quick_sort = []
heap_sort = []
shell_1_sort = []
shell_2_sort = []
for row in reader1:
    counting_sort.append(int(row[26]))
    radix_sort.append(int(row[30]))
    merge_sort.append(int(row[34]))
    quick_sort.append(int(row[38]))
    heap_sort.append(int(row[42]))
    shell_1_sort.append(int(row[46]))
    shell_2_sort.append(int(row[50]))
plt.plot(first_scale, counting_sort, label = 'Counting')
plt.plot(first_scale, radix_sort, label = 'Radix')
plt.plot(first_scale, merge_sort, label = 'Merge')
plt.plot(first_scale, quick_sort, label = 'Quick')
plt.plot(first_scale, heap_sort, label = 'Heap')
plt.plot(first_scale, shell_1_sort, label = 'Shell + Ciur')
plt.plot(first_scale, shell_2_sort, label = 'Shell + Shell')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Almost sorted')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




Ухудшение быстрой сортировки с первым опорным элементом, остальные примерно линия, подсчёт также походит на константу

```
first_time = open("C:\\Users\\pupki\\Desktop\\first_time.csv", "r")
reader1 = csv.reader(first_time)
selection_sort = []
bubble_sort = []
bubble_1_sort = []
bubble_2_sort = []
insertion_sort = []
binary_insertion_sort = []
for row in reader1:
    selection_sort.append(int(row[3]))
    bubble_sort.append(int(row[7]))
    bubble_1_sort.append(int(row[11]))
    bubble_2_sort.append(int(row[15]))
    insertion_sort.append(int(row[19]))
    binary_insertion_sort.append(int(row[23]))
plt.plot(first_scale, selection_sort, label = 'Selection')
plt.plot(first_scale, bubble_sort, label = 'Bubble')
plt.plot(first_scale, bubble_1_sort, label = 'Bubble + Aiverson')
plt.plot(first_scale, bubble_2_sort, label = 'Bubble + Aiverson x 2')
plt.plot(first_scale, insertion_sort, label = 'Insertion')
plt.plot(first_scale, binary_insertion_sort, label = 'Binary
```

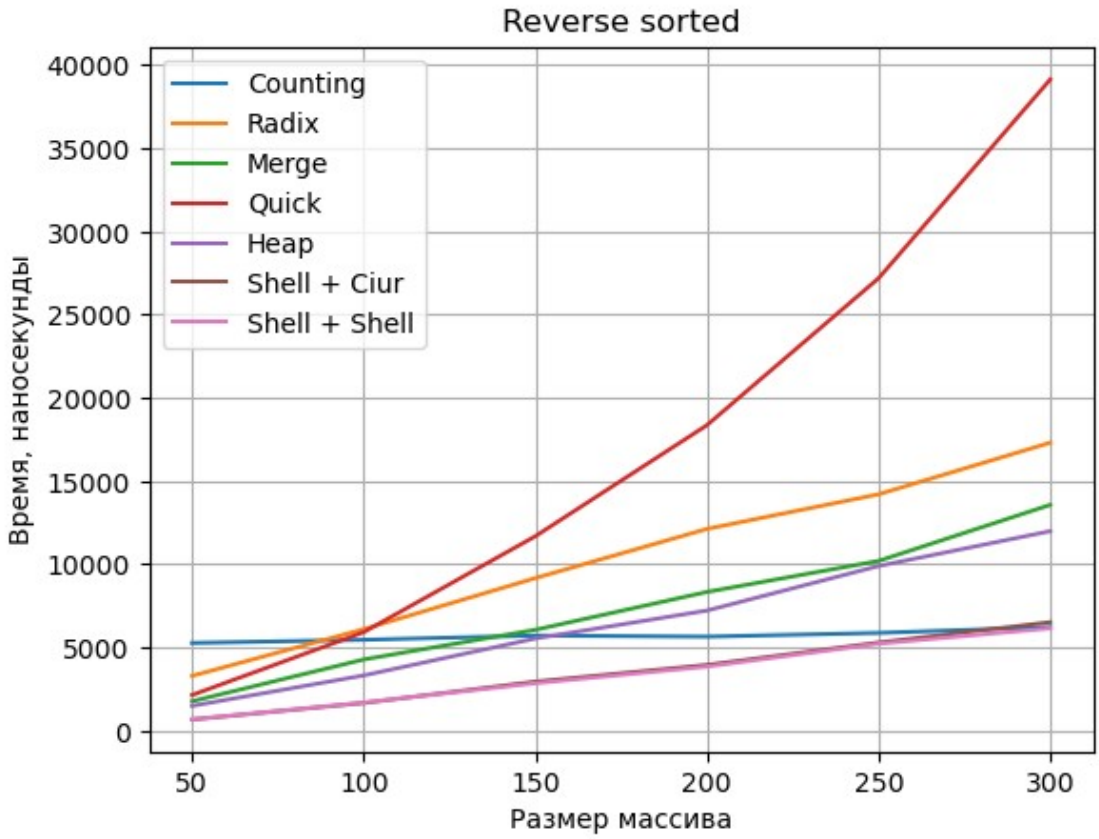
```
Insertion')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Reverse sorted')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




Отчётливо прослеживается квадратичная сложность

```
first_time = open("C:\\Users\\pupki\\Desktop\\first_time.csv", "r")
reader1 = csv.reader(first_time)
counting_sort = []
radix_sort = []
merge_sort = []
quick_sort = []
heap_sort = []
shell_1_sort = []
shell_2_sort = []
for row in reader1:
    counting_sort.append(int(row[27]))
    radix_sort.append(int(row[31]))
    merge_sort.append(int(row[35]))
    quick_sort.append(int(row[39]))
```

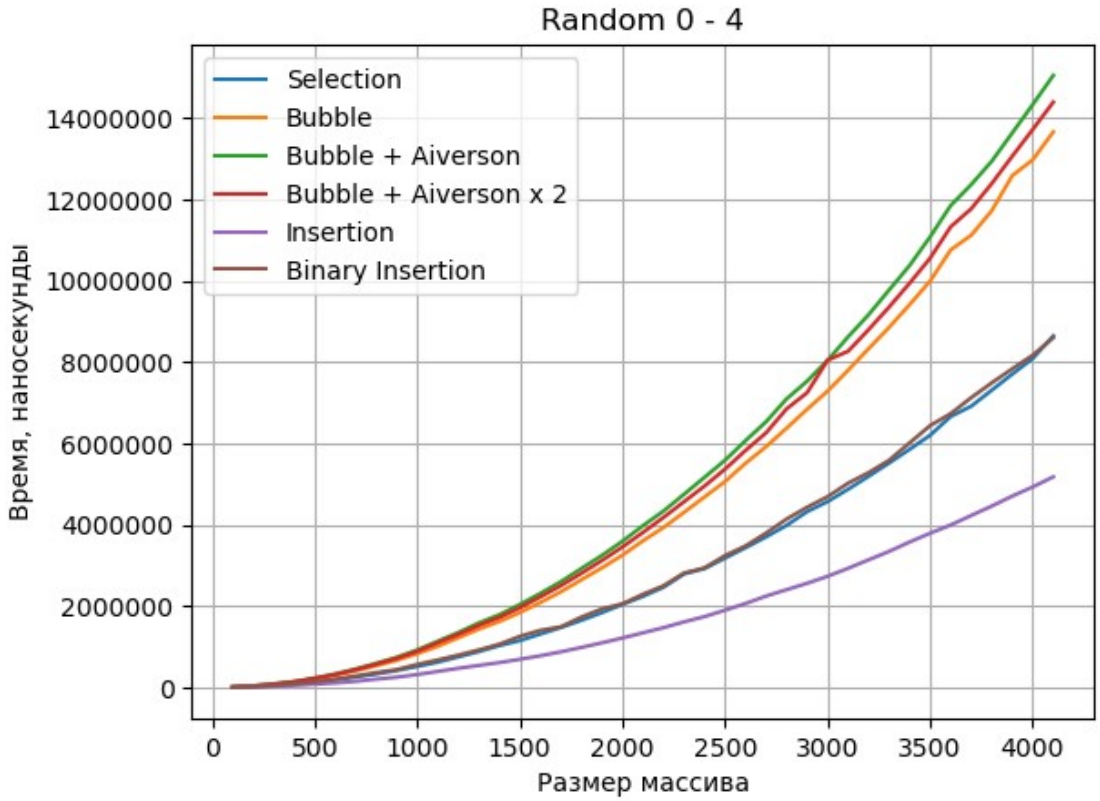
```
    heap_sort.append(int(row[43]))
    shell_1_sort.append(int(row[47]))
    shell_2_sort.append(int(row[51]))
plt.plot(first_scale, counting_sort, label = 'Counting')
plt.plot(first_scale, radix_sort, label = 'Radix')
plt.plot(first_scale, merge_sort, label = 'Merge')
plt.plot(first_scale, quick_sort, label = 'Quick')
plt.plot(first_scale, heap_sort, label = 'Heap')
plt.plot(first_scale, shell_1_sort, label = 'Shell + Ciur')
plt.plot(first_scale, shell_2_sort, label = 'Shell + Shell')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Reverse sorted')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




Квадратичная сложность у быстрой сортировки, так как она совпала с сортировкой выбора

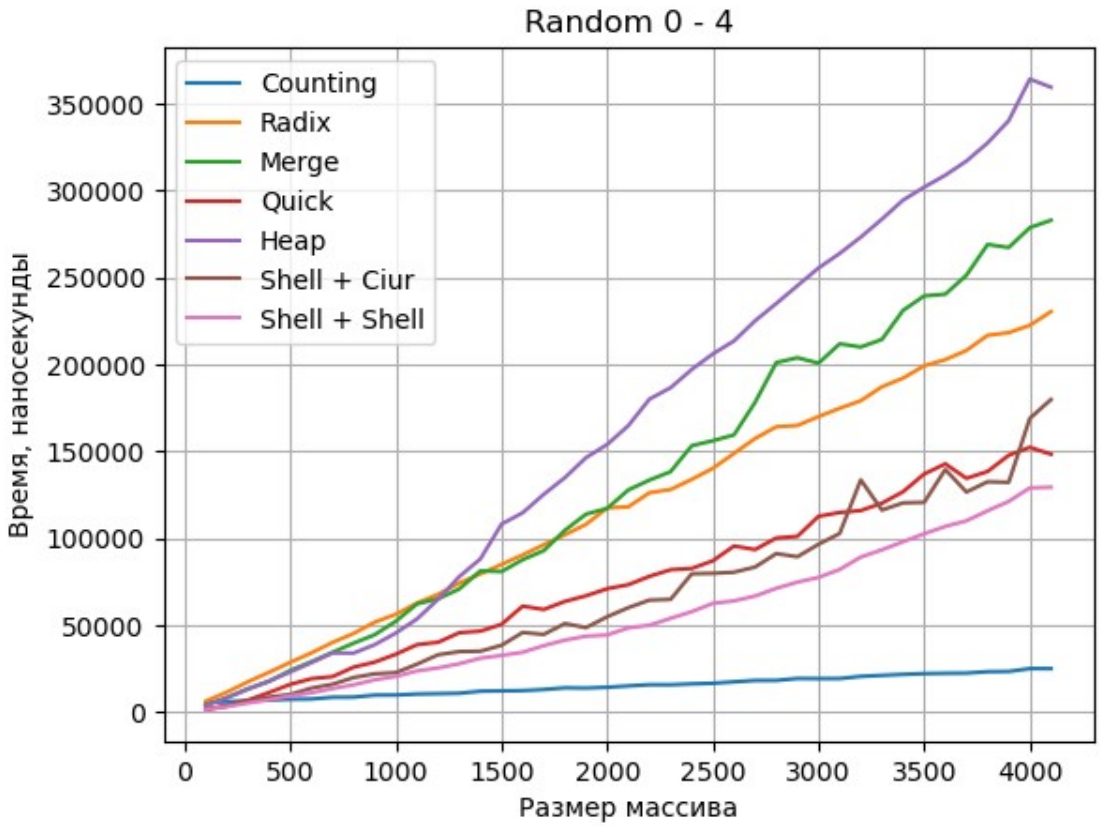
```
second_time = open("C:\\Users\\pupki\\Desktop\\second_time.csv", "r")
reader2 = csv.reader(second_time)
selection_sort = []
bubble_sort = []
```

```
bubble_1_sort = []
bubble_2_sort = []
insertion_sort = []
binary_insertion_sort = []
for row in reader2:
    selection_sort.append(int(row[0]))
    bubble_sort.append(int(row[4]))
    bubble_1_sort.append(int(row[8]))
    bubble_2_sort.append(int(row[12]))
    insertion_sort.append(int(row[16]))
    binary_insertion_sort.append(int(row[20]))
plt.plot(second_scale, selection_sort, label = 'Selection')
plt.plot(second_scale, bubble_sort, label = 'Bubble')
plt.plot(second_scale, bubble_1_sort, label = 'Bubble + Aiverson')
plt.plot(second_scale, bubble_2_sort, label = 'Bubble + Aiverson x 2')
plt.plot(second_scale, insertion_sort, label = 'Insertion')
plt.plot(second_scale, binary_insertion_sort, label = 'Binary
Insertion')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Random 0 - 4')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```



Отчётливый квадрат, вставка показывает хорошую константу, бинарного улучшения нет(

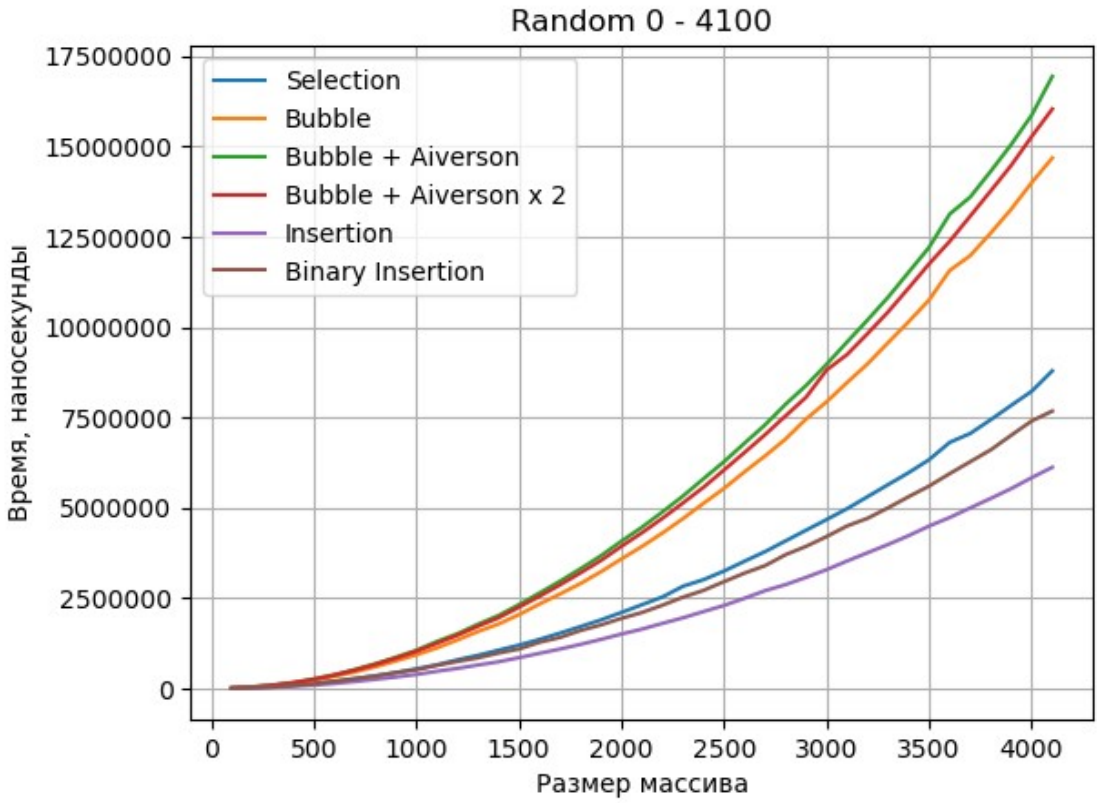
```
second_time = open("C:\\Users\\pupki\\Desktop\\second_time.csv", "r")
reader2 = csv.reader(second_time)
counting_sort = []
radix_sort = []
merge_sort = []
quick_sort = []
heap_sort = []
shell_1_sort = []
shell_2_sort = []
for row in reader2:
    counting_sort.append(int(row[24]))
    radix_sort.append(int(row[28]))
    merge_sort.append(int(row[32]))
    quick_sort.append(int(row[36]))
    heap_sort.append(int(row[40]))
    shell_1_sort.append(int(row[44]))
    shell_2_sort.append(int(row[48]))
plt.plot(second_scale, counting_sort, label = 'Counting')
plt.plot(second_scale, radix_sort, label = 'Radix')
plt.plot(second_scale, merge_sort, label = 'Merge')
plt.plot(second_scale, quick_sort, label = 'Quick')
plt.plot(second_scale, heap_sort, label = 'Heap')
plt.plot(second_scale, shell_1_sort, label = 'Shell + Ciur')
plt.plot(second_scale, shell_2_sort, label = 'Shell + Shell')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Random 0 - 4')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




Отличное время для подсчёта из-за её устройства, последовательность Шелла все ещё лучше Циура, всё стремится к линии

```
second_time = open("C:\\Users\\pupki\\Desktop\\second_time.csv", "r")
reader2 = csv.reader(second_time)
selection_sort = []
bubble_sort = []
bubble_1_sort = []
bubble_2_sort = []
insertion_sort = []
binary_insertion_sort = []
for row in reader2:
    selection_sort.append(int(row[1]))
    bubble_sort.append(int(row[5]))
    bubble_1_sort.append(int(row[9]))
    bubble_2_sort.append(int(row[13]))
    insertion_sort.append(int(row[17]))
    binary_insertion_sort.append(int(row[21]))
plt.plot(second_scale, selection_sort, label = 'Selection')
plt.plot(second_scale, bubble_sort, label = 'Bubble')
plt.plot(second_scale, bubble_1_sort, label = 'Bubble + Aiverson')
plt.plot(second_scale, bubble_2_sort, label = 'Bubble + Aiverson x 2')
plt.plot(second_scale, insertion_sort, label = 'Insertion')
plt.plot(second_scale, binary_insertion_sort, label = 'Binary
Insertion')
```

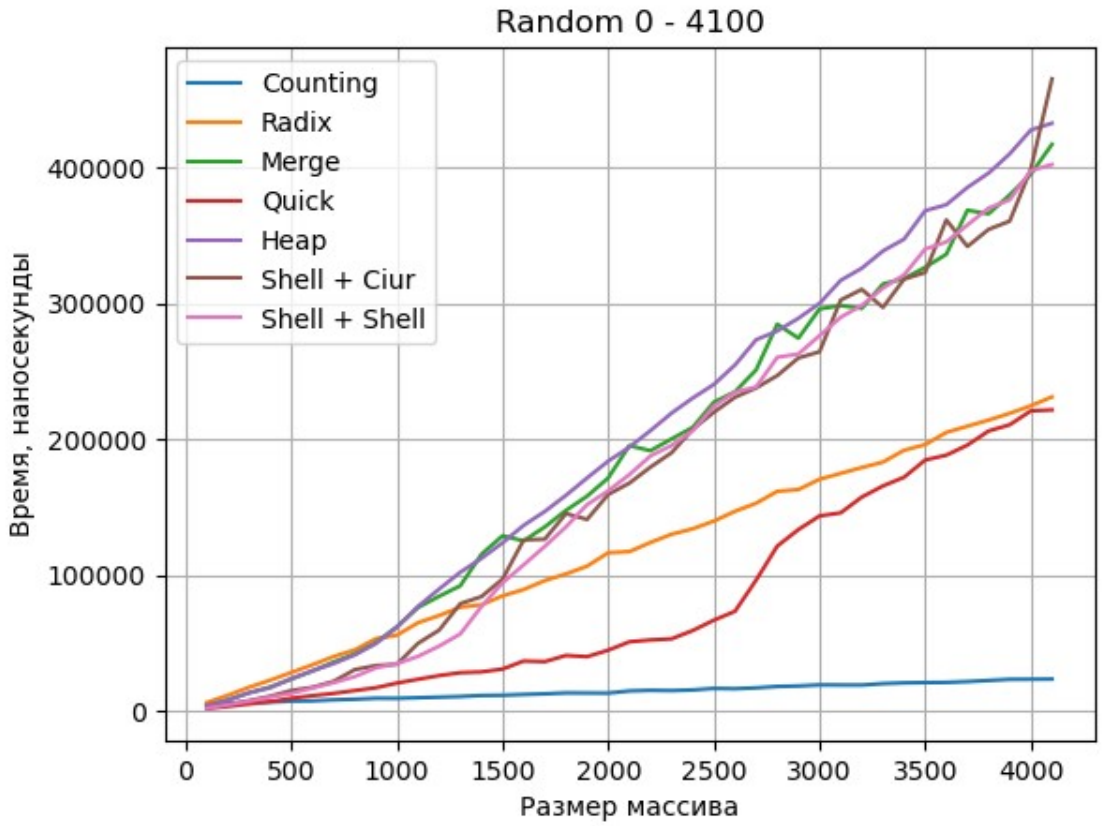
```
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Random 0 - 4100')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




Минимальные изменения по сравнению с меньшим диапазоном

```
second_time = open("C:\\Users\\pupki\\Desktop\\second_time.csv", "r")
reader2 = csv.reader(second_time)
counting_sort = []
radix_sort = []
merge_sort = []
quick_sort = []
heap_sort = []
shell_1_sort = []
shell_2_sort = []
for row in reader2:
    counting_sort.append(int(row[25]))
    radix_sort.append(int(row[29]))
    merge_sort.append(int(row[33]))
    quick_sort.append(int(row[37]))
    heap_sort.append(int(row[41]))
    shell_1_sort.append(int(row[45]))
```

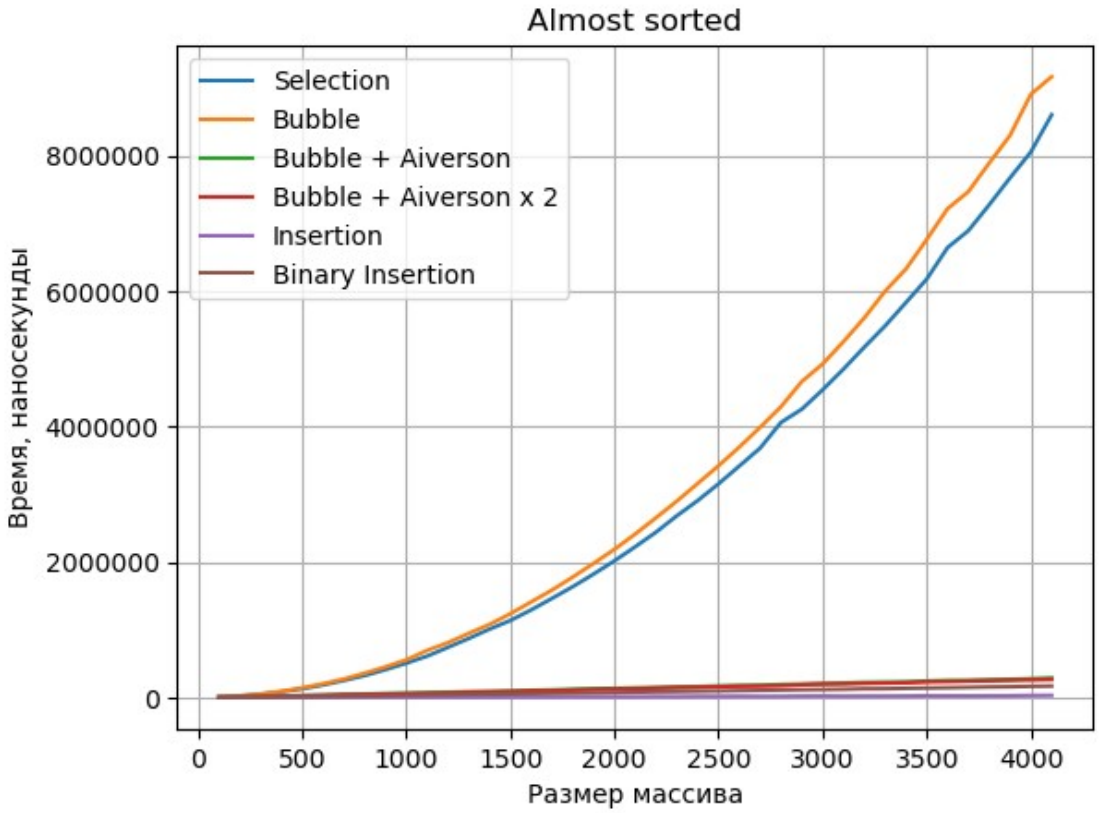
```
    shell_2_sort.append(int(row[49]))
plt.plot(second_scale, counting_sort, label = 'Counting')
plt.plot(second_scale, radix_sort, label = 'Radix')
plt.plot(second_scale, merge_sort, label = 'Merge')
plt.plot(second_scale, quick_sort, label = 'Quick')
plt.plot(second_scale, heap_sort, label = 'Heap')
plt.plot(second_scale, shell_1_sort, label = 'Shell + Ciur')
plt.plot(second_scale, shell_2_sort, label = 'Shell + Shell')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Random 0 - 4100')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




