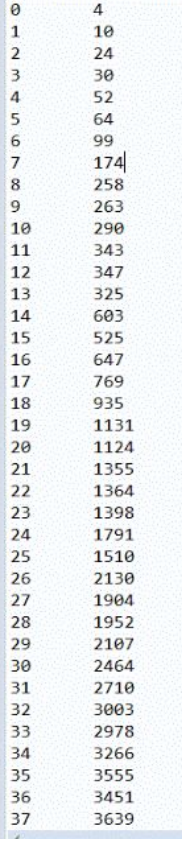
Quadratic Sorts

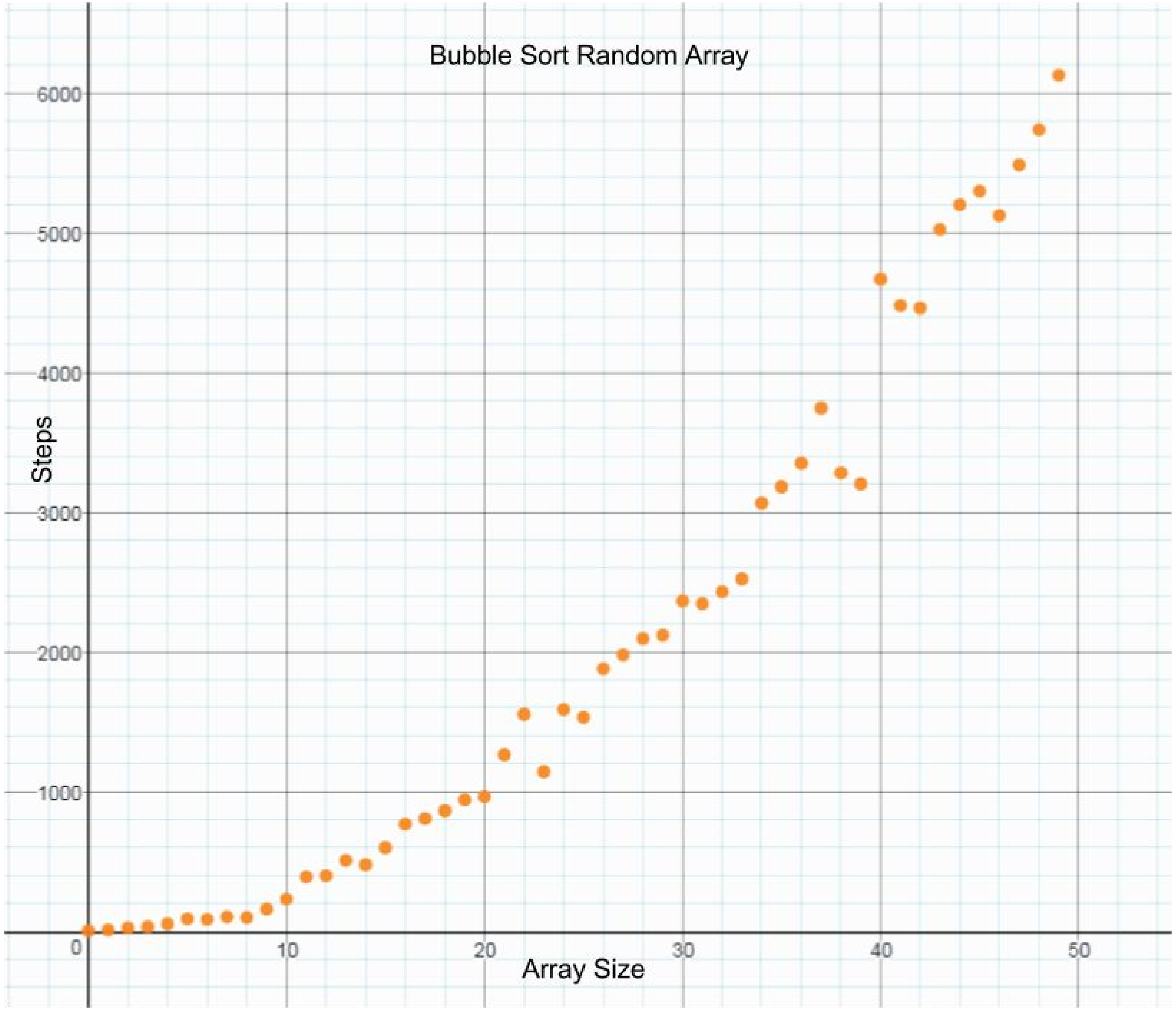
Mayaank Vadlamani & Kashif Peshimam

Random

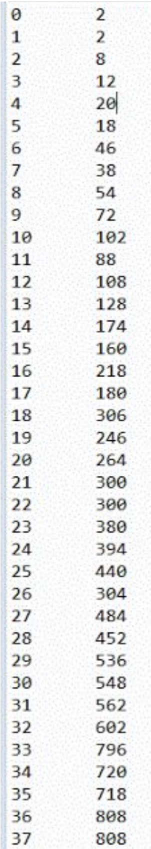
Array Size | Steps



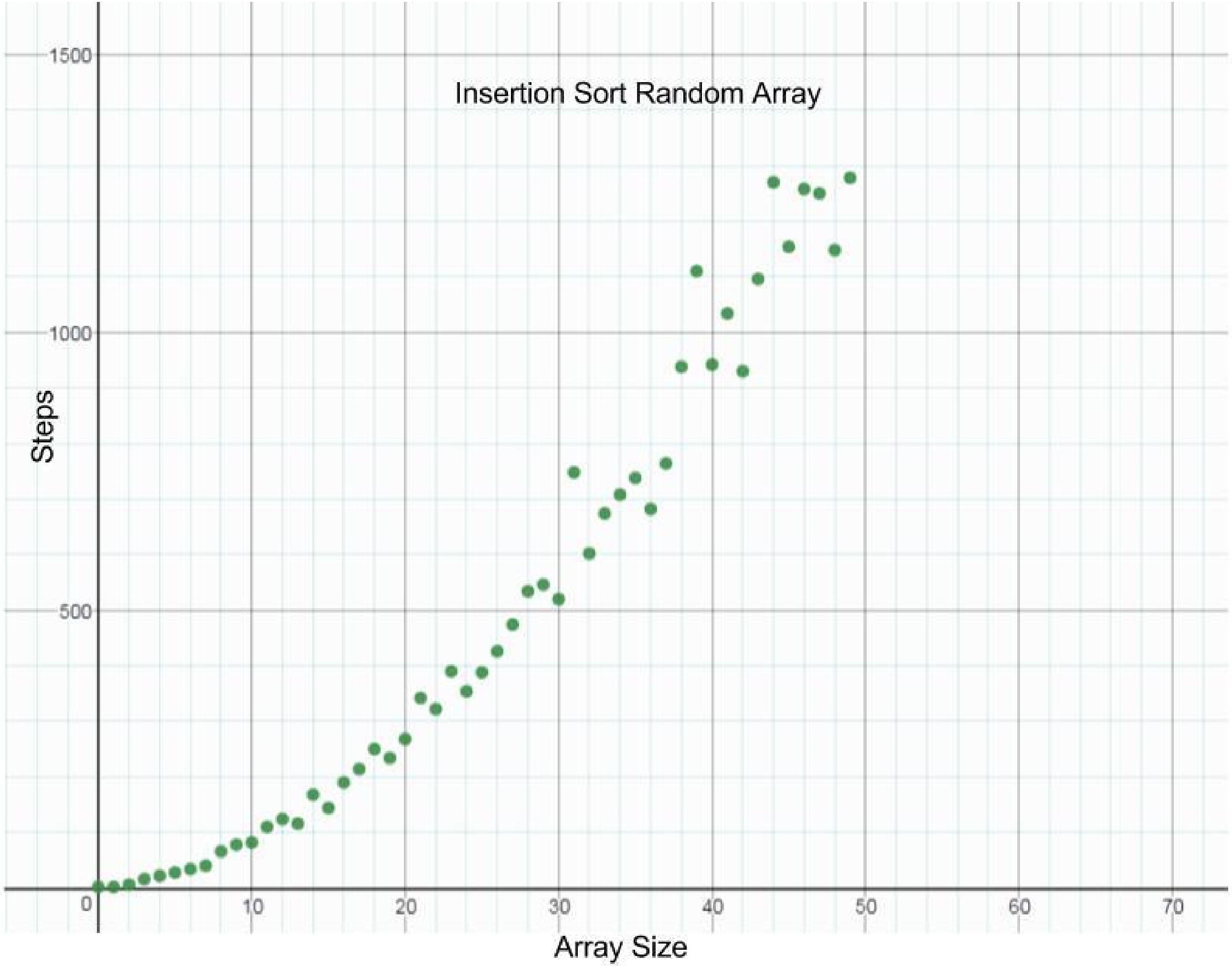
0	4
1	10
2	24
3	30
4	52
5	64
6	99
7	174
8	258
9	263
10	290
11	343
12	347
13	325
14	603
15	525
16	647
17	769
18	935
19	1131
20	1124
21	1355
22	1364
23	1398
24	1791
25	1510
26	2130
27	1904
28	1952
29	2107
30	2464
31	2710