Большинство сортировок показывают почти одинаковое время

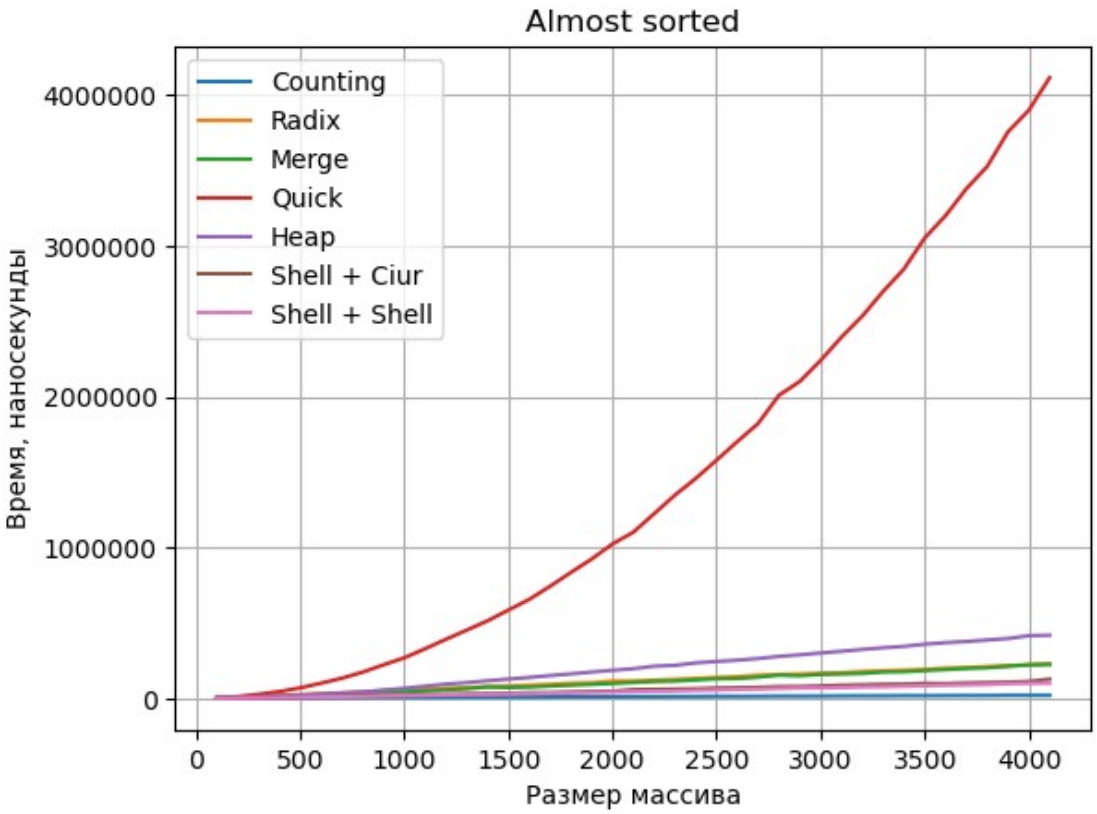
```
second_time = open("C:\\Users\\pupki\\Desktop\\second_time.csv", "r")
reader2 = csv.reader(second_time)
selection_sort = []
bubble_sort = []
bubble_1_sort = []
bubble_2_sort = []
insertion_sort = []
```

```
binary_insertion_sort = []
for row in reader2:
    selection_sort.append(int(row[2]))
    bubble_sort.append(int(row[6]))
    bubble_1_sort.append(int(row[10]))
    bubble_2_sort.append(int(row[14]))
    insertion_sort.append(int(row[18]))
    binary_insertion_sort.append(int(row[22]))
plt.plot(second_scale, selection_sort, label = 'Selection')
plt.plot(second_scale, bubble_sort, label = 'Bubble')
plt.plot(second_scale, bubble_1_sort, label = 'Bubble + Aiverson')
plt.plot(second_scale, bubble_2_sort, label = 'Bubble + Aiverson x 2')
plt.plot(second_scale, insertion_sort, label = 'Insertion')
plt.plot(second_scale, binary_insertion_sort, label = 'Binary
Insertion')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Almost sorted')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




Почти все сортировки показали отличное время, пузырёк без дополнительных условий и выбор - квадрат

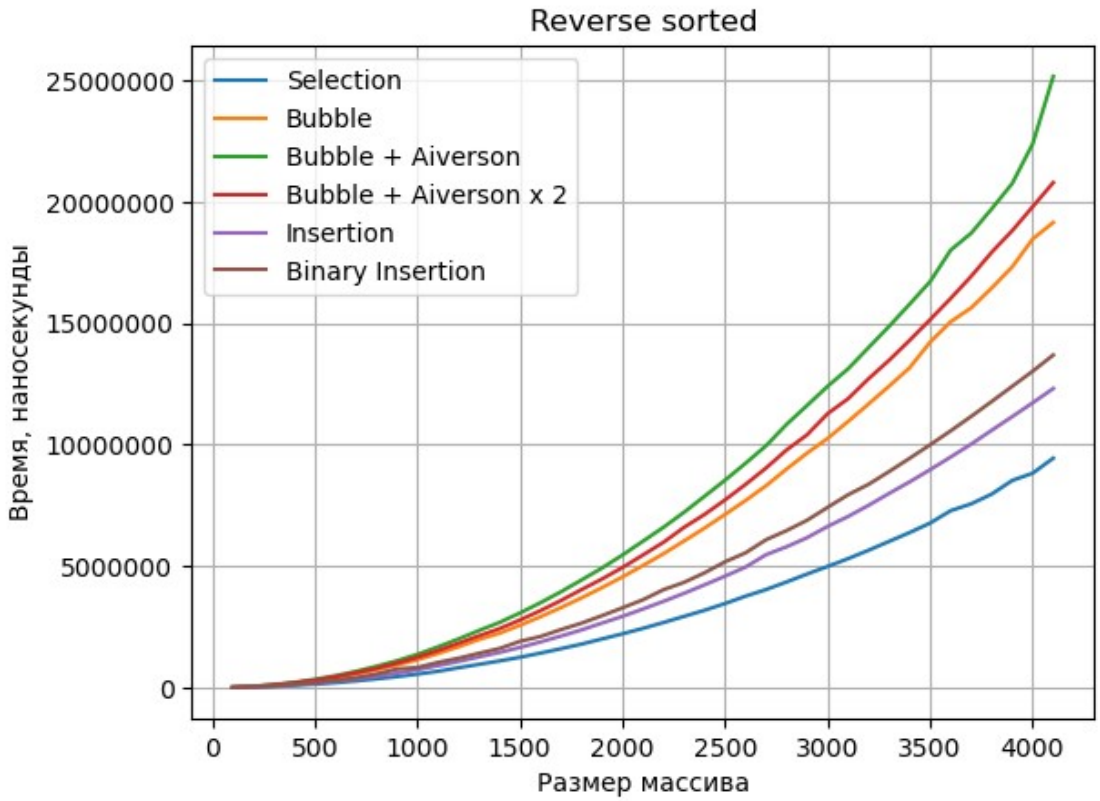
```
second_time = open("C:\\Users\\pupki\\Desktop\\second_time.csv", "r")
reader2 = csv.reader(second_time)
counting_sort = []
radix_sort = []
merge_sort = []
quick_sort = []
heap_sort = []
shell_1_sort = []
shell_2_sort = []
for row in reader2:
    counting_sort.append(int(row[26]))
    radix_sort.append(int(row[30]))
    merge_sort.append(int(row[34]))
    quick_sort.append(int(row[38]))
    heap_sort.append(int(row[42]))
    shell_1_sort.append(int(row[46]))
    shell_2_sort.append(int(row[50]))
plt.plot(second_scale, counting_sort, label = 'Counting')
plt.plot(second_scale, radix_sort, label = 'Radix')
plt.plot(second_scale, merge_sort, label = 'Merge')
plt.plot(second_scale, quick_sort, label = 'Quick')
plt.plot(second_scale, heap_sort, label = 'Heap')
plt.plot(second_scale, shell_1_sort, label = 'Shell + Ciur')
plt.plot(second_scale, shell_2_sort, label = 'Shell + Shell')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Almost sorted')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




Быстрая сортировка стремится к квадрату

```
second_time = open("C:\\Users\\pupki\\Desktop\\second_time.csv", "r")
reader2 = csv.reader(second_time)
selection_sort = []
bubble_sort = []
bubble_1_sort = []
bubble_2_sort = []
insertion_sort = []
binary_insertion_sort = []
for row in reader2:
    selection_sort.append(int(row[3]))
    bubble_sort.append(int(row[7]))
    bubble_1_sort.append(int(row[11]))
    bubble_2_sort.append(int(row[15]))
    insertion_sort.append(int(row[19]))
    binary_insertion_sort.append(int(row[23]))
plt.plot(second_scale, selection_sort, label = 'Selection')
plt.plot(second_scale, bubble_sort, label = 'Bubble')
plt.plot(second_scale, bubble_1_sort, label = 'Bubble + Aiverson')
plt.plot(second_scale, bubble_2_sort, label = 'Bubble + Aiverson x 2')
plt.plot(second_scale, insertion_sort, label = 'Insertion')
plt.plot(second_scale, binary_insertion_sort, label = 'Binary
Insertion')
plt.grid(True)
```

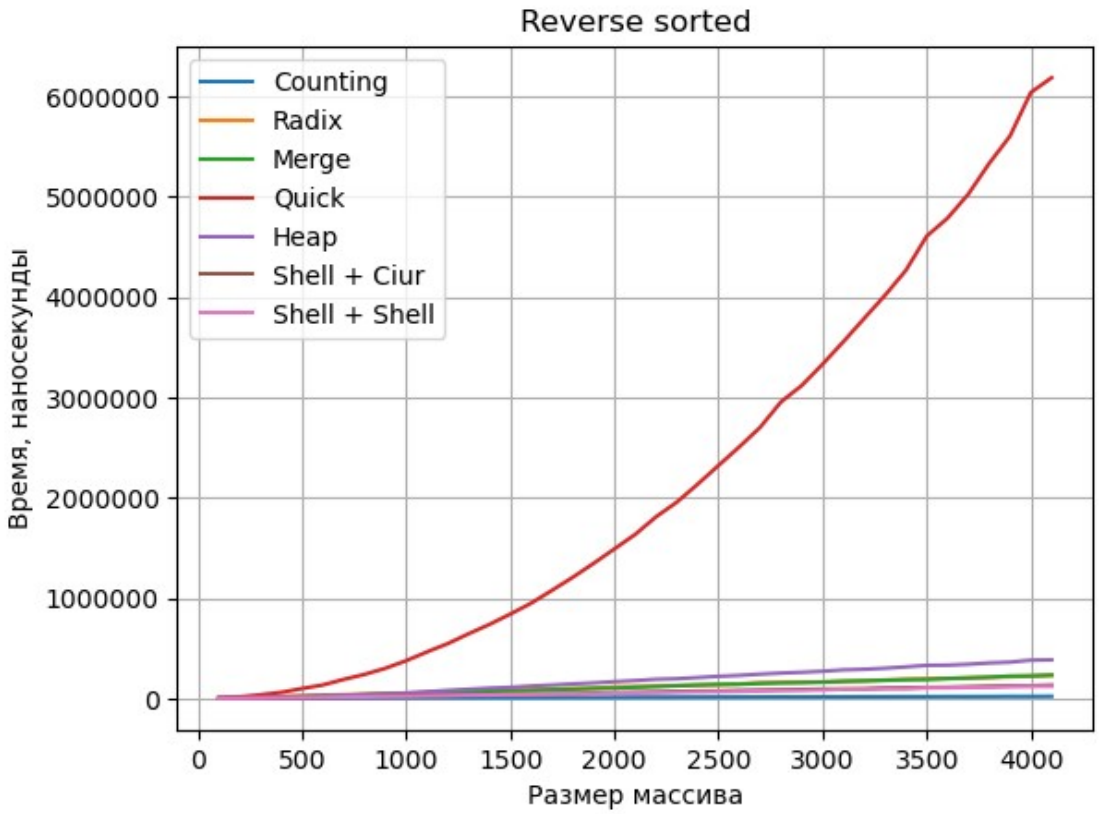
```
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Reverse sorted')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




Правила Айверсона только ухудшают время

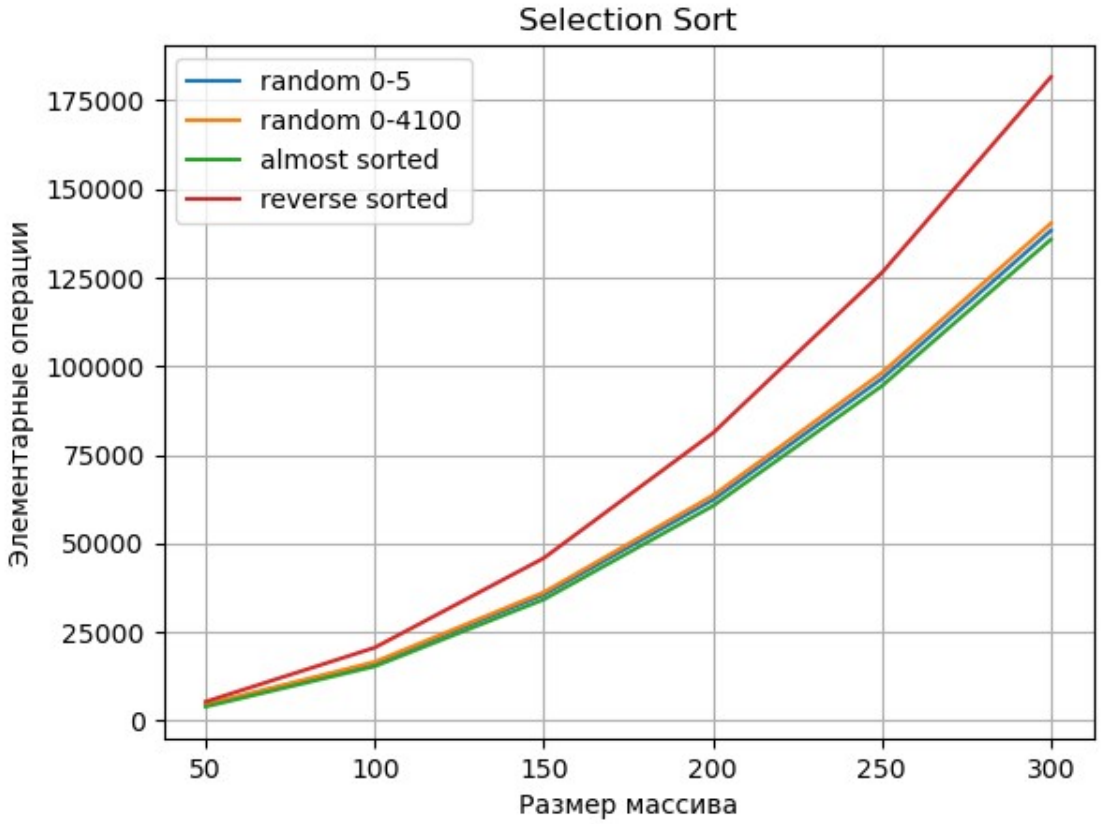
```
second_time = open("C:\\Users\\pupki\\Desktop\\second_time.csv", "r")
reader2 = csv.reader(second_time)
counting_sort = []
radix_sort = []
merge_sort = []
quick_sort = []
heap_sort = []
shell_1_sort = []
shell_2_sort = []
for row in reader2:
    counting_sort.append(int(row[27]))
    radix_sort.append(int(row[31]))
    merge_sort.append(int(row[35]))
    quick_sort.append(int(row[39]))
    heap_sort.append(int(row[43]))
    shell_1_sort.append(int(row[47]))
    shell_2_sort.append(int(row[51]))
```

```
plt.plot(second_scale, counting_sort, label = 'Counting')
plt.plot(second_scale, radix_sort, label = 'Radix')
plt.plot(second_scale, merge_sort, label = 'Merge')
plt.plot(second_scale, quick_sort, label = 'Quick')
plt.plot(second_scale, heap_sort, label = 'Heap')
plt.plot(second_scale, shell_1_sort, label = 'Shell + Ciur')
plt.plot(second_scale, shell_2_sort, label = 'Shell + Shell')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Reverse sorted')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```



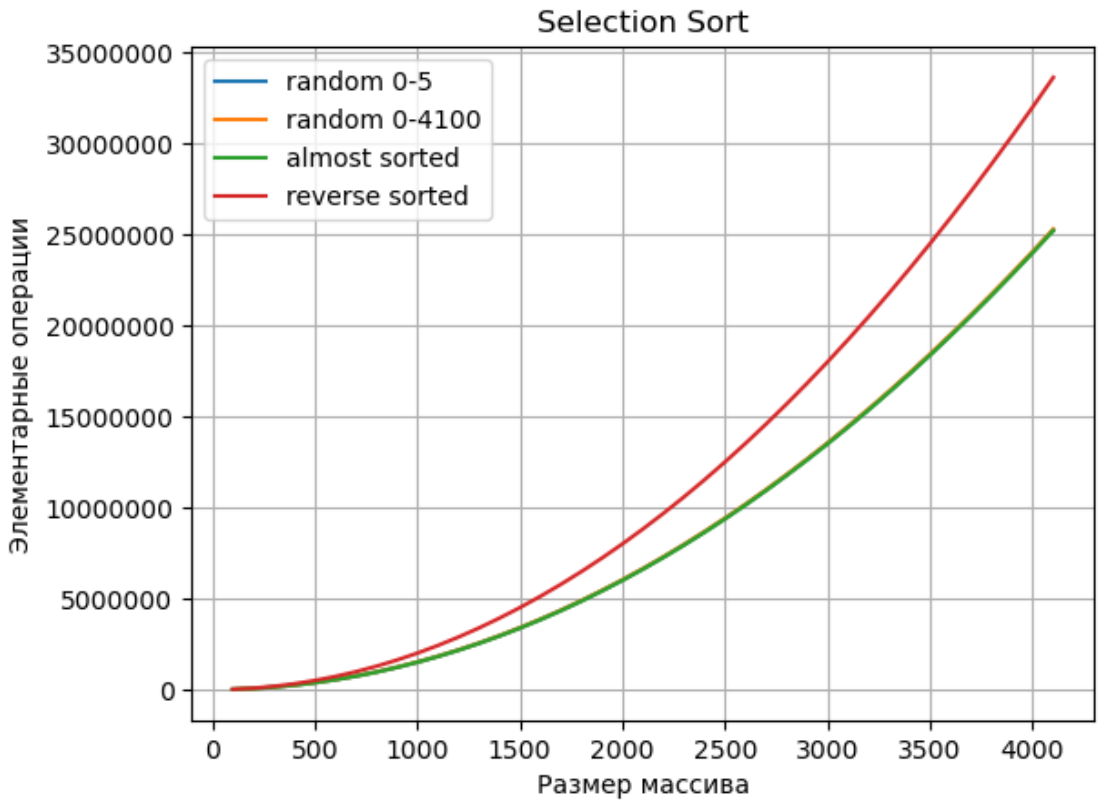

```
first_operations = open("C:\\Users\\pupki\\Desktop\\
first_operations.csv", "r")
reader1 = csv.reader(first_operations)
selection_sort_r5 = []
selection_sort_r4000 = []
selection_sort_as = []
selection_sort_rs = []
for row in reader1:
    selection_sort_r5.append(int(row[0]))
    selection_sort_r4000.append(int(row[1]))
```

```
    selection_sort_as.append(int(row[2]))
    selection_sort_rs.append(int(row[3]))
plt.plot(first_scale, selection_sort_r5, label = 'random 0-5')
plt.plot(first_scale, selection_sort_r4000, label = 'random 0-4100')
plt.plot(first_scale, selection_sort_as, label = 'almost sorted')
plt.plot(first_scale, selection_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Selection Sort')
plt.legend(loc = 'best')
plt.show()
```



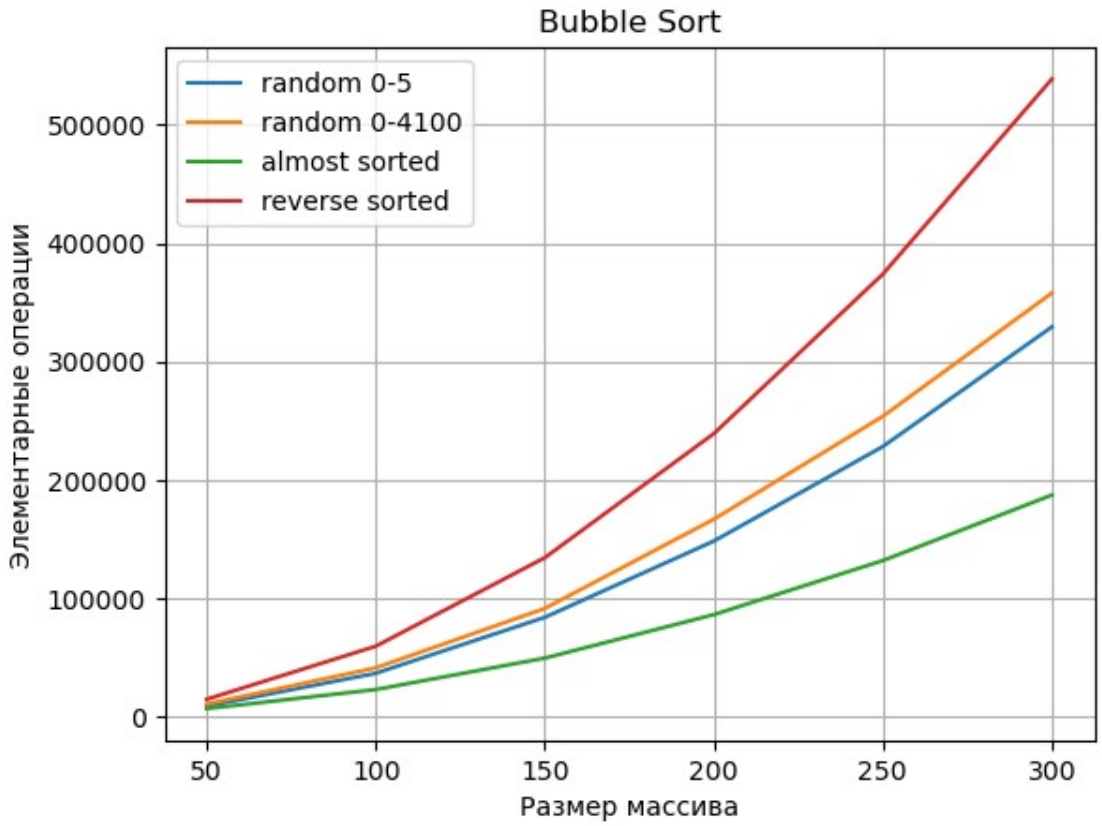

```
second_operations = open("C:\\Users\\pupki\\Desktop\\
second_operations.csv", "r")
reader2 = csv.reader(second_operations)
selection_sort_r5 = []
selection_sort_r4000 = []
selection_sort_as = []
selection_sort_rs = []
for row in reader2:
    selection_sort_r5.append(int(row[0]))
    selection_sort_r4000.append(int(row[1]))
    selection_sort_as.append(int(row[2]))
    selection_sort_rs.append(int(row[3]))
```

```
plt.plot(second_scale, selection_sort_r5, label = 'random 0-5')
plt.plot(second_scale, selection_sort_r4000, label = 'random 0-4100')
plt.plot(second_scale, selection_sort_as, label = 'almost sorted')
plt.plot(second_scale, selection_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Selection Sort')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```



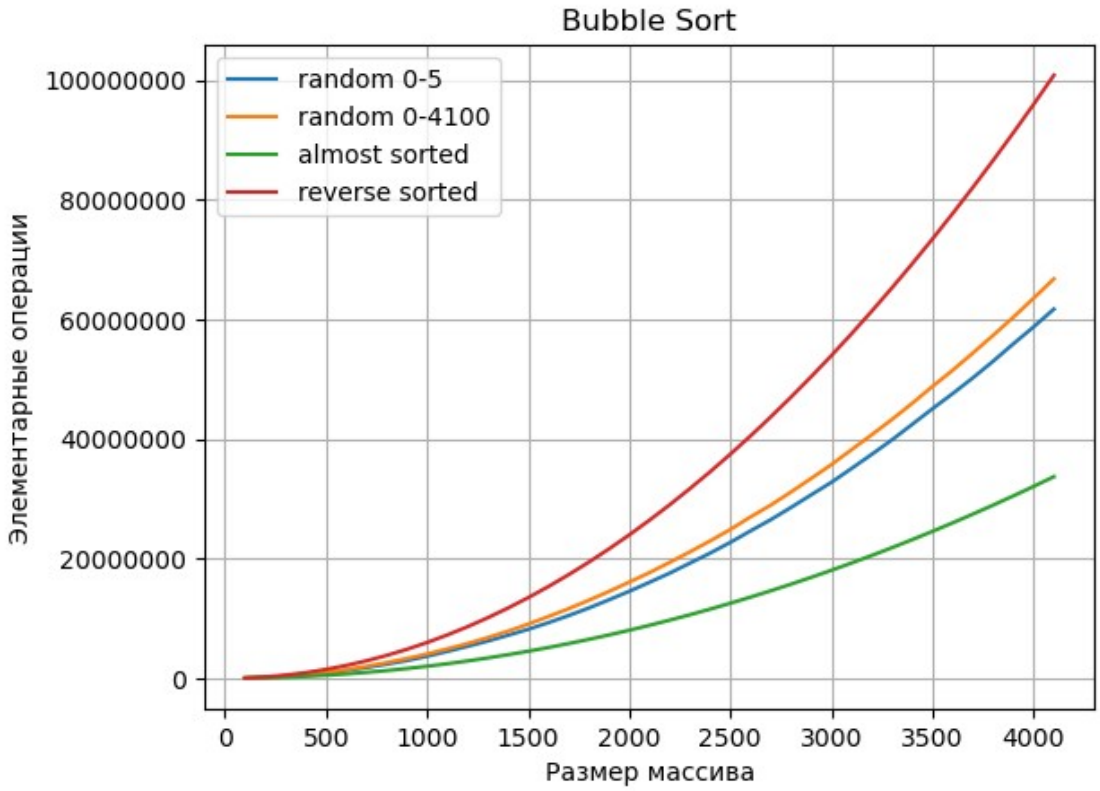

```
first_operations = open("C:\\Users\\pupki\\Desktop\\
first_operations.csv", "r")
reader1 = csv.reader(first_operations)
bubble_sort_r5 = []
bubble_sort_r4000 = []
bubble_sort_as = []
bubble_sort_rs = []
for row in reader1:
    bubble_sort_r5.append(int(row[4]))
    bubble_sort_r4000.append(int(row[5]))
    bubble_sort_as.append(int(row[6]))
    bubble_sort_rs.append(int(row[7]))
plt.plot(first_scale, bubble_sort_r5, label = 'random 0-5')
plt.plot(first_scale, bubble_sort_r4000, label = 'random 0-4100')
```

```
plt.plot(first_scale, bubble_sort_as, label = 'almost sorted')
plt.plot(first_scale, bubble_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Bubble Sort')
plt.legend(loc = 'best')
plt.show()
```



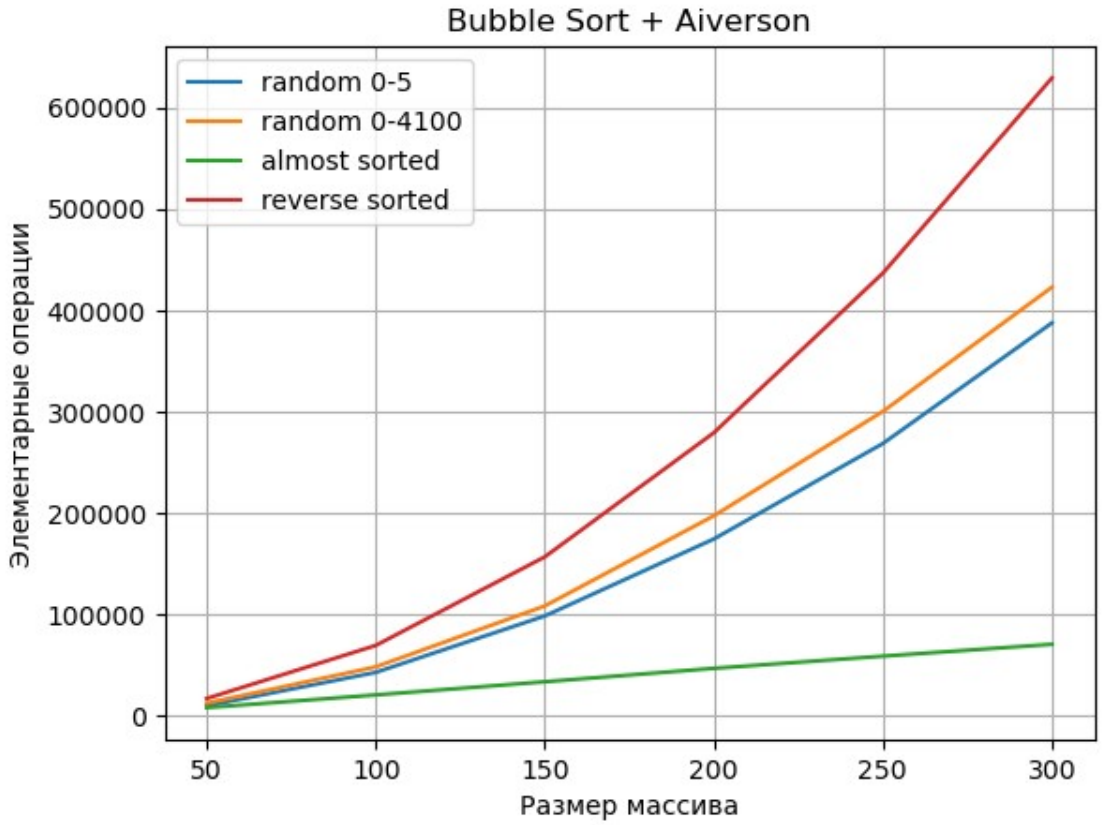

```
second_operations = open("C:\\Users\\pupki\\Desktop\\
second_operations.csv", "r")
reader2 = csv.reader(second_operations)
bubble_sort_r5 = []
bubble_sort_r4000 = []
bubble_sort_as = []
bubble_sort_rs = []
for row in reader2:
    bubble_sort_r5.append(int(row[4]))
    bubble_sort_r4000.append(int(row[5]))
    bubble_sort_as.append(int(row[6]))
    bubble_sort_rs.append(int(row[7]))
plt.plot(second_scale, bubble_sort_r5, label = 'random 0-5')
plt.plot(second_scale, bubble_sort_r4000, label = 'random 0-4100')
plt.plot(second_scale, bubble_sort_as, label = 'almost sorted')
plt.plot(second_scale, bubble_sort_rs, label = 'reverse sorted')
```

```
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Bubble Sort')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```



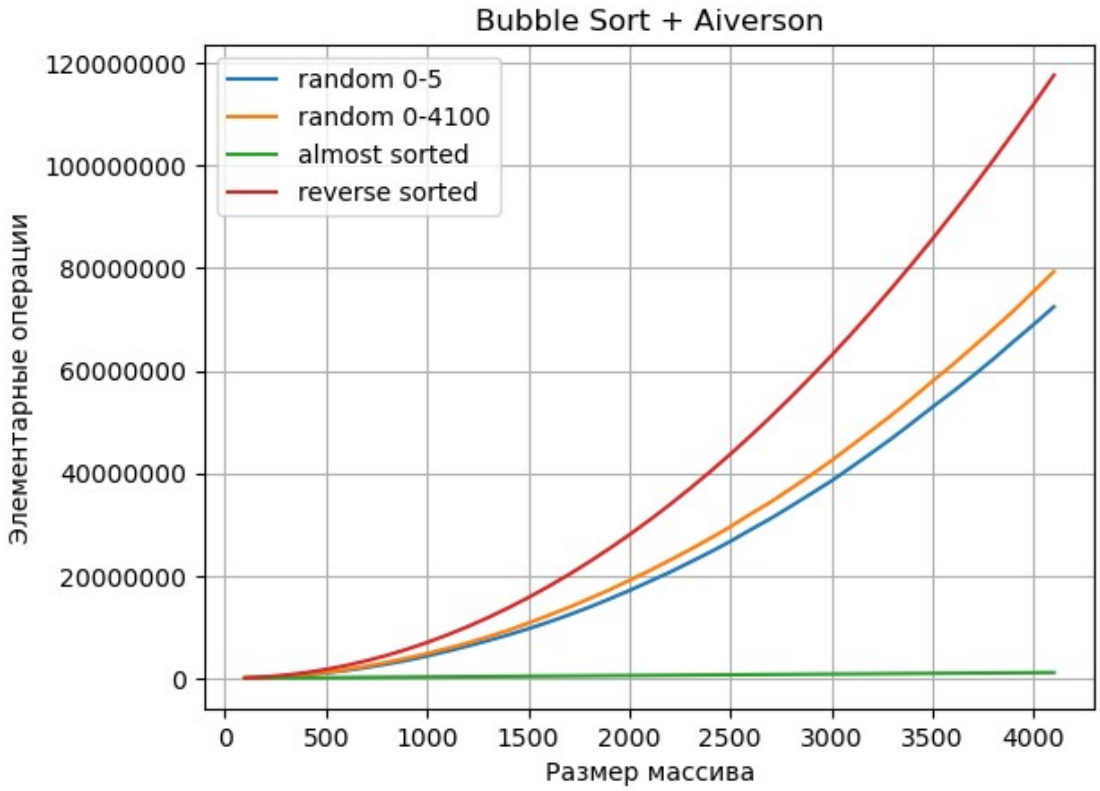

```
first_operations = open("C:\\Users\\pupki\\Desktop\\
first_operations.csv", "r")
reader1 = csv.reader(first_operations)
bubble_1_sort_r5 = []
bubble_1_sort_r4000 = []
bubble_1_sort_as = []
bubble_1_sort_rs = []
for row in reader1:
    bubble_1_sort_r5.append(int(row[8]))
    bubble_1_sort_r4000.append(int(row[9]))
    bubble_1_sort_as.append(int(row[10]))
    bubble_1_sort_rs.append(int(row[11]))
plt.plot(first_scale, bubble_1_sort_r5, label = 'random 0-5')
plt.plot(first_scale, bubble_1_sort_r4000, label = 'random 0-4100')
plt.plot(first_scale, bubble_1_sort_as, label = 'almost sorted')
plt.plot(first_scale, bubble_1_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
```

```
plt.ylabel('Элементарные операции')
plt.title('Bubble Sort + Aiverson')
plt.legend(loc = 'best')
plt.show()
```

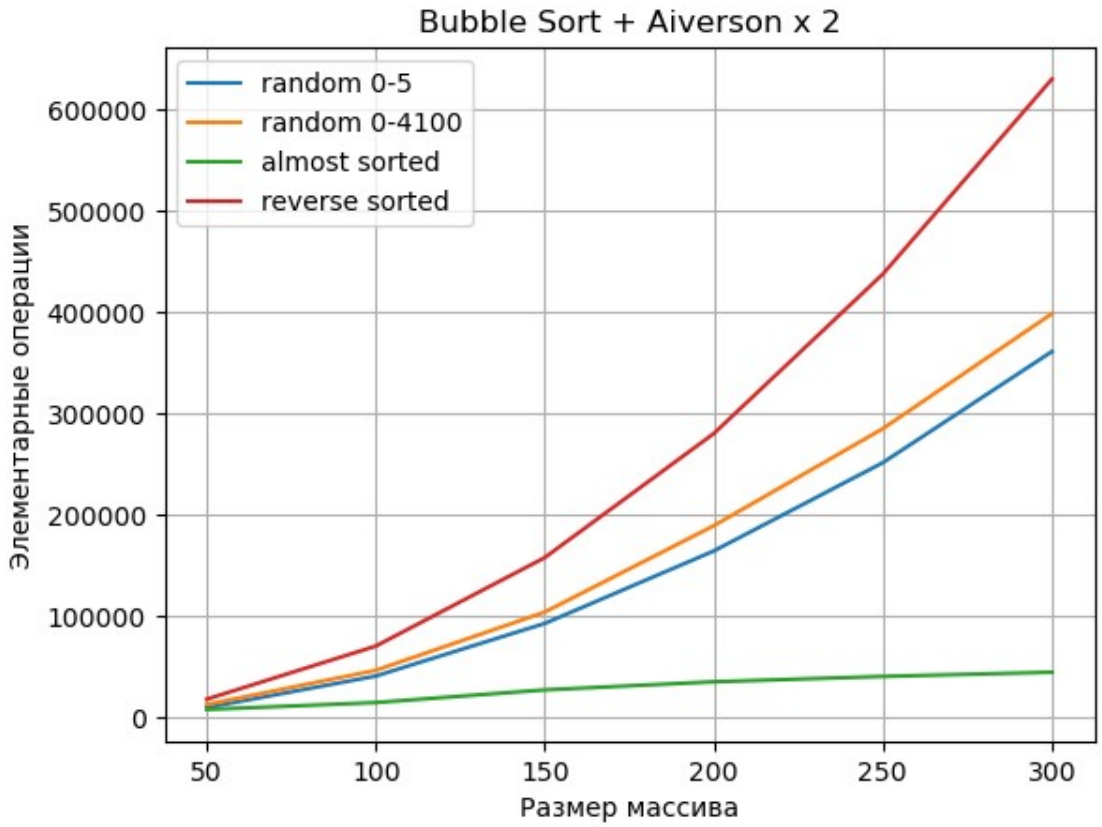



```
second_operations = open("C:\\Users\\pupki\\Desktop\\
second_operations.csv", "r")
reader2 = csv.reader(second_operations)
bubble_1_sort_r5 = []
bubble_1_sort_r4000 = []
bubble_1_sort_as = []
bubble_1_sort_rs = []
for row in reader2:
    bubble_1_sort_r5.append(int(row[8]))
    bubble_1_sort_r4000.append(int(row[9]))
    bubble_1_sort_as.append(int(row[10]))
    bubble_1_sort_rs.append(int(row[11]))
plt.plot(second_scale, bubble_1_sort_r5, label = 'random 0-5')
plt.plot(second_scale, bubble_1_sort_r4000, label = 'random 0-4100')
plt.plot(second_scale, bubble_1_sort_as, label = 'almost sorted')
plt.plot(second_scale, bubble_1_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Bubble Sort + Aiverson')
```

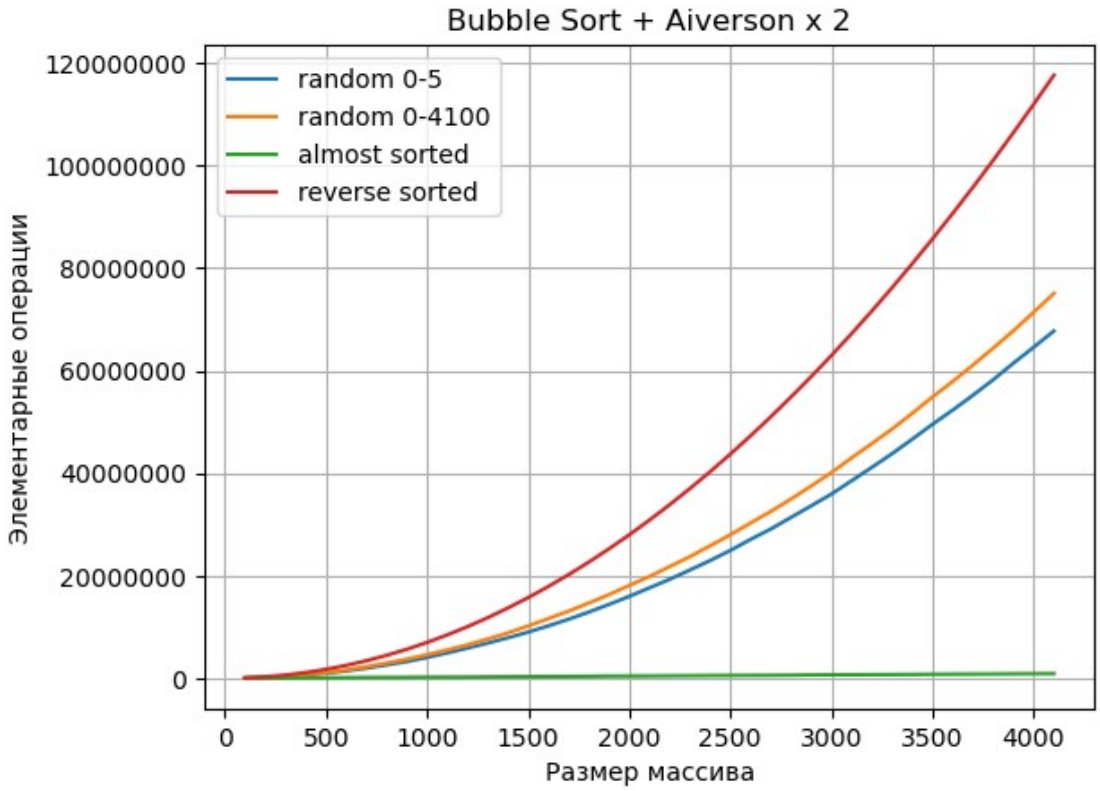
```
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```



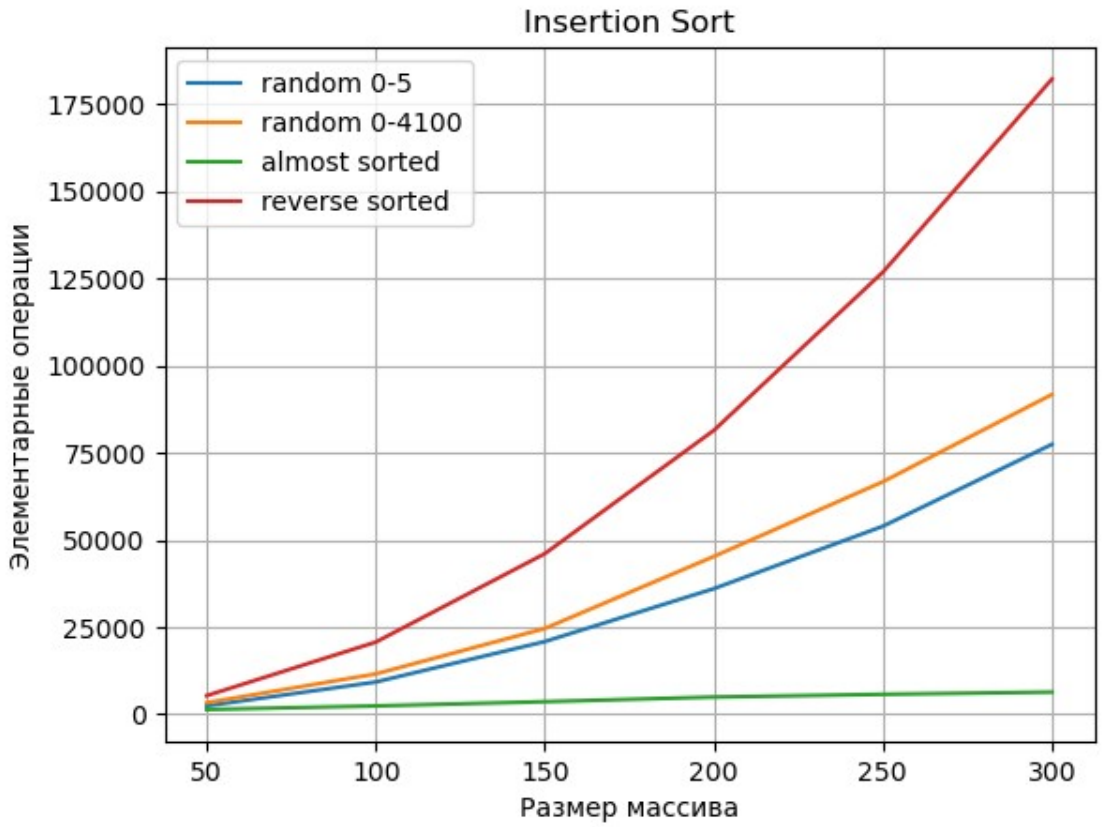

```
first_operations = open("C:\\Users\\pupki\\Desktop\\
first_operations.csv", "r")
reader1 = csv.reader(first_operations)
bubble_2_sort_r5 = []
bubble_2_sort_r4000 = []
bubble_2_sort_as = []
bubble_2_sort_rs = []
for row in reader1:
    bubble_2_sort_r5.append(int(row[12]))
    bubble_2_sort_r4000.append(int(row[13]))
    bubble_2_sort_as.append(int(row[14]))
    bubble_2_sort_rs.append(int(row[15]))
plt.plot(first_scale, bubble_2_sort_r5, label = 'random 0-5')
plt.plot(first_scale, bubble_2_sort_r4000, label = 'random 0-4100')
plt.plot(first_scale, bubble_2_sort_as, label = 'almost sorted')
plt.plot(first_scale, bubble_2_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Bubble Sort + Aiverson x 2')
plt.legend(loc = 'best')
plt.show()
```



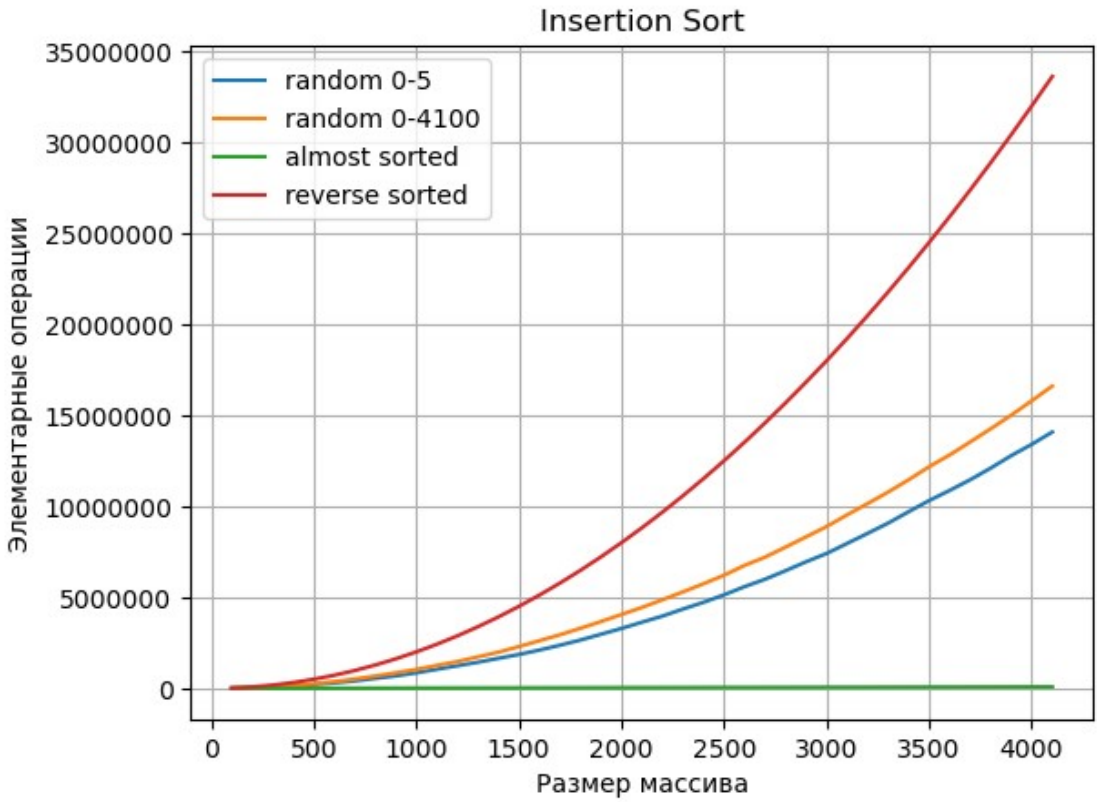

```
second_operations = open("C:\\Users\\pupki\\Desktop\\
second_operations.csv", "r")
reader2 = csv.reader(second_operations)
bubble_1_sort_r5 = []
bubble_1_sort_r4000 = []
bubble_1_sort_as = []
bubble_1_sort_rs = []
for row in reader2:
    bubble_1_sort_r5.append(int(row[12]))
    bubble_1_sort_r4000.append(int(row[13]))
    bubble_1_sort_as.append(int(row[14]))
    bubble_1_sort_rs.append(int(row[15]))
plt.plot(second_scale, bubble_1_sort_r5, label = 'random 0-5')
plt.plot(second_scale, bubble_1_sort_r4000, label = 'random 0-4100')
plt.plot(second_scale, bubble_1_sort_as, label = 'almost sorted')
plt.plot(second_scale, bubble_1_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Bubble Sort + Aiverson x 2')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




```
first_operations = open("C:\\Users\\pupki\\Desktop\\
first_operations.csv", "r")
reader1 = csv.reader(first_operations)
insertion_sort_r5 = []
insertion_sort_r4000 = []
insertion_sort_as = []
insertion_sort_rs = []
for row in reader1:
    insertion_sort_r5.append(int(row[16]))
    insertion_sort_r4000.append(int(row[17]))
    insertion_sort_as.append(int(row[18]))
    insertion_sort_rs.append(int(row[19]))
plt.plot(first_scale, insertion_sort_r5, label = 'random 0-5')
plt.plot(first_scale, insertion_sort_r4000, label = 'random 0-4100')
plt.plot(first_scale, insertion_sort_as, label = 'almost sorted')
plt.plot(first_scale, insertion_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Insertion Sort')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```

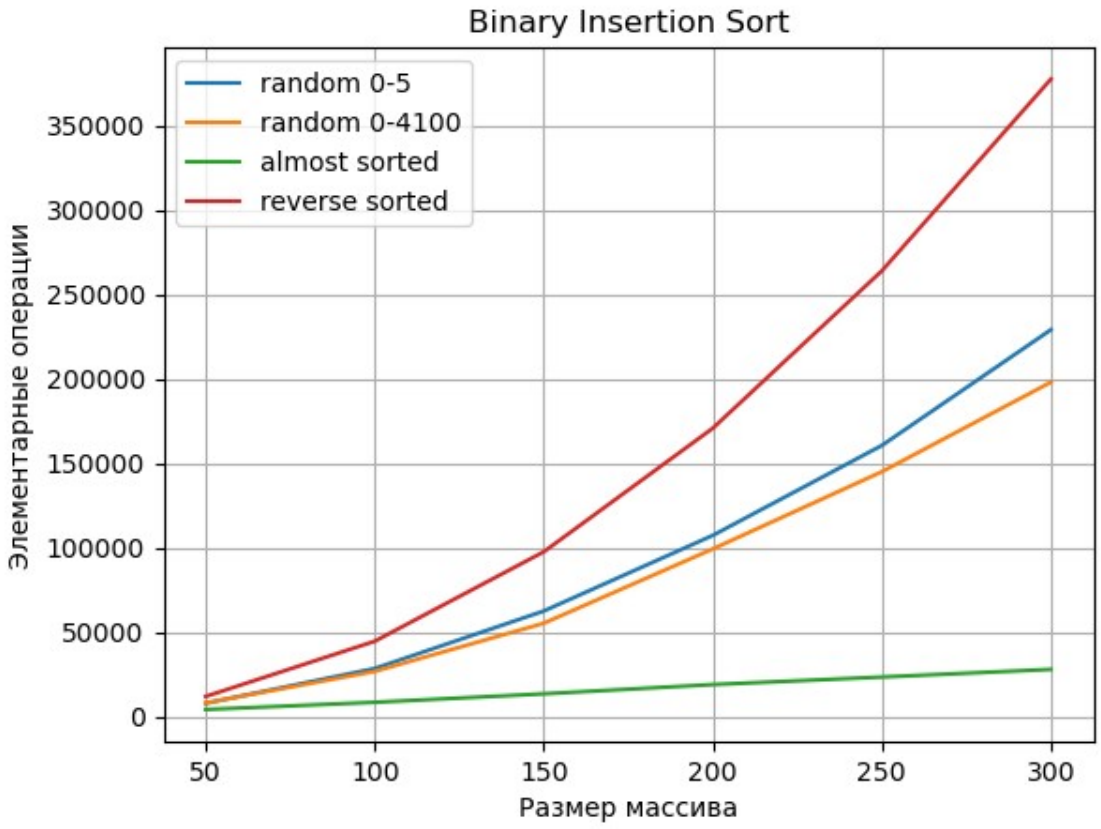



```
second_operations = open("C:\\Users\\pupki\\Desktop\\
second_operations.csv", "r")
reader2 = csv.reader(second_operations)
insertion_sort_r5 = []
insertion_sort_r4000 = []
insertion_sort_as = []
insertion_sort_rs = []
for row in reader2:
    insertion_sort_r5.append(int(row[16]))
    insertion_sort_r4000.append(int(row[17]))
    insertion_sort_as.append(int(row[18]))
    insertion_sort_rs.append(int(row[19]))
plt.plot(second_scale, insertion_sort_r5, label = 'random 0-5')
plt.plot(second_scale, insertion_sort_r4000, label = 'random 0-4100')
plt.plot(second_scale, insertion_sort_as, label = 'almost sorted')
plt.plot(second_scale, insertion_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Insertion Sort')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```



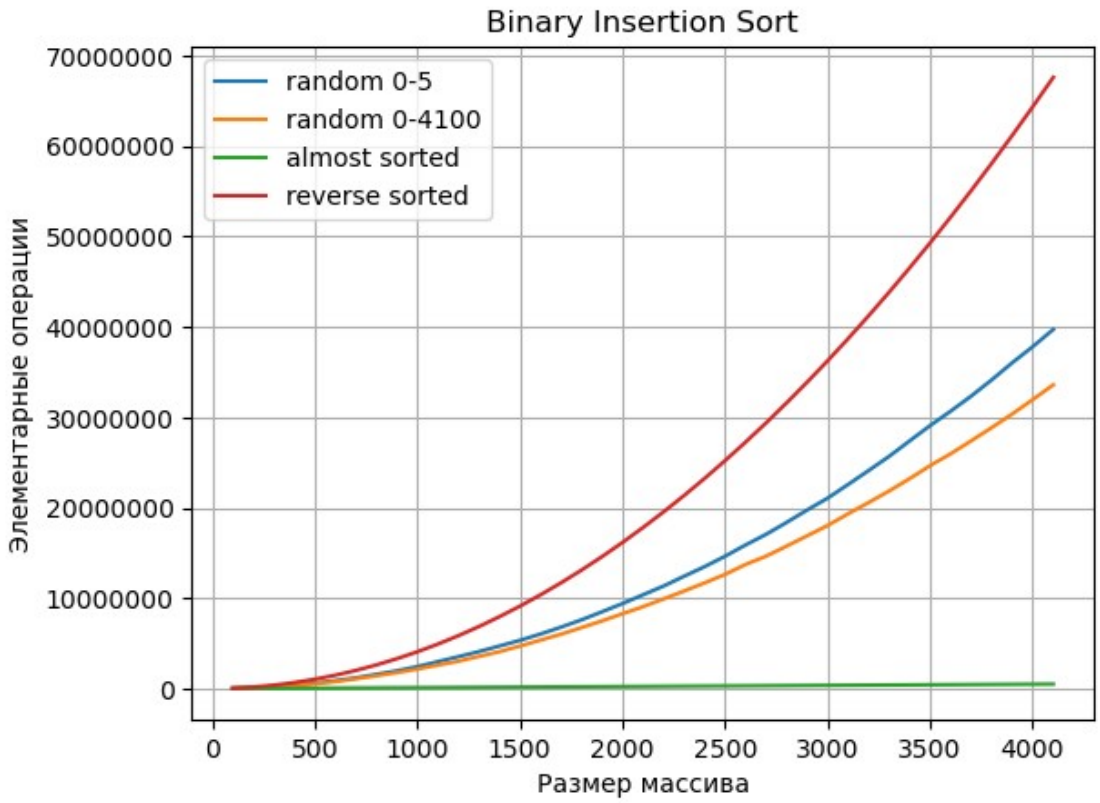

```
first_operations = open("C:\\Users\\pupki\\Desktop\\
first_operations.csv", "r")
reader1 = csv.reader(first_operations)
binary_insertion_sort_r5 = []
binary_insertion_sort_r4000 = []
binary_insertion_sort_as = []
binary_insertion_sort_rs = []
for row in reader1:
    binary_insertion_sort_r5.append(int(row[20]))
    binary_insertion_sort_r4000.append(int(row[21]))
    binary_insertion_sort_as.append(int(row[22]))
    binary_insertion_sort_rs.append(int(row[23]))
plt.plot(first_scale, binary_insertion_sort_r5, label = 'random 0-5')
plt.plot(first_scale, binary_insertion_sort_r4000, label = 'random 0-
4100')
plt.plot(first_scale, binary_insertion_sort_as, label = 'almost
sorted')
plt.plot(first_scale, binary_insertion_sort_rs, label = 'reverse
sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Binary Insertion Sort')
plt.legend(loc = 'best')
```

```
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```



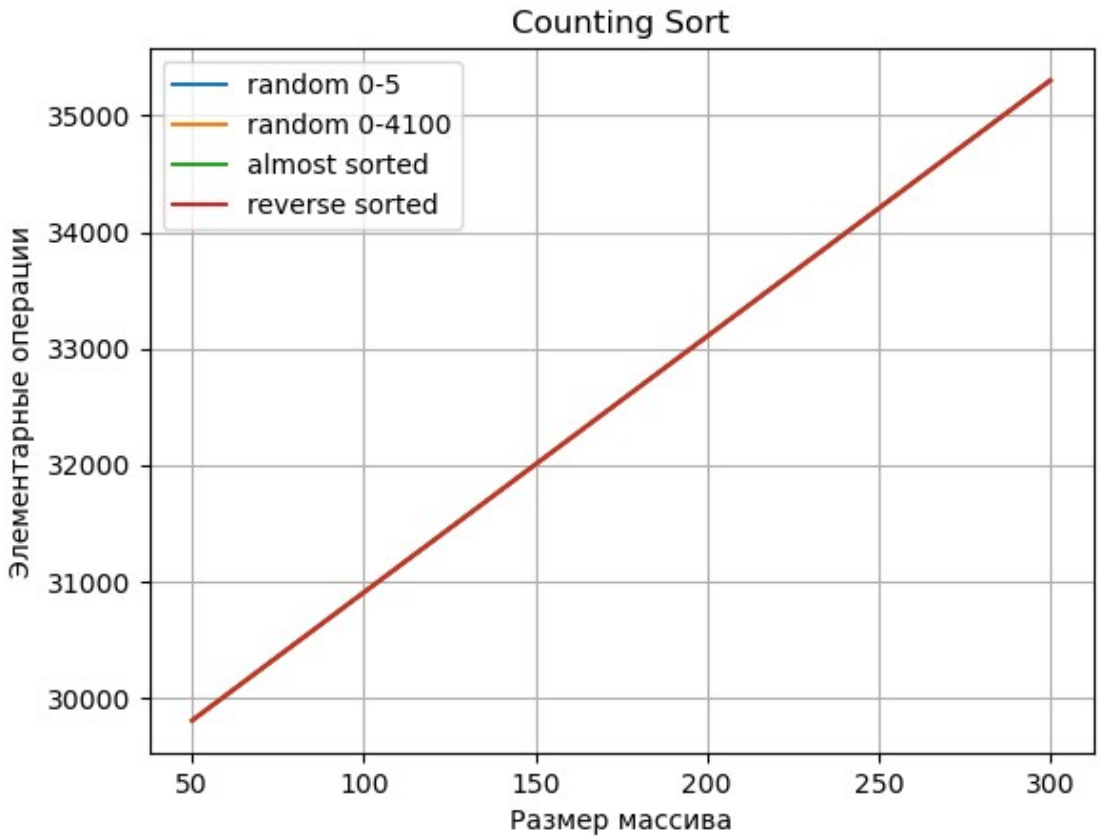

```
second_operations = open("C:\\Users\\pupki\\Desktop\\
second_operations.csv", "r")
reader2 = csv.reader(second_operations)
binary_insertion_sort_r5 = []
binary_insertion_sort_r4000 = []
binary_insertion_sort_as = []
binary_insertion_sort_rs = []
for row in reader2:
    binary_insertion_sort_r5.append(int(row[20]))
    binary_insertion_sort_r4000.append(int(row[21]))
    binary_insertion_sort_as.append(int(row[22]))
    binary_insertion_sort_rs.append(int(row[23]))
plt.plot(second_scale, binary_insertion_sort_r5, label = 'random 0-5')
plt.plot(second_scale, binary_insertion_sort_r4000, label = 'random 0-
4100')
plt.plot(second_scale, binary_insertion_sort_as, label = 'almost
sorted')
plt.plot(second_scale, binary_insertion_sort_rs, label = 'reverse
sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
```

```
plt.title('Binary Insertion Sort')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```



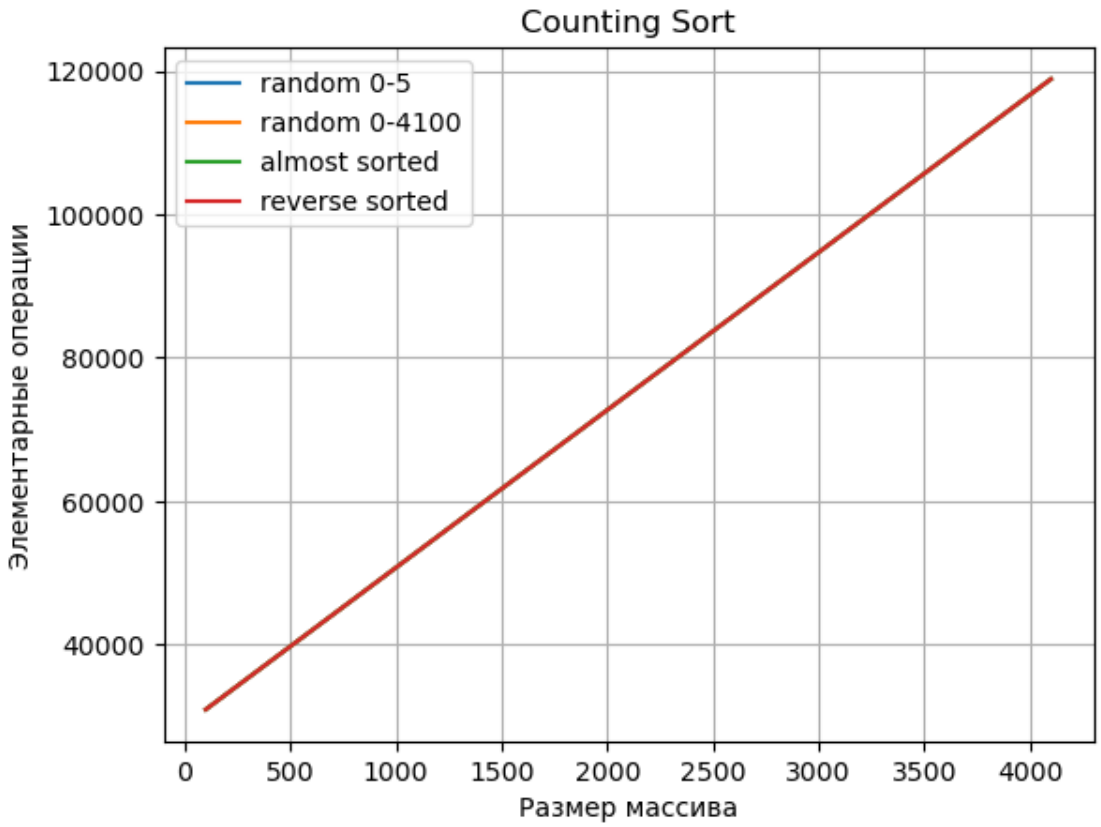

```
first_operations = open("C:\\Users\\pupki\\Desktop\\
first_operations.csv", "r")
reader1 = csv.reader(first_operations)
counting_sort_r5 = []
counting_sort_r4000 = []
counting_sort_as = []
counting_sort_rs = []
for row in reader1:
    counting_sort_r5.append(int(row[24]))
    counting_sort_r4000.append(int(row[25]))
    counting_sort_as.append(int(row[26]))
    counting_sort_rs.append(int(row[27]))
plt.plot(first_scale, counting_sort_r5, label = 'random 0-5')
plt.plot(first_scale, counting_sort_r4000, label = 'random 0-4100')
plt.plot(first_scale, counting_sort_as, label = 'almost sorted')
plt.plot(first_scale, counting_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Counting Sort')
plt.legend(loc = 'best')
```

```
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```



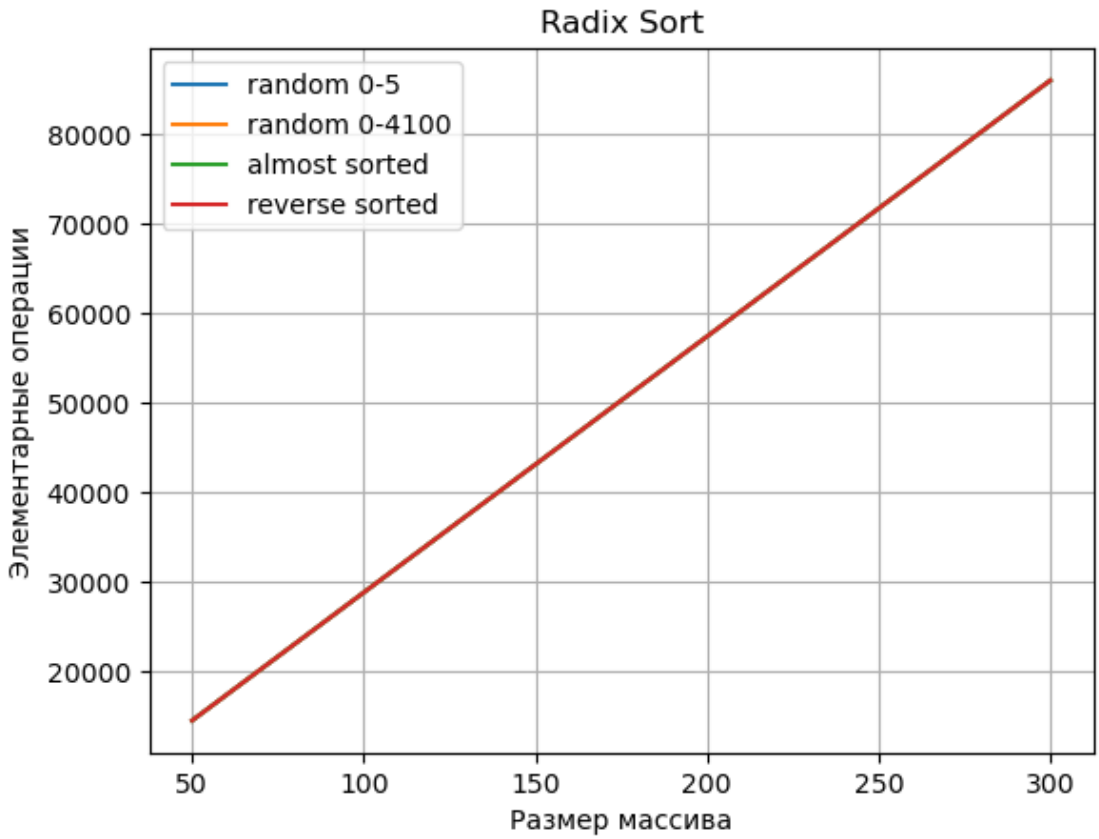

```
second_operations = open("C:\\Users\\pupki\\Desktop\\
second_operations.csv", "r")
reader2 = csv.reader(second_operations)
counting_sort_r5 = []
counting_sort_r4000 = []
counting_sort_as = []
counting_sort_rs = []
for row in reader2:
    counting_sort_r5.append(int(row[24]))
    counting_sort_r4000.append(int(row[25]))
    counting_sort_as.append(int(row[26]))
    counting_sort_rs.append(int(row[27]))
plt.plot(second_scale, counting_sort_r5, label = 'random 0-5')
plt.plot(second_scale, counting_sort_r4000, label = 'random 0-4100')
plt.plot(second_scale, counting_sort_as, label = 'almost sorted')
plt.plot(second_scale, counting_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Counting Sort')
plt.legend(loc = 'best')
```

```
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```



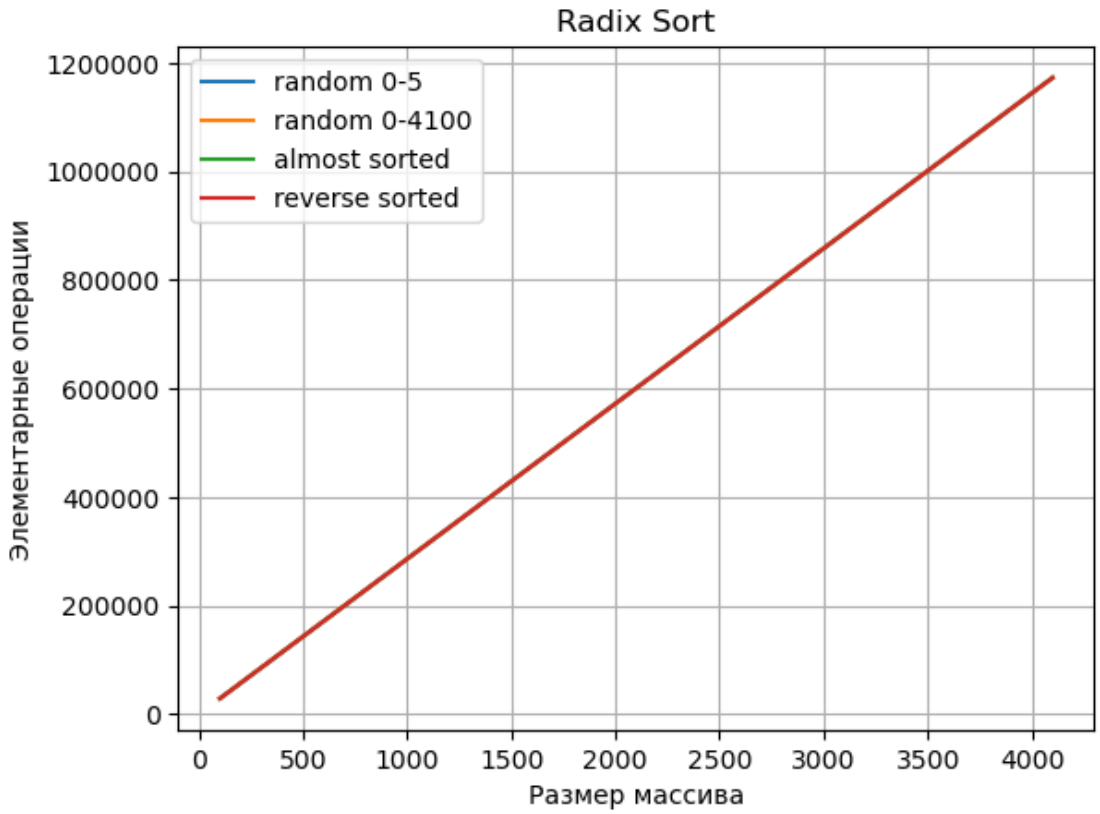

```
first_operations = open("C:\\Users\\pupki\\Desktop\\
first_operations.csv", "r")
reader1 = csv.reader(first_operations)
radix_sort_r5 = []
radix_sort_r4000 = []
radix_sort_as = []
radix_sort_rs = []
for row in reader1:
    radix_sort_r5.append(int(row[28]))
    radix_sort_r4000.append(int(row[29]))
    radix_sort_as.append(int(row[30]))
    radix_sort_rs.append(int(row[31]))
plt.plot(first_scale, radix_sort_r5, label = 'random 0-5')
plt.plot(first_scale, radix_sort_r4000, label = 'random 0-4100')
plt.plot(first_scale, radix_sort_as, label = 'almost sorted')
plt.plot(first_scale, radix_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Radix Sort')
plt.legend(loc = 'best')
```

```
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```



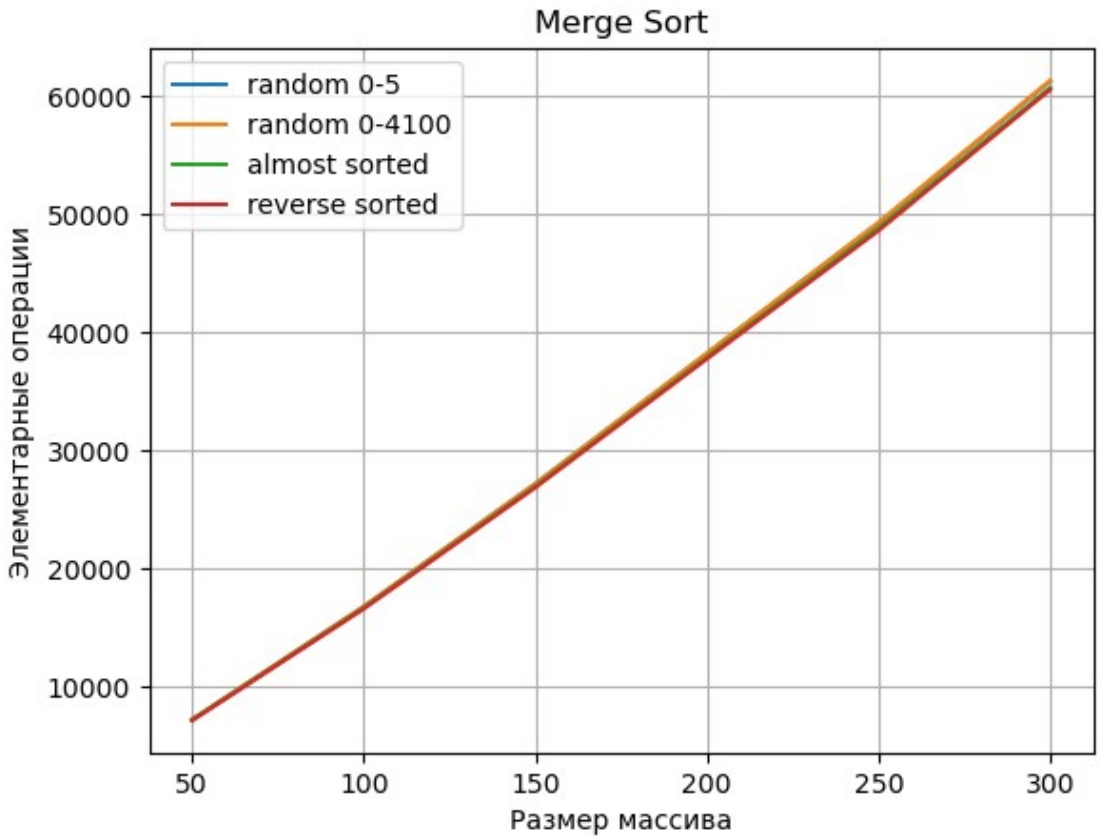

```
second_operations = open("C:\\Users\\pupki\\Desktop\\
second_operations.csv", "r")
reader2 = csv.reader(second_operations)
radix_sort_r5 = []
radix_sort_r4000 = []
radix_sort_as = []
radix_sort_rs = []
for row in reader2:
    radix_sort_r5.append(int(row[28]))
    radix_sort_r4000.append(int(row[29]))
    radix_sort_as.append(int(row[30]))
    radix_sort_rs.append(int(row[31]))
plt.plot(second_scale, radix_sort_r5, label = 'random 0-5')
plt.plot(second_scale, radix_sort_r4000, label = 'random 0-4100')
plt.plot(second_scale, radix_sort_as, label = 'almost sorted')
plt.plot(second_scale, radix_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Radix Sort')
plt.legend(loc = 'best')
```

```
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```



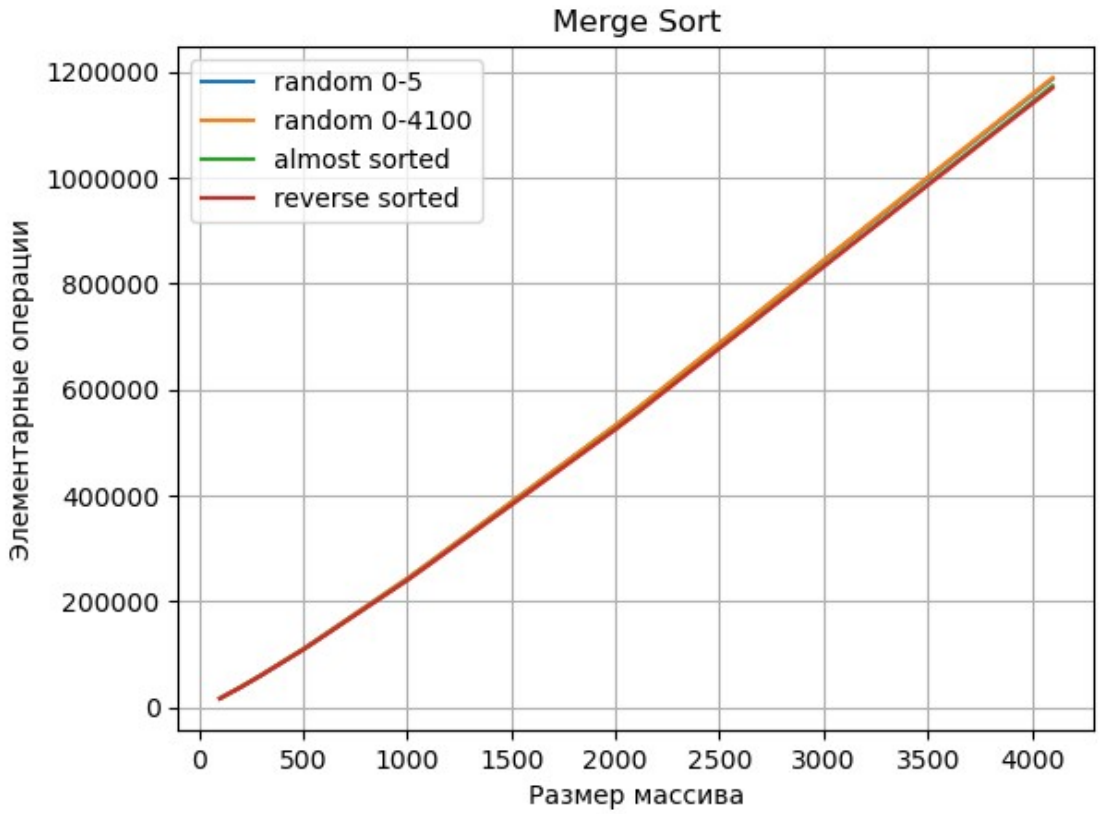

```
first_operations = open("C:\\Users\\pupki\\Desktop\\
first_operations.csv", "r")
reader1 = csv.reader(first_operations)
merge_sort_r5 = []
merge_sort_r4000 = []
merge_sort_as = []
merge_sort_rs = []
for row in reader1:
    merge_sort_r5.append(int(row[32]))
    merge_sort_r4000.append(int(row[33]))
    merge_sort_as.append(int(row[34]))
    merge_sort_rs.append(int(row[35]))
plt.plot(first_scale, merge_sort_r5, label = 'random 0-5')
plt.plot(first_scale, merge_sort_r4000, label = 'random 0-4100')
plt.plot(first_scale, merge_sort_as, label = 'almost sorted')
plt.plot(first_scale, merge_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Merge Sort')
plt.legend(loc = 'best')
```

```
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```



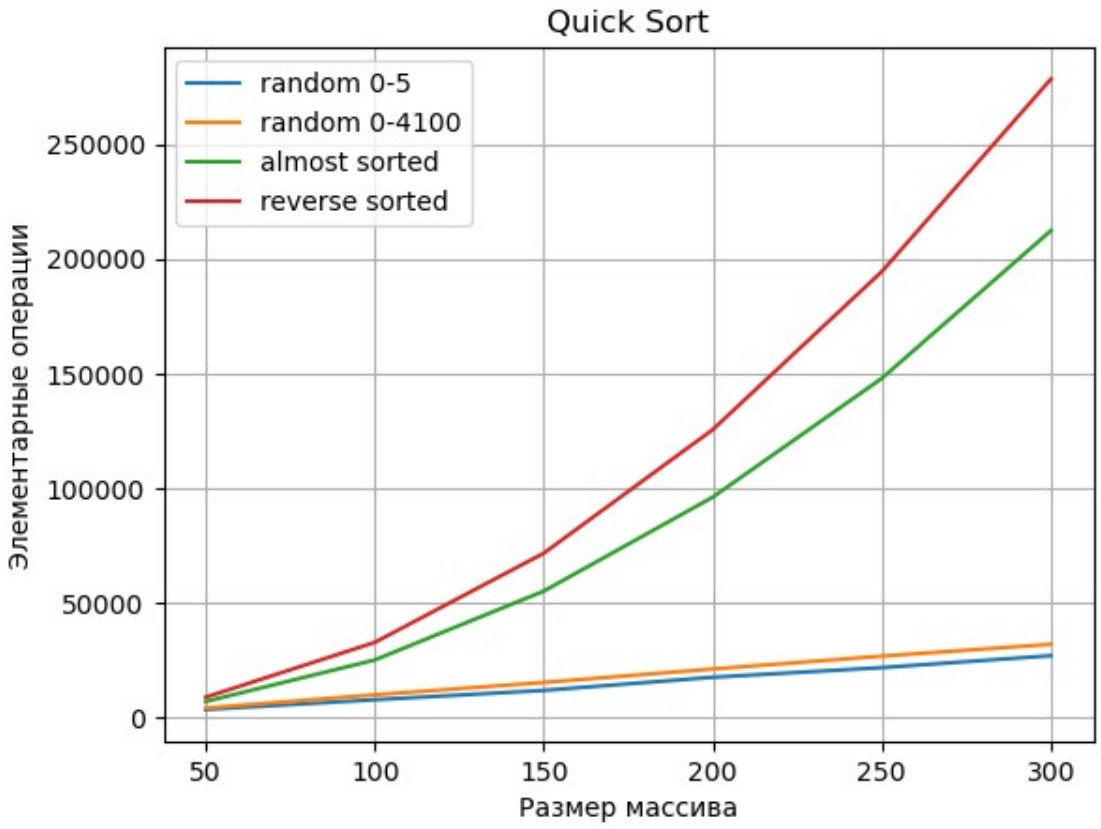

```
second_operations = open("C:\\Users\\pupki\\Desktop\\
second_operations.csv", "r")
reader2 = csv.reader(second_operations)
merge_sort_r5 = []
merge_sort_r4000 = []
merge_sort_as = []
merge_sort_rs = []
for row in reader2:
    merge_sort_r5.append(int(row[32]))
    merge_sort_r4000.append(int(row[33]))
    merge_sort_as.append(int(row[34]))
    merge_sort_rs.append(int(row[35]))
plt.plot(second_scale, merge_sort_r5, label = 'random 0-5')
plt.plot(second_scale, merge_sort_r4000, label = 'random 0-4100')
plt.plot(second_scale, merge_sort_as, label = 'almost sorted')
plt.plot(second_scale, merge_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Merge Sort')
plt.legend(loc = 'best')
```

```
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```



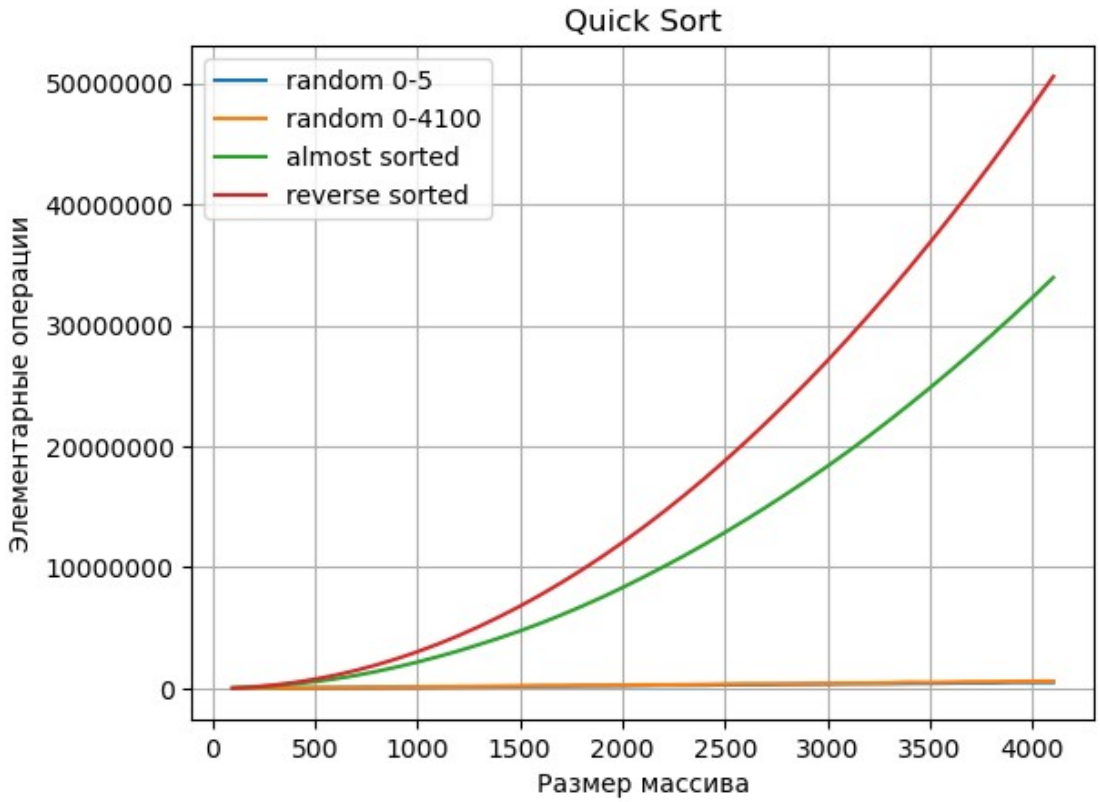

```
first_operations = open("C:\\Users\\pupki\\Desktop\\
first_operations.csv", "r")
reader1 = csv.reader(first_operations)
quick_sort_r5 = []
quick_sort_r4000 = []
quick_sort_as = []
quick_sort_rs = []
for row in reader1:
    quick_sort_r5.append(int(row[36]))
    quick_sort_r4000.append(int(row[37]))
    quick_sort_as.append(int(row[38]))
    quick_sort_rs.append(int(row[39]))
plt.plot(first_scale, quick_sort_r5, label = 'random 0-5')
plt.plot(first_scale, quick_sort_r4000, label = 'random 0-4100')
plt.plot(first_scale, quick_sort_as, label = 'almost sorted')
plt.plot(first_scale, quick_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Quick Sort')
plt.legend(loc = 'best')
```

```
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```

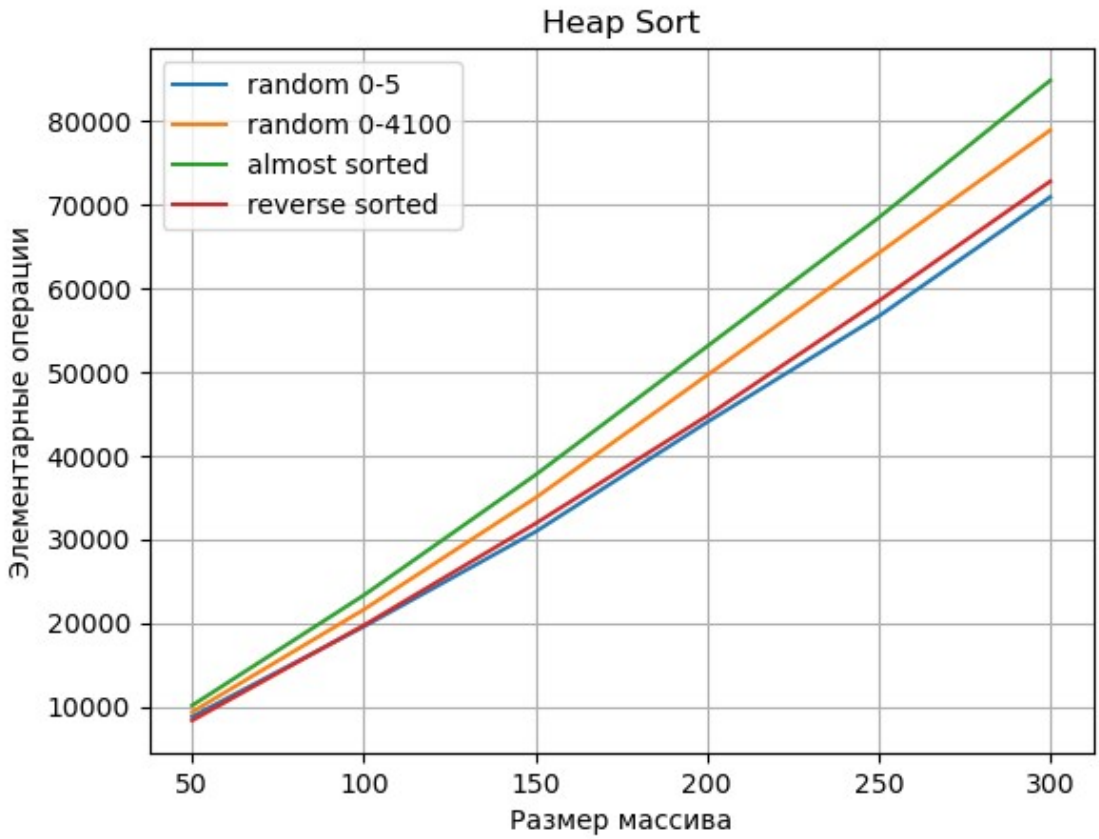



```
second_operations = open("C:\\Users\\pupki\\Desktop\\
second_operations.csv", "r")
reader2 = csv.reader(second_operations)
quick_sort_r5 = []
quick_sort_r4000 = []
quick_sort_as = []
quick_sort_rs = []
for row in reader2:
    quick_sort_r5.append(int(row[36]))
    quick_sort_r4000.append(int(row[37]))
    quick_sort_as.append(int(row[38]))
    quick_sort_rs.append(int(row[39]))
plt.plot(second_scale, quick_sort_r5, label = 'random 0-5')
plt.plot(second_scale, quick_sort_r4000, label = 'random 0-4100')
plt.plot(second_scale, quick_sort_as, label = 'almost sorted')
plt.plot(second_scale, quick_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Quick Sort')
plt.legend(loc = 'best')
```

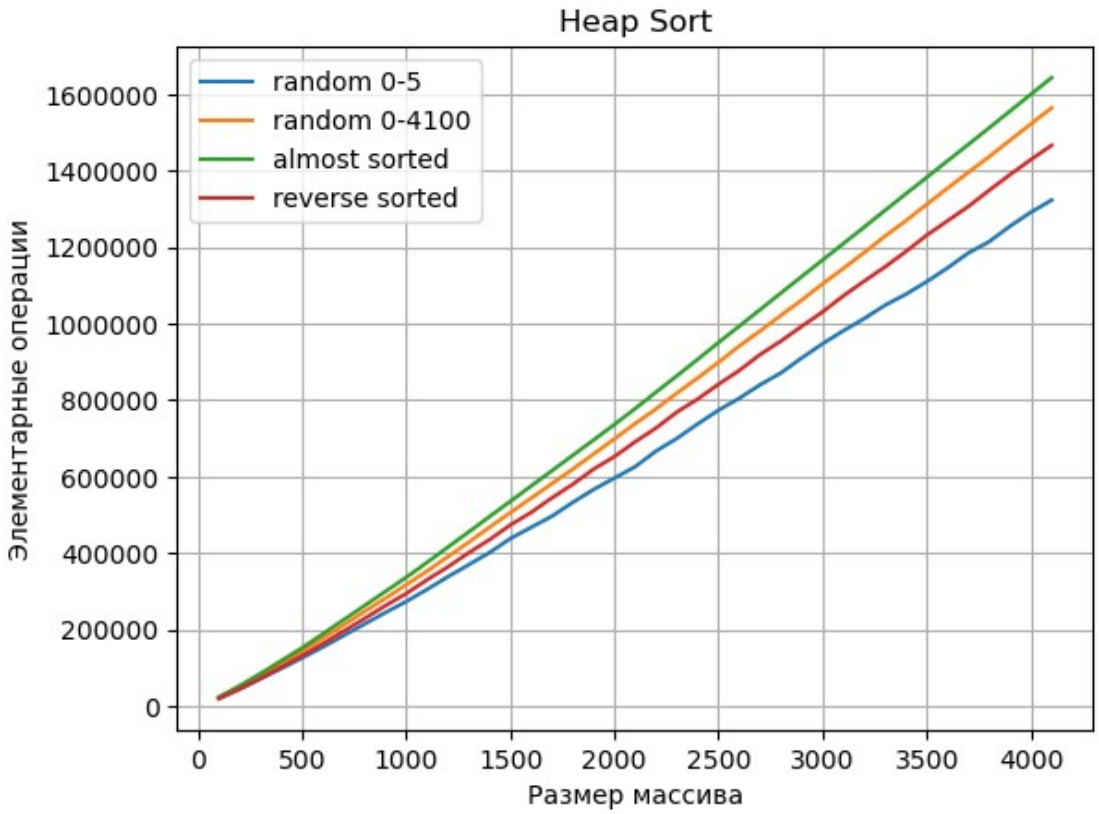
```
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```



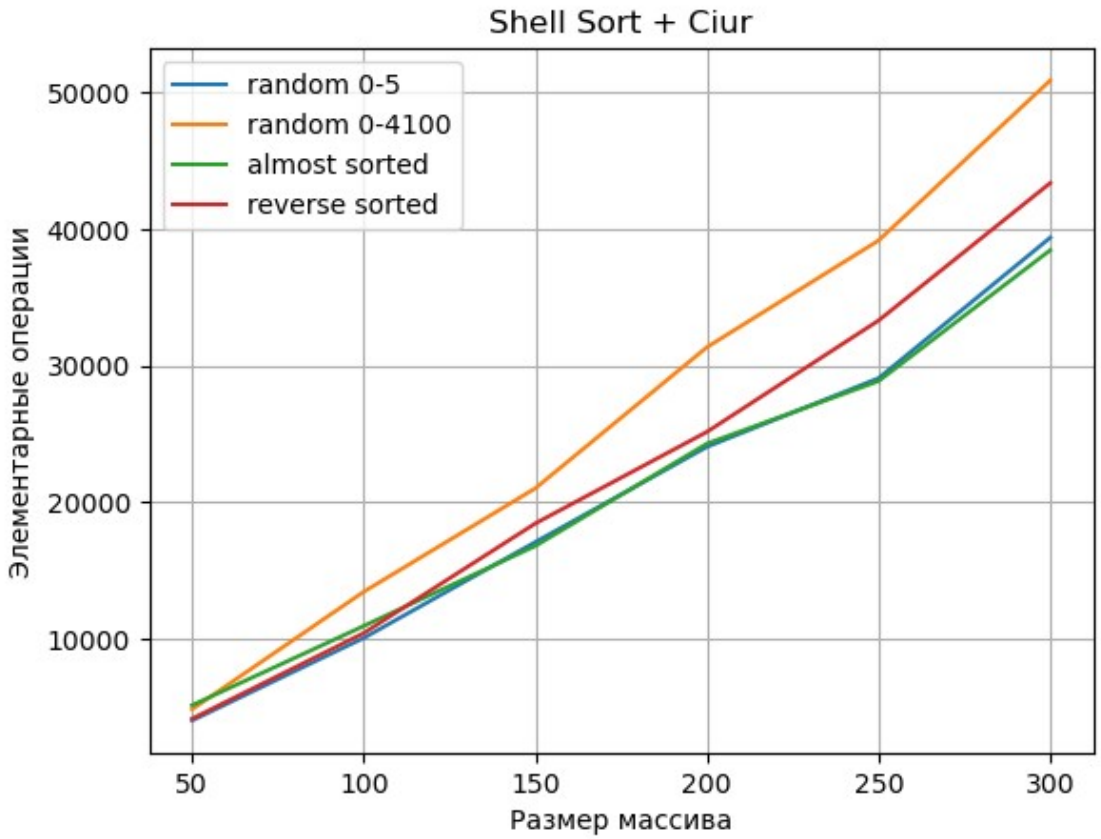

```
first_operations = open("C:\\Users\\pupki\\Desktop\\
first_operations.csv", "r")
reader1 = csv.reader(first_operations)
heap_sort_r5 = []
heap_sort_r4000 = []
heap_sort_as = []
heap_sort_rs = []
for row in reader1:
    heap_sort_r5.append(int(row[40]))
    heap_sort_r4000.append(int(row[41]))
    heap_sort_as.append(int(row[42]))
    heap_sort_rs.append(int(row[43]))
plt.plot(first_scale, heap_sort_r5, label = 'random 0-5')
plt.plot(first_scale, heap_sort_r4000, label = 'random 0-4100')
plt.plot(first_scale, heap_sort_as, label = 'almost sorted')
plt.plot(first_scale, heap_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Heap Sort')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```



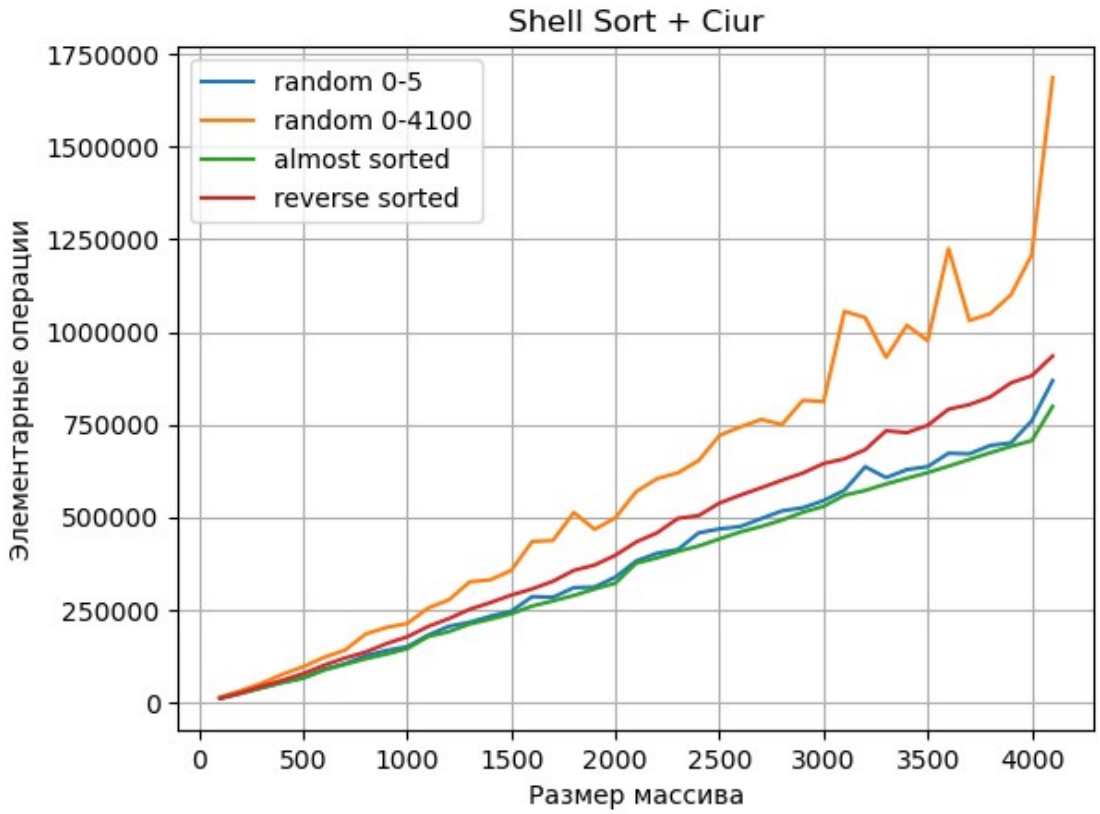

```
second_operations = open("C:\\Users\\pupki\\Desktop\\
second_operations.csv", "r")
reader2 = csv.reader(second_operations)
heap_sort_r5 = []
heap_sort_r4000 = []
heap_sort_as = []
heap_sort_rs = []
for row in reader2:
    heap_sort_r5.append(int(row[40]))
    heap_sort_r4000.append(int(row[41]))
    heap_sort_as.append(int(row[42]))
    heap_sort_rs.append(int(row[43]))
plt.plot(second_scale, heap_sort_r5, label = 'random 0-5')
plt.plot(second_scale, heap_sort_r4000, label = 'random 0-4100')
plt.plot(second_scale, heap_sort_as, label = 'almost sorted')
plt.plot(second_scale, heap_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Heap Sort')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```



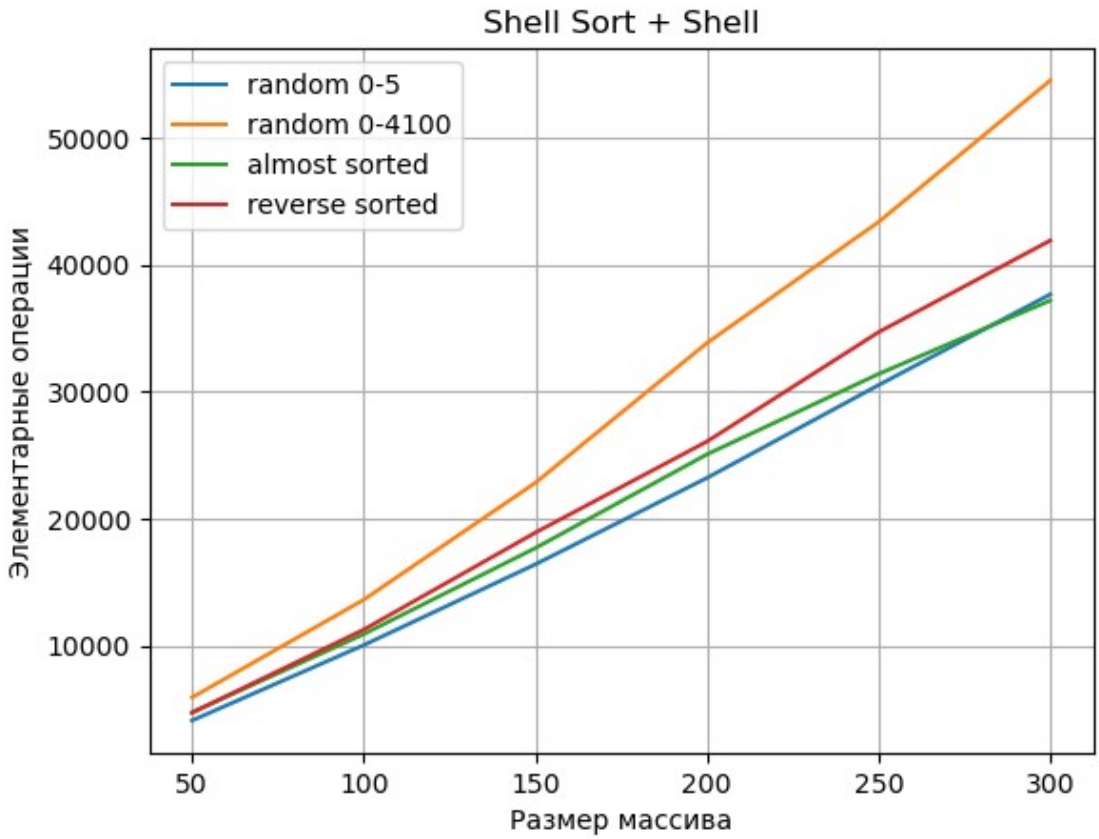

```
first_operations = open("C:\\Users\\pupki\\Desktop\\
first_operations.csv", "r")
reader1 = csv.reader(first_operations)
shell1_sort_r5 = []
shell1_sort_r4000 = []
shell1_sort_as = []
shell1_sort_rs = []
for row in reader1:
    shell1_sort_r5.append(int(row[44]))
    shell1_sort_r4000.append(int(row[45]))
    shell1_sort_as.append(int(row[46]))
    shell1_sort_rs.append(int(row[47]))
plt.plot(first_scale, shell1_sort_r5, label = 'random 0-5')
plt.plot(first_scale, shell1_sort_r4000, label = 'random 0-4100')
plt.plot(first_scale, shell1_sort_as, label = 'almost sorted')
plt.plot(first_scale, shell1_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Shell Sort + Ciur')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```



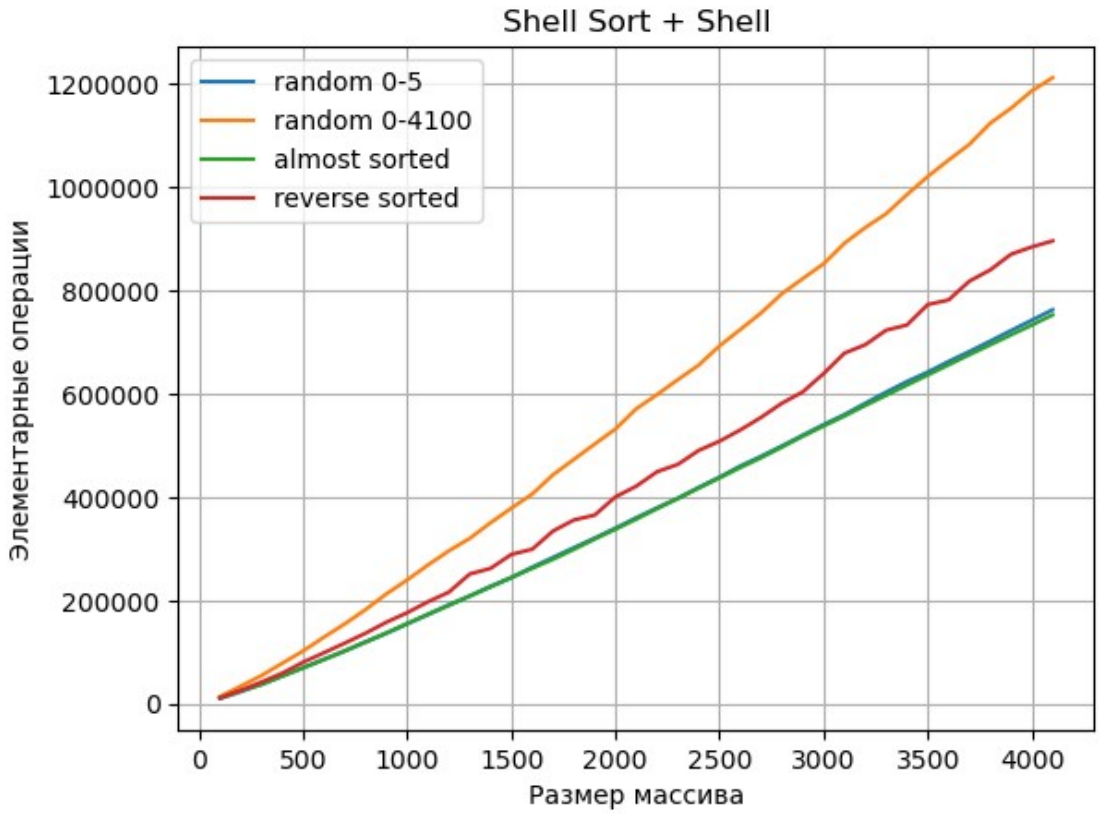

```
second_operations = open("C:\\Users\\pupki\\Desktop\\
second_operations.csv", "r")
reader2 = csv.reader(second_operations)
shell1_sort_r5 = []
shell1_sort_r4000 = []
shell1_sort_as = []
shell1_sort_rs = []
for row in reader2:
    shell1_sort_r5.append(int(row[44]))
    shell1_sort_r4000.append(int(row[45]))
    shell1_sort_as.append(int(row[46]))
    shell1_sort_rs.append(int(row[47]))
plt.plot(second_scale, shell1_sort_r5, label = 'random 0-5')
plt.plot(second_scale, shell1_sort_r4000, label = 'random 0-4100')
plt.plot(second_scale, shell1_sort_as, label = 'almost sorted')
plt.plot(second_scale, shell1_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Shell Sort + Ciur')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




```
first_operations = open("C:\\Users\\pupki\\Desktop\\
first_operations.csv", "r")
reader1 = csv.reader(first_operations)
shell2_sort_r5 = []
shell2_sort_r4000 = []
shell2_sort_as = []
shell2_sort_rs = []
for row in reader1:
    shell2_sort_r5.append(int(row[48]))
    shell2_sort_r4000.append(int(row[49]))
    shell2_sort_as.append(int(row[50]))
    shell2_sort_rs.append(int(row[51]))
plt.plot(first_scale, shell2_sort_r5, label = 'random 0-5')
plt.plot(first_scale, shell2_sort_r4000, label = 'random 0-4100')
plt.plot(first_scale, shell2_sort_as, label = 'almost sorted')
plt.plot(first_scale, shell2_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Shell Sort + Shell')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```

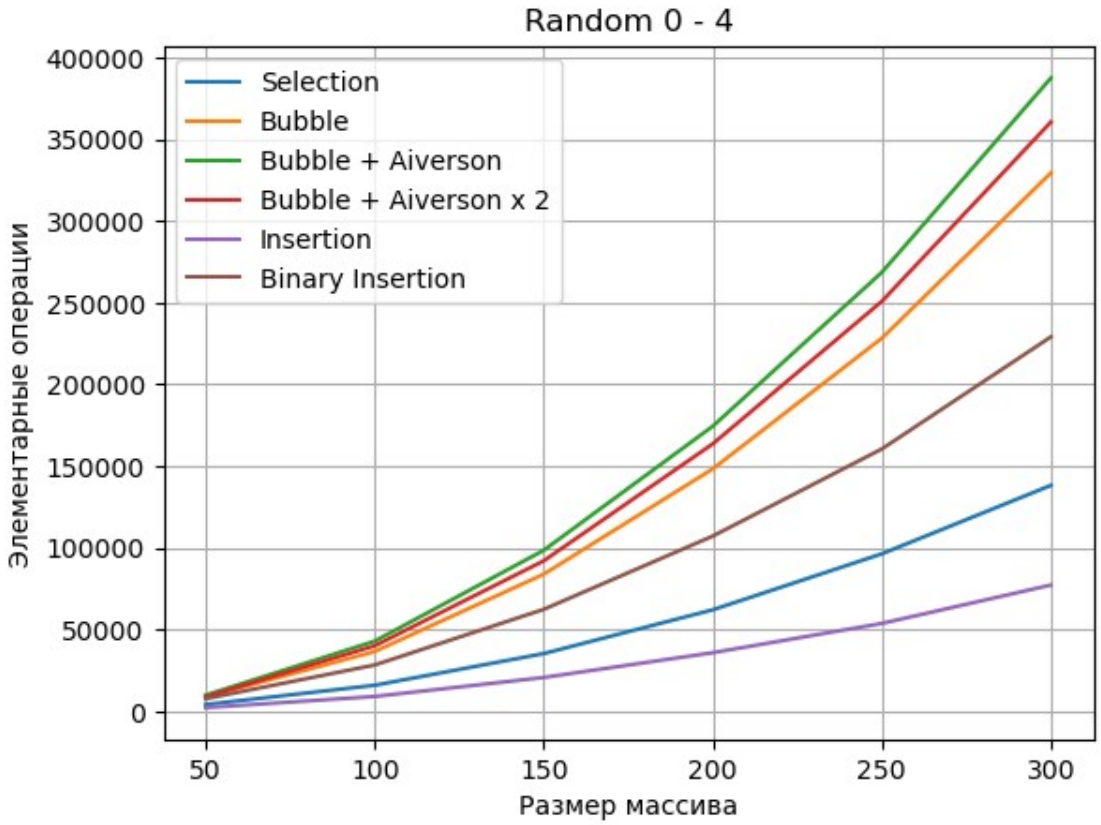



```
second_operations = open("C:\\Users\\pupki\\Desktop\\
second_operations.csv", "r")
reader2 = csv.reader(second_operations)
shell2_sort_r5 = []
shell2_sort_r4000 = []
shell2_sort_as = []
shell2_sort_rs = []
for row in reader2:
    shell2_sort_r5.append(int(row[48]))
    shell2_sort_r4000.append(int(row[49]))
    shell2_sort_as.append(int(row[50]))
    shell2_sort_rs.append(int(row[51]))
plt.plot(second_scale, shell2_sort_r5, label = 'random 0-5')
plt.plot(second_scale, shell2_sort_r4000, label = 'random 0-4100')
plt.plot(second_scale, shell2_sort_as, label = 'almost sorted')
plt.plot(second_scale, shell2_sort_rs, label = 'reverse sorted')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Shell Sort + Shell')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




```
first_operations = open("C:\\Users\\pupki\\Desktop\\
first_operations.csv", "r")
reader1 = csv.reader(first_operations)
selection_sort = []
bubble_sort = []
bubble_1_sort = []
bubble_2_sort = []
insertion_sort = []
binary_insertion_sort = []
for row in reader1:
    selection_sort.append(int(row[0]))
    bubble_sort.append(int(row[4]))
    bubble_1_sort.append(int(row[8]))
    bubble_2_sort.append(int(row[12]))
    insertion_sort.append(int(row[16]))
    binary_insertion_sort.append(int(row[20]))
plt.plot(first_scale, selection_sort, label = 'Selection')
plt.plot(first_scale, bubble_sort, label = 'Bubble')
plt.plot(first_scale, bubble_1_sort, label = 'Bubble + Aiverson')
plt.plot(first_scale, bubble_2_sort, label = 'Bubble + Aiverson x 2')
plt.plot(first_scale, insertion_sort, label = 'Insertion')
plt.plot(first_scale, binary_insertion_sort, label = 'Binary
Insertion')
plt.grid(True)
plt.xlabel('Размер массива')
```

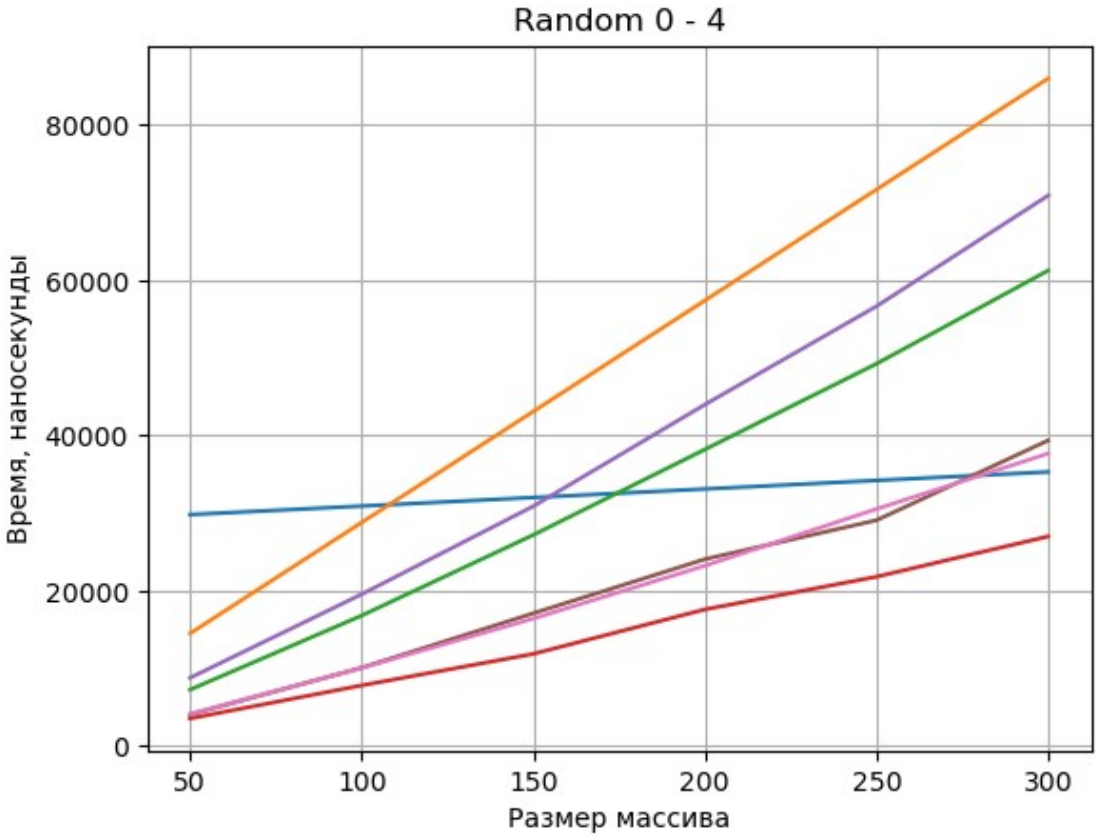
```
plt.ylabel('Элементарные операции')
plt.title('Random 0 - 4')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




```
first_operations = open("C:\\Users\\pupki\\Desktop\\
first_operations.csv", "r")
reader1 = csv.reader(first_operations)
counting_sort = []
radix_sort = []
merge_sort = []
quick_sort = []
heap_sort = []
shell_1_sort = []
shell_2_sort = []
for row in reader1:
    counting_sort.append(int(row[24]))
    radix_sort.append(int(row[28]))
    merge_sort.append(int(row[32]))
    quick_sort.append(int(row[36]))
    heap_sort.append(int(row[40]))
    shell_1_sort.append(int(row[44]))
    shell_2_sort.append(int(row[48]))
plt.plot(first_scale, counting_sort, label = 'Counting')
```

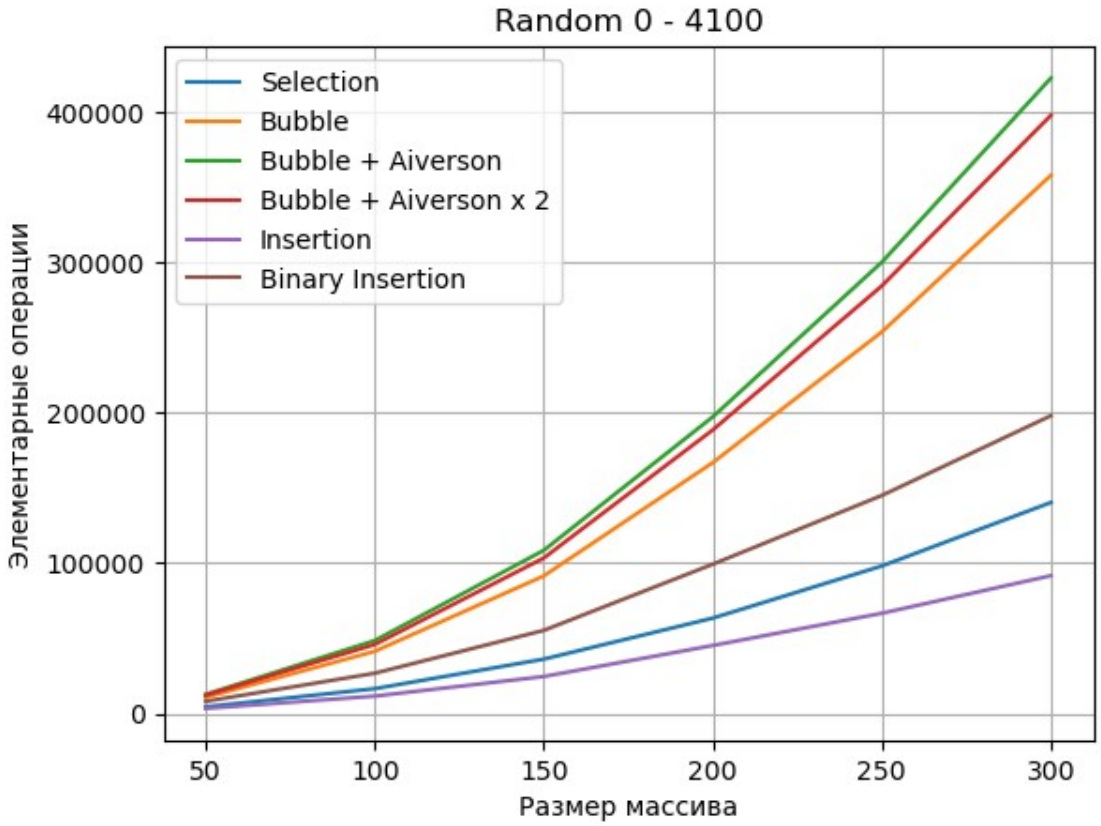
```
plt.plot(first_scale, radix_sort, label = 'Radix')
plt.plot(first_scale, merge_sort, label = 'Merge')
plt.plot(first_scale, quick_sort, label = 'Quick')
plt.plot(first_scale, heap_sort, label = 'Heap')
plt.plot(first_scale, shell_1_sort, label = 'Shell + Ciur')
plt.plot(first_scale, shell_2_sort, label = 'Shell + Shell')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Random 0 - 4')
```

```
Text(0.5, 1.0, 'Random 0 - 4')
```



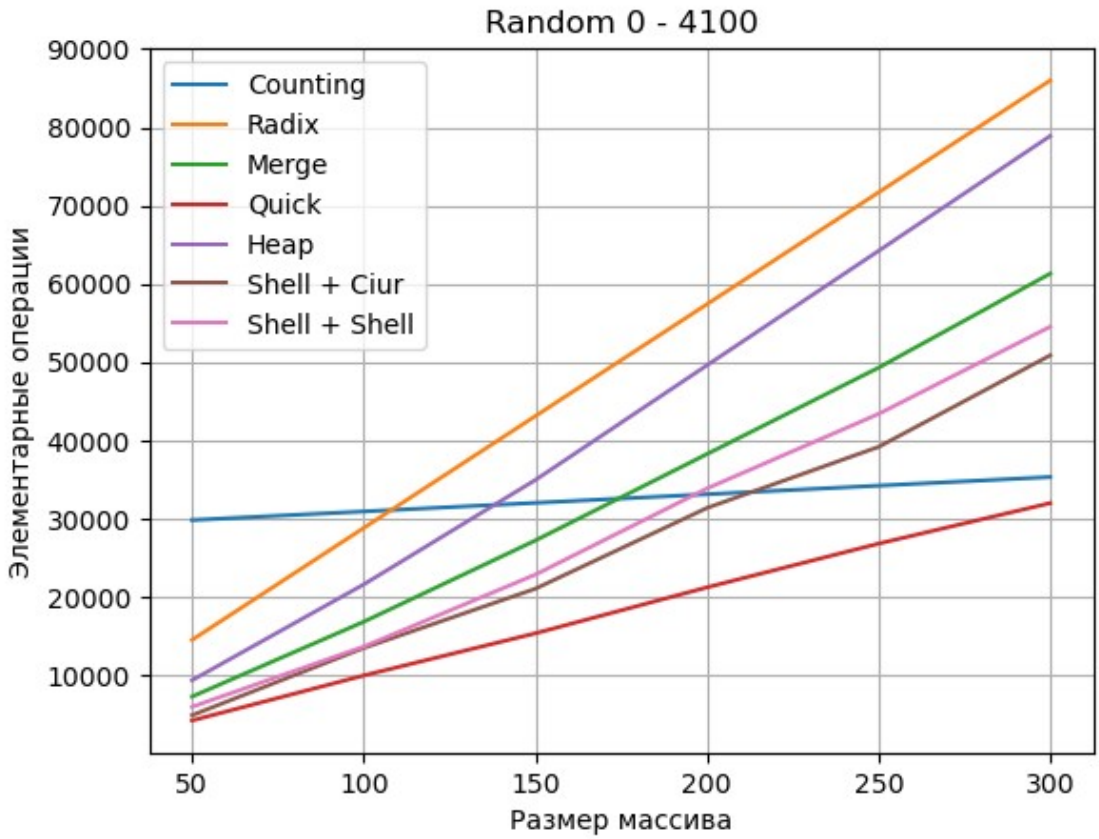

```
first_operations = open("C:\\Users\\pupki\\Desktop\\
first_operations.csv", "r")
reader1 = csv.reader(first_operations)
selection_sort = []
bubble_sort = []
bubble_1_sort = []
bubble_2_sort = []
insertion_sort = []
binary_insertion_sort = []
for row in reader1:
    selection_sort.append(int(row[1]))
    bubble_sort.append(int(row[5]))
```

```
    bubble_1_sort.append(int(row[9]))
    bubble_2_sort.append(int(row[13]))
    insertion_sort.append(int(row[17]))
    binary_insertion_sort.append(int(row[21]))
plt.plot(first_scale, selection_sort, label = 'Selection')
plt.plot(first_scale, bubble_sort, label = 'Bubble')
plt.plot(first_scale, bubble_1_sort, label = 'Bubble + Aiverson')
plt.plot(first_scale, bubble_2_sort, label = 'Bubble + Aiverson x 2')
plt.plot(first_scale, insertion_sort, label = 'Insertion')
plt.plot(first_scale, binary_insertion_sort, label = 'Binary
Insertion')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Random 0 - 4100')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```



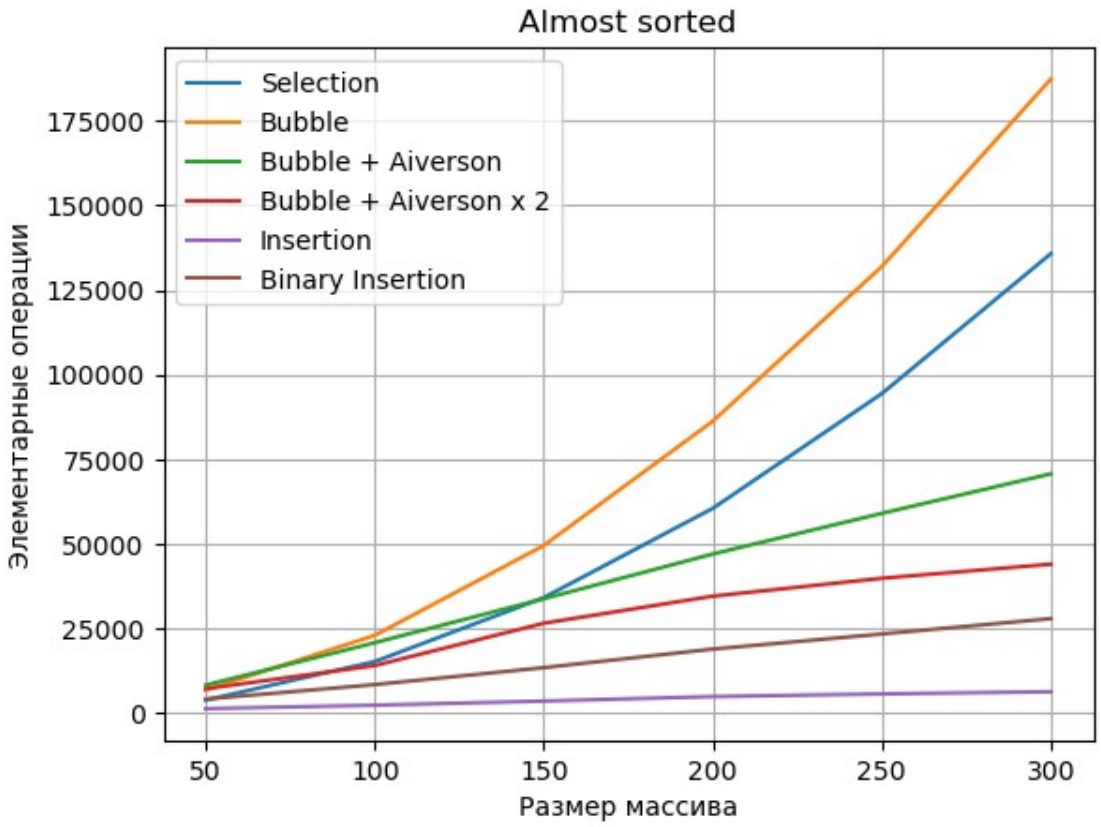

```
first_operations = open("C:\\Users\\pupki\\Desktop\\
first_operations.csv", "r")
reader1 = csv.reader(first_operations)
counting_sort = []
radix_sort = []
merge_sort = []
```

```
quick_sort = []
heap_sort = []
shell_1_sort = []
shell_2_sort = []
for row in reader1:
    counting_sort.append(int(row[25]))
    radix_sort.append(int(row[29]))
    merge_sort.append(int(row[33]))
    quick_sort.append(int(row[37]))
    heap_sort.append(int(row[41]))
    shell_1_sort.append(int(row[45]))
    shell_2_sort.append(int(row[49]))
plt.plot(first_scale, counting_sort, label = 'Counting')
plt.plot(first_scale, radix_sort, label = 'Radix')
plt.plot(first_scale, merge_sort, label = 'Merge')
plt.plot(first_scale, quick_sort, label = 'Quick')
plt.plot(first_scale, heap_sort, label = 'Heap')
plt.plot(first_scale, shell_1_sort, label = 'Shell + Ciur')
plt.plot(first_scale, shell_2_sort, label = 'Shell + Shell')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Random 0 - 4100')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```



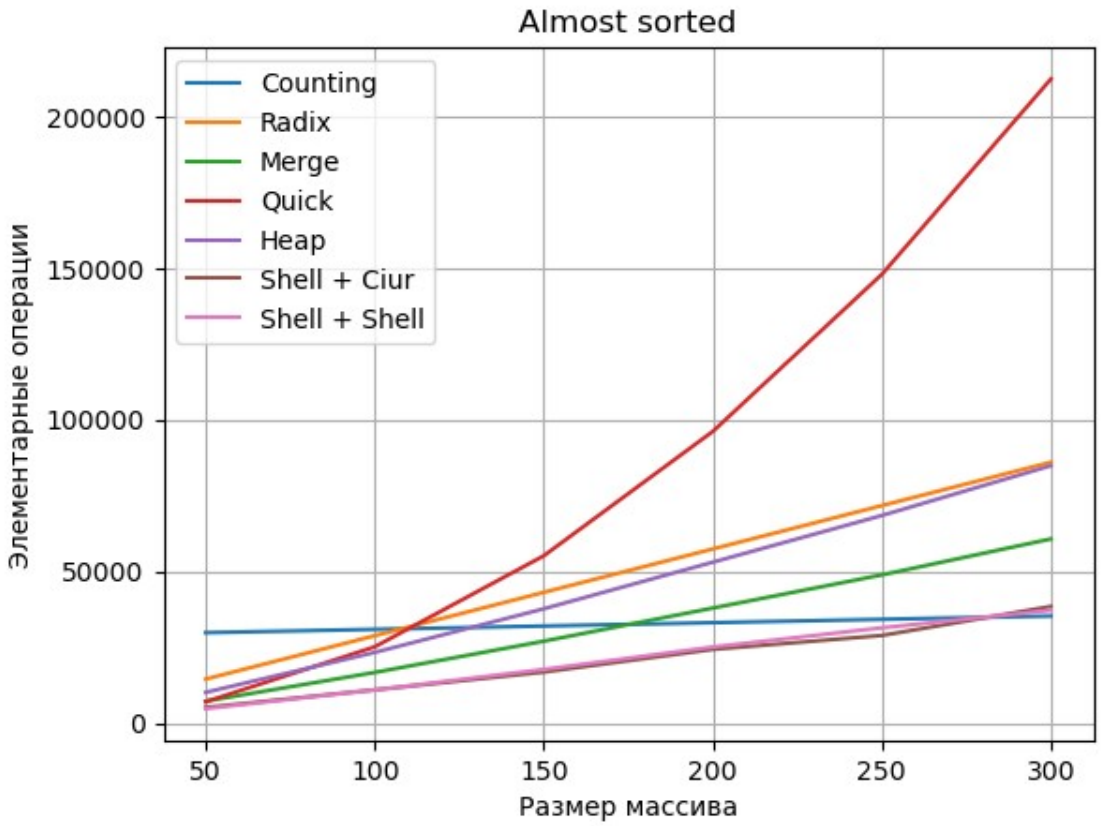

```
first_operations = open("C:\\Users\\pupki\\Desktop\\
first_operations.csv", "r")
reader1 = csv.reader(first_operations)
selection_sort = []
bubble_sort = []
bubble_1_sort = []
bubble_2_sort = []
insertion_sort = []
binary_insertion_sort = []
for row in reader1:
    selection_sort.append(int(row[2]))
    bubble_sort.append(int(row[6]))
    bubble_1_sort.append(int(row[10]))
    bubble_2_sort.append(int(row[14]))
    insertion_sort.append(int(row[18]))
    binary_insertion_sort.append(int(row[22]))
plt.plot(first_scale, selection_sort, label = 'Selection')
plt.plot(first_scale, bubble_sort, label = 'Bubble')
plt.plot(first_scale, bubble_1_sort, label = 'Bubble + Aiverson')
plt.plot(first_scale, bubble_2_sort, label = 'Bubble + Aiverson x 2')
plt.plot(first_scale, insertion_sort, label = 'Insertion')
plt.plot(first_scale, binary_insertion_sort, label = 'Binary
Insertion')
plt.grid(True)
```

```
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Almost sorted')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```



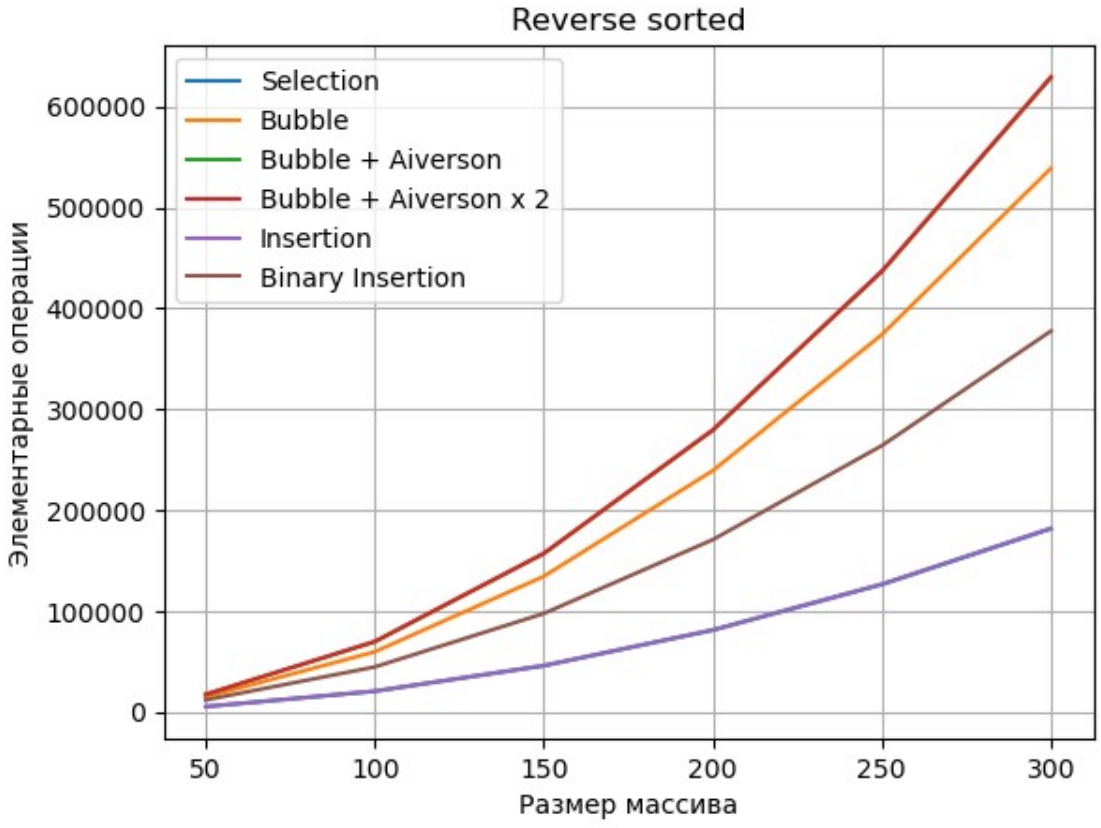

```
first_operations = open("C:\\Users\\pupki\\Desktop\\
first_operations.csv", "r")
reader1 = csv.reader(first_operations)
counting_sort = []
radix_sort = []
merge_sort = []
quick_sort = []
heap_sort = []
shell_1_sort = []
shell_2_sort = []
for row in reader1:
    counting_sort.append(int(row[26]))
    radix_sort.append(int(row[30]))
    merge_sort.append(int(row[34]))
    quick_sort.append(int(row[38]))
    heap_sort.append(int(row[42]))
    shell_1_sort.append(int(row[46]))
    shell_2_sort.append(int(row[50]))
```

```
plt.plot(first_scale, counting_sort, label = 'Counting')
plt.plot(first_scale, radix_sort, label = 'Radix')
plt.plot(first_scale, merge_sort, label = 'Merge')
plt.plot(first_scale, quick_sort, label = 'Quick')
plt.plot(first_scale, heap_sort, label = 'Heap')
plt.plot(first_scale, shell_1_sort, label = 'Shell + Ciur')
plt.plot(first_scale, shell_2_sort, label = 'Shell + Shell')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Almost sorted')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```



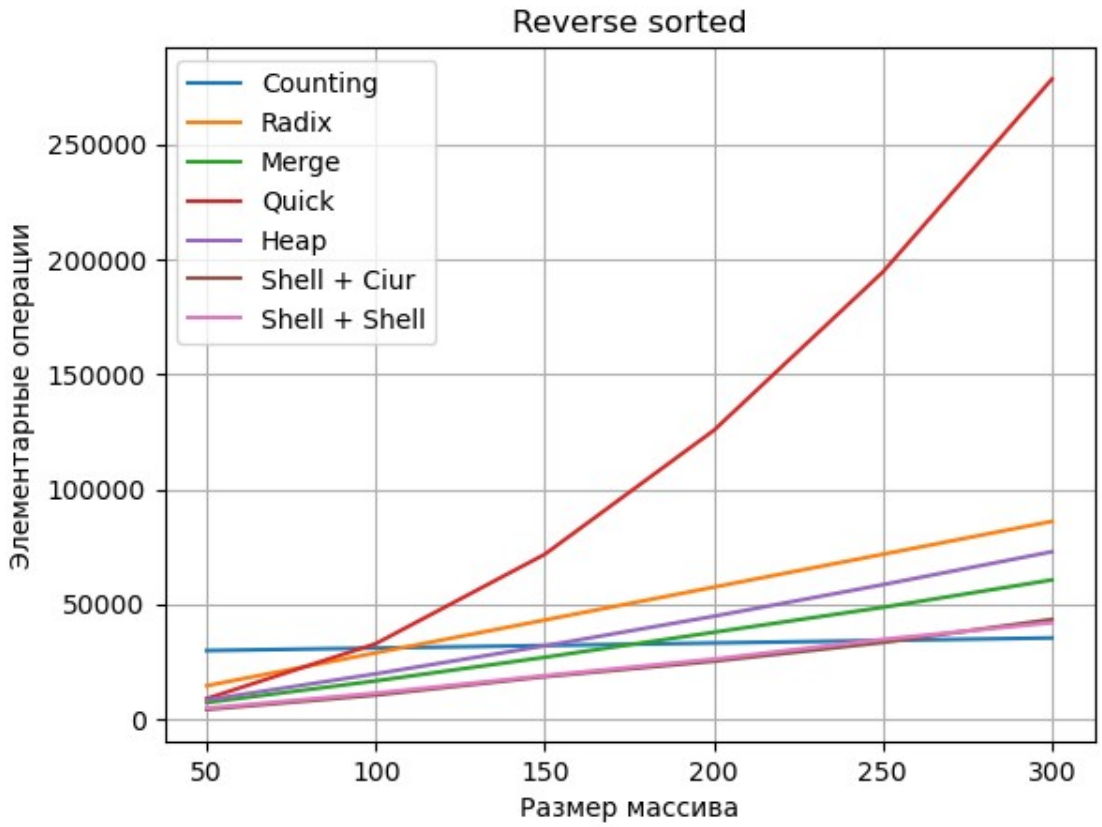

```
first_operations = open("C:\\Users\\pupki\\Desktop\\
first_operations.csv", "r")
reader1 = csv.reader(first_operations)
selection_sort = []
bubble_sort = []
bubble_1_sort = []
bubble_2_sort = []
insertion_sort = []
binary_insertion_sort = []
for row in reader1:
```

```
        selection_sort.append(int(row[3]))
        bubble_sort.append(int(row[7]))
        bubble_1_sort.append(int(row[11]))
        bubble_2_sort.append(int(row[15]))
        insertion_sort.append(int(row[19]))
        binary_insertion_sort.append(int(row[23]))
plt.plot(first_scale, selection_sort, label = 'Selection')
plt.plot(first_scale, bubble_sort, label = 'Bubble')
plt.plot(first_scale, bubble_1_sort, label = 'Bubble + Aiverson')
plt.plot(first_scale, bubble_2_sort, label = 'Bubble + Aiverson x 2')
plt.plot(first_scale, insertion_sort, label = 'Insertion')
plt.plot(first_scale, binary_insertion_sort, label = 'Binary
Insertion')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Reverse sorted')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```



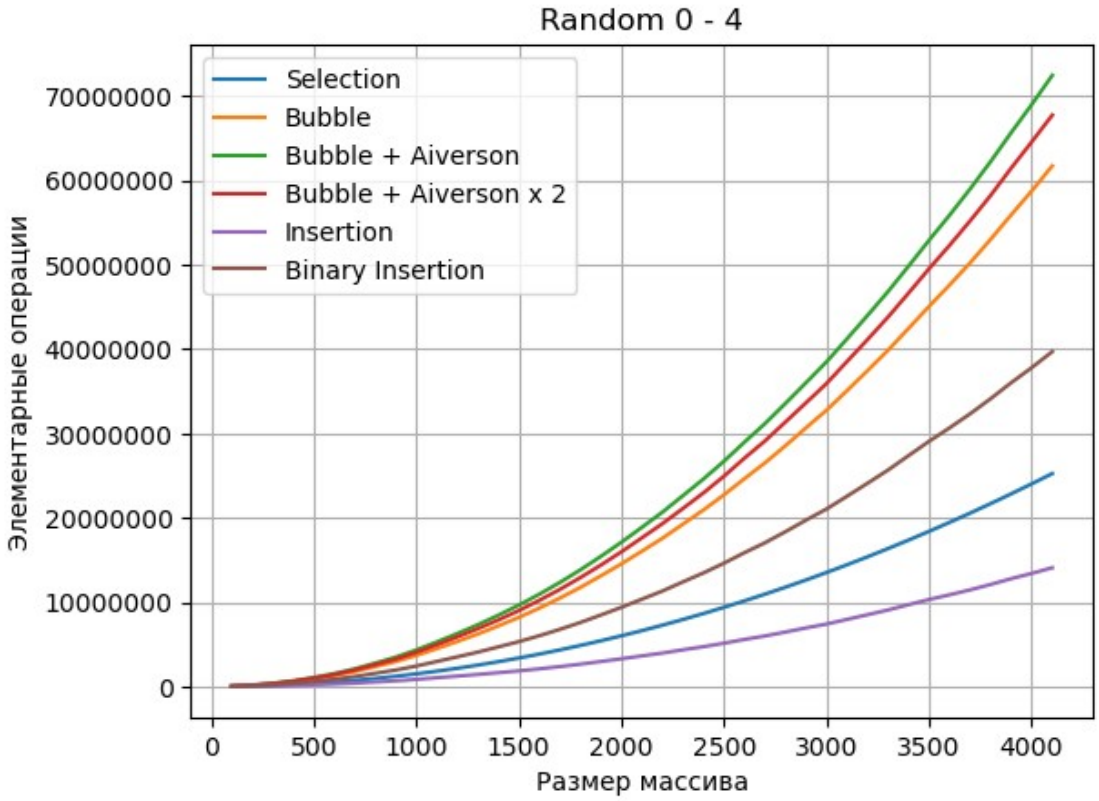

```
first_operations = open("C:\\Users\\pupki\\Desktop\\
first_operations.csv", "r")
reader1 = csv.reader(first_operations)
counting_sort = []
```

```
radix_sort = []
merge_sort = []
quick_sort = []
heap_sort = []
shell_1_sort = []
shell_2_sort = []
for row in reader1:
    counting_sort.append(int(row[27]))
    radix_sort.append(int(row[31]))
    merge_sort.append(int(row[35]))
    quick_sort.append(int(row[39]))
    heap_sort.append(int(row[43]))
    shell_1_sort.append(int(row[47]))
    shell_2_sort.append(int(row[51]))
plt.plot(first_scale, counting_sort, label = 'Counting')
plt.plot(first_scale, radix_sort, label = 'Radix')
plt.plot(first_scale, merge_sort, label = 'Merge')
plt.plot(first_scale, quick_sort, label = 'Quick')
plt.plot(first_scale, heap_sort, label = 'Heap')
plt.plot(first_scale, shell_1_sort, label = 'Shell + Ciur')
plt.plot(first_scale, shell_2_sort, label = 'Shell + Shell')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Reverse sorted')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```



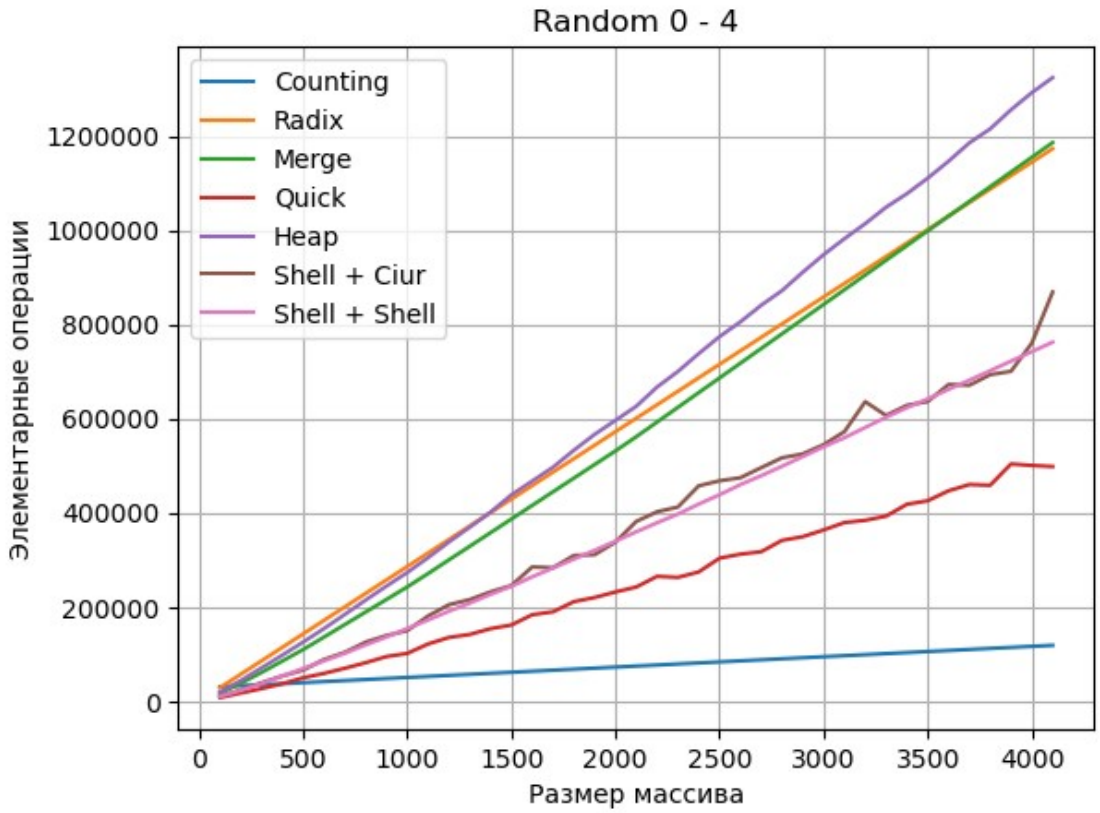

```
second_operations = open("C:\\Users\\pupki\\Desktop\\
second_operations.csv", "r")
reader2 = csv.reader(second_operations)
selection_sort = []
bubble_sort = []
bubble_1_sort = []
bubble_2_sort = []
insertion_sort = []
binary_insertion_sort = []
for row in reader2:
    selection_sort.append(int(row[0]))
    bubble_sort.append(int(row[4]))
    bubble_1_sort.append(int(row[8]))
    bubble_2_sort.append(int(row[12]))
    insertion_sort.append(int(row[16]))
    binary_insertion_sort.append(int(row[20]))
plt.plot(second_scale, selection_sort, label = 'Selection')
plt.plot(second_scale, bubble_sort, label = 'Bubble')
plt.plot(second_scale, bubble_1_sort, label = 'Bubble + Aiverson')
plt.plot(second_scale, bubble_2_sort, label = 'Bubble + Aiverson x 2')
plt.plot(second_scale, insertion_sort, label = 'Insertion')
plt.plot(second_scale, binary_insertion_sort, label = 'Binary
Insertion')
plt.grid(True)
plt.xlabel('Размер массива')
```

```
plt.ylabel('Элементарные операции')
plt.title('Random 0 - 4')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```



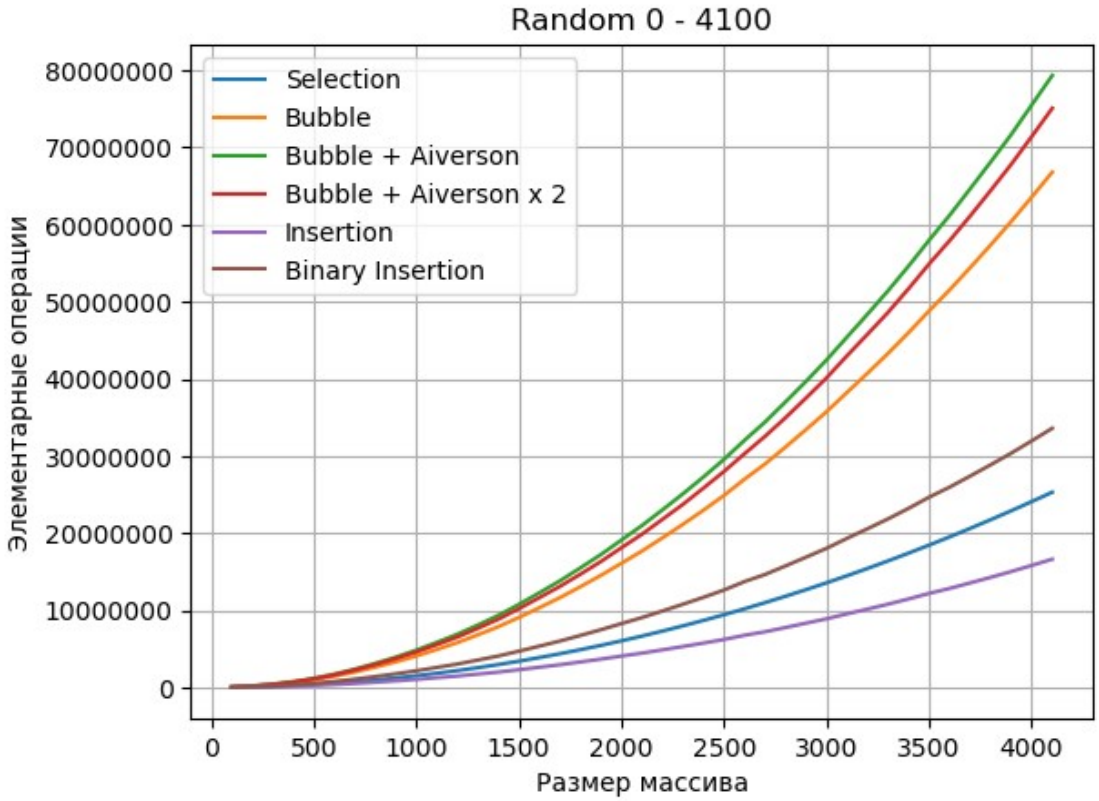

```
second_operations = open("C:\\Users\\pupki\\Desktop\\
second_operations.csv", "r")
reader2 = csv.reader(second_operations)
counting_sort = []
radix_sort = []
merge_sort = []
quick_sort = []
heap_sort = []
shell_1_sort = []
shell_2_sort = []
for row in reader2:
    counting_sort.append(int(row[24]))
    radix_sort.append(int(row[28]))
    merge_sort.append(int(row[32]))
    quick_sort.append(int(row[36]))
    heap_sort.append(int(row[40]))
    shell_1_sort.append(int(row[44]))
    shell_2_sort.append(int(row[48]))
plt.plot(second_scale, counting_sort, label = 'Counting')
plt.plot(second_scale, radix_sort, label = 'Radix')
```

```
plt.plot(second_scale, merge_sort, label = 'Merge')
plt.plot(second_scale, quick_sort, label = 'Quick')
plt.plot(second_scale, heap_sort, label = 'Heap')
plt.plot(second_scale, shell_1_sort, label = 'Shell + Ciur')
plt.plot(second_scale, shell_2_sort, label = 'Shell + Shell')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Random 0 - 4')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```



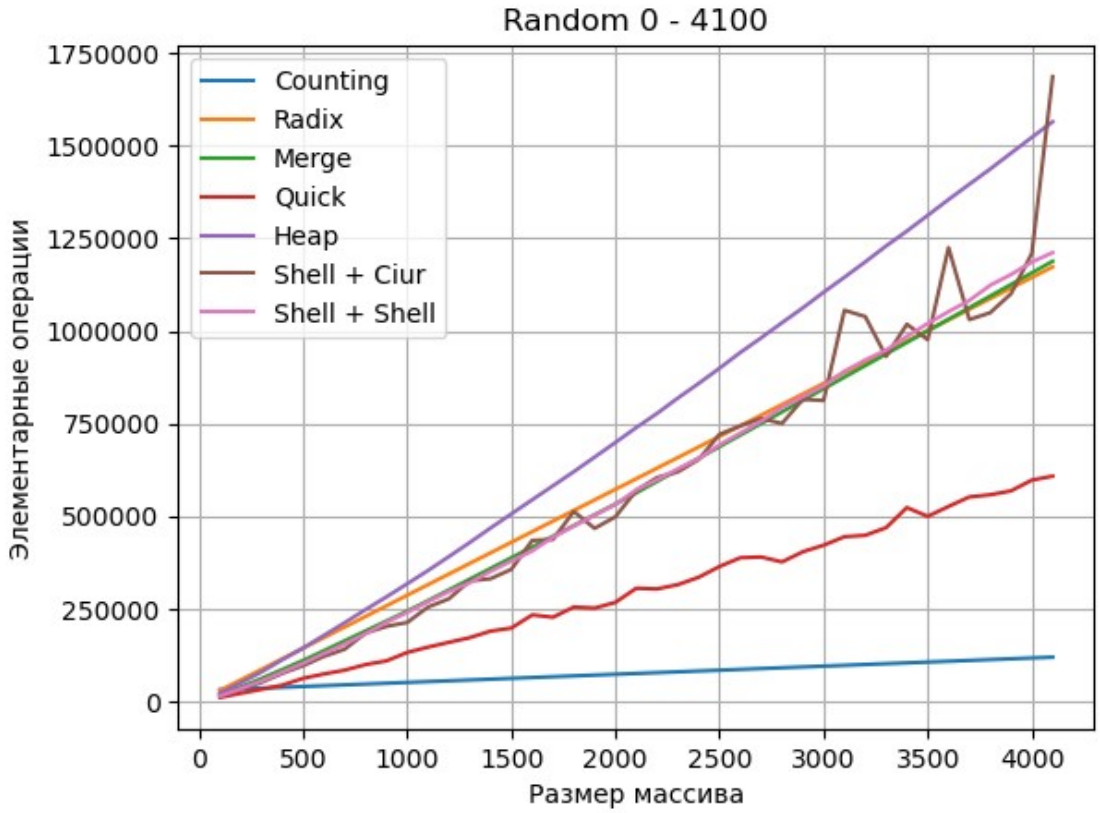

```
second_operations = open("C:\\Users\\pupki\\Desktop\\
second_operations.csv", "r")
reader2 = csv.reader(second_operations)
selection_sort = []
bubble_sort = []
bubble_1_sort = []
bubble_2_sort = []
insertion_sort = []
binary_insertion_sort = []
for row in reader2:
    selection_sort.append(int(row[1]))
    bubble_sort.append(int(row[5]))
```

```
    bubble_1_sort.append(int(row[9]))
    bubble_2_sort.append(int(row[13]))
    insertion_sort.append(int(row[17]))
    binary_insertion_sort.append(int(row[21]))
plt.plot(second_scale, selection_sort, label = 'Selection')
plt.plot(second_scale, bubble_sort, label = 'Bubble')
plt.plot(second_scale, bubble_1_sort, label = 'Bubble + Aiverson')
plt.plot(second_scale, bubble_2_sort, label = 'Bubble + Aiverson x 2')
plt.plot(second_scale, insertion_sort, label = 'Insertion')
plt.plot(second_scale, binary_insertion_sort, label = 'Binary
Insertion')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Random 0 - 4100')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```

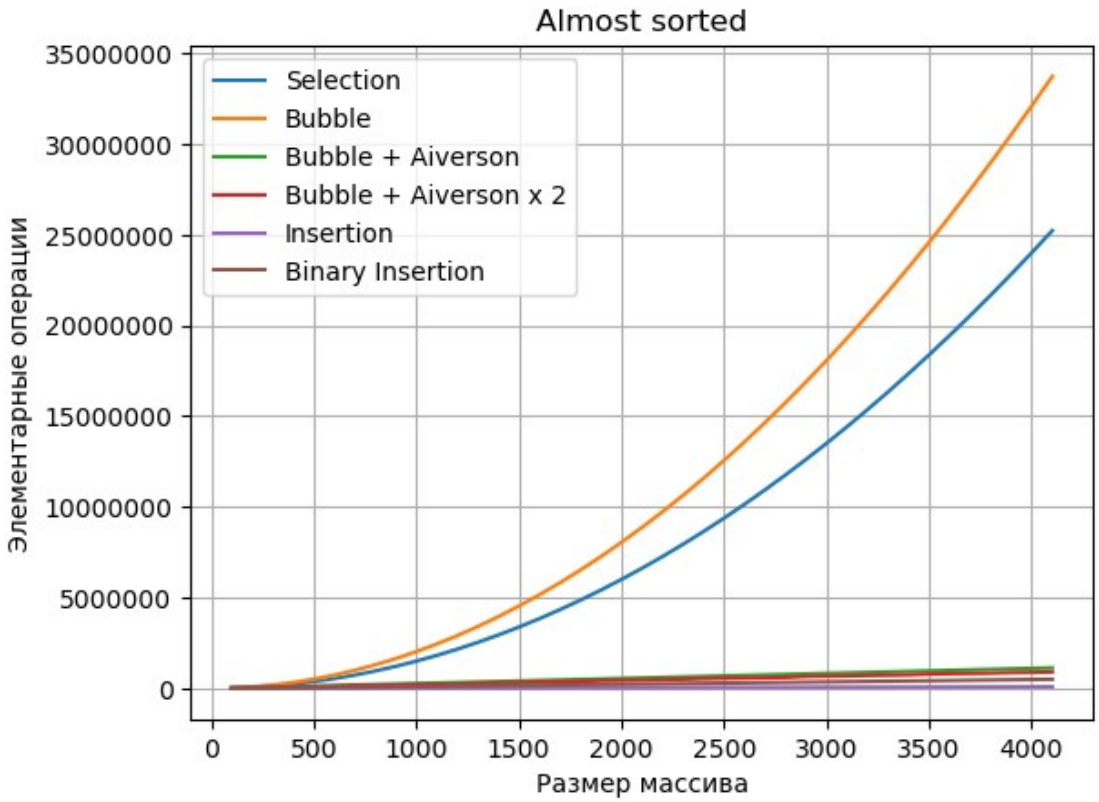



```
second_operations = open("C:\\Users\\pupki\\Desktop\\
second_operations.csv", "r")
reader2 = csv.reader(second_operations)
counting_sort = []
radix_sort = []
merge_sort = []
quick_sort = []
```

```
heap_sort = []
shell_1_sort = []
shell_2_sort = []
for row in reader2:
    counting_sort.append(int(row[25]))
    radix_sort.append(int(row[29]))
    merge_sort.append(int(row[33]))
    quick_sort.append(int(row[37]))
    heap_sort.append(int(row[41]))
    shell_1_sort.append(int(row[45]))
    shell_2_sort.append(int(row[49]))
plt.plot(second_scale, counting_sort, label = 'Counting')
plt.plot(second_scale, radix_sort, label = 'Radix')
plt.plot(second_scale, merge_sort, label = 'Merge')
plt.plot(second_scale, quick_sort, label = 'Quick')
plt.plot(second_scale, heap_sort, label = 'Heap')
plt.plot(second_scale, shell_1_sort, label = 'Shell + Ciur')
plt.plot(second_scale, shell_2_sort, label = 'Shell + Shell')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Random 0 - 4100')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```

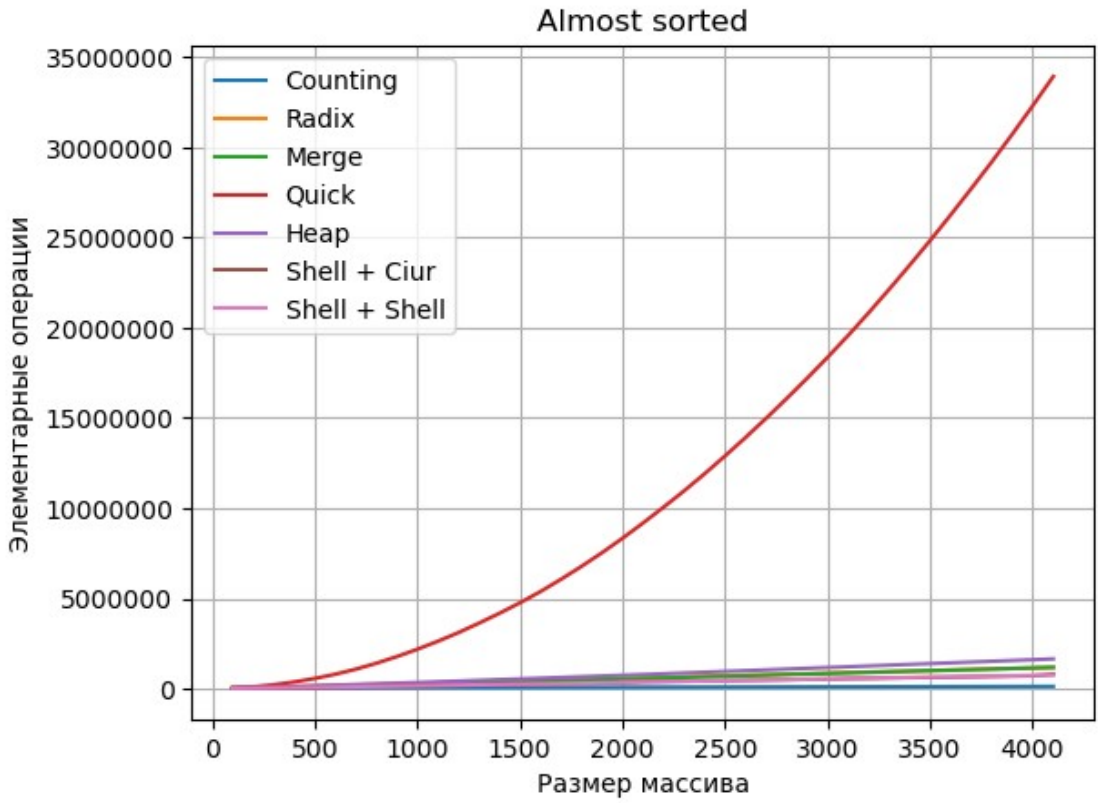


```
second_operations = open("C:\\Users\\pupki\\Desktop\\
second_operations.csv", "r")
reader2 = csv.reader(second_operations)
selection_sort = []
bubble_sort = []
bubble_1_sort = []
bubble_2_sort = []
insertion_sort = []
binary_insertion_sort = []
for row in reader2:
    selection_sort.append(int(row[2]))
    bubble_sort.append(int(row[6]))
    bubble_1_sort.append(int(row[10]))
    bubble_2_sort.append(int(row[14]))
    insertion_sort.append(int(row[18]))
    binary_insertion_sort.append(int(row[22]))
plt.plot(second_scale, selection_sort, label = 'Selection')
plt.plot(second_scale, bubble_sort, label = 'Bubble')
plt.plot(second_scale, bubble_1_sort, label = 'Bubble + Aiverson')
plt.plot(second_scale, bubble_2_sort, label = 'Bubble + Aiverson x 2')
plt.plot(second_scale, insertion_sort, label = 'Insertion')
plt.plot(second_scale, binary_insertion_sort, label = 'Binary
Insertion')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Almost sorted')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```



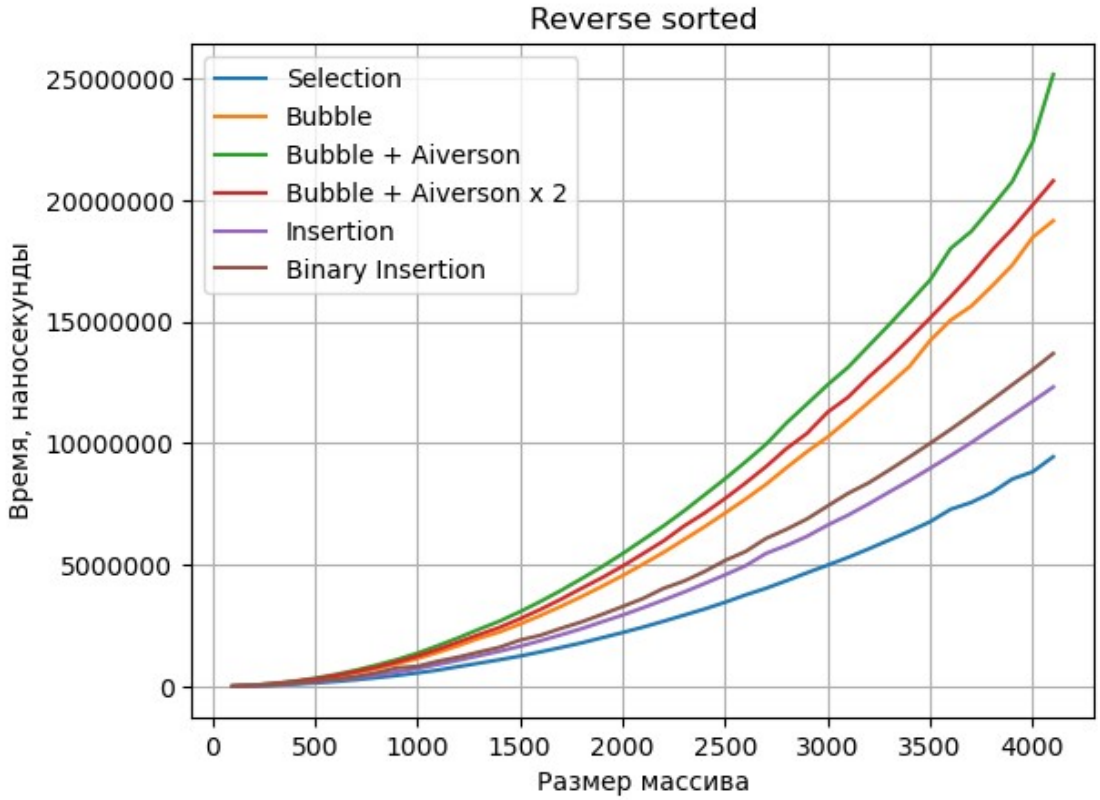

```
second_operations = open("C:\\Users\\pupki\\Desktop\\
second_operations.csv", "r")
reader2 = csv.reader(second_operations)
counting_sort = []
radix_sort = []
merge_sort = []
quick_sort = []
heap_sort = []
shell_1_sort = []
shell_2_sort = []
for row in reader2:
    counting_sort.append(int(row[26]))
    radix_sort.append(int(row[30]))
    merge_sort.append(int(row[34]))
    quick_sort.append(int(row[38]))
    heap_sort.append(int(row[42]))
    shell_1_sort.append(int(row[46]))
    shell_2_sort.append(int(row[50]))
plt.plot(second_scale, counting_sort, label = 'Counting')
plt.plot(second_scale, radix_sort, label = 'Radix')
plt.plot(second_scale, merge_sort, label = 'Merge')
plt.plot(second_scale, quick_sort, label = 'Quick')
plt.plot(second_scale, heap_sort, label = 'Heap')
plt.plot(second_scale, shell_1_sort, label = 'Shell + Ciur')
plt.plot(second_scale, shell_2_sort, label = 'Shell + Shell')
```

```
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Almost sorted')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




```
second_time = open("C:\\Users\\pupki\\Desktop\\second_time.csv", "r")
reader2 = csv.reader(second_time)
selection_sort = []
bubble_sort = []
bubble_1_sort = []
bubble_2_sort = []
insertion_sort = []
binary_insertion_sort = []
for row in reader2:
    selection_sort.append(int(row[3]))
    bubble_sort.append(int(row[7]))
    bubble_1_sort.append(int(row[11]))
    bubble_2_sort.append(int(row[15]))
    insertion_sort.append(int(row[19]))
    binary_insertion_sort.append(int(row[23]))
plt.plot(second_scale, selection_sort, label = 'Selection')
plt.plot(second_scale, bubble_sort, label = 'Bubble')
plt.plot(second_scale, bubble_1_sort, label = 'Bubble + Aiverson')
```

```
plt.plot(second_scale, bubble_2_sort, label = 'Bubble + Aiverson x 2')
plt.plot(second_scale, insertion_sort, label = 'Insertion')
plt.plot(second_scale, binary_insertion_sort, label = 'Binary
Insertion')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Время, наносекунды')
plt.title('Reverse sorted')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```




```
second_operations = open("C:\\Users\\pupki\\Desktop\\
second_operations.csv", "r")
reader2 = csv.reader(second_operations)
counting_sort = []
radix_sort = []
merge_sort = []
quick_sort = []
heap_sort = []
shell_1_sort = []
shell_2_sort = []
for row in reader2:
    counting_sort.append(int(row[27]))
    radix_sort.append(int(row[31]))
    merge_sort.append(int(row[35]))
```

```
        quick_sort.append(int(row[39]))
        heap_sort.append(int(row[43]))
        shell_1_sort.append(int(row[47]))
        shell_2_sort.append(int(row[51]))
plt.plot(second_scale, counting_sort, label = 'Counting')
plt.plot(second_scale, radix_sort, label = 'Radix')
plt.plot(second_scale, merge_sort, label = 'Merge')
plt.plot(second_scale, quick_sort, label = 'Quick')
plt.plot(second_scale, heap_sort, label = 'Heap')
plt.plot(second_scale, shell_1_sort, label = 'Shell + Ciur')
plt.plot(second_scale, shell_2_sort, label = 'Shell + Shell')
plt.grid(True)
plt.xlabel('Размер массива')
plt.ylabel('Элементарные операции')
plt.title('Reverse sorted')
plt.legend(loc = 'best')
plt.ticklabel_format(style='plain', axis='y')
plt.show()
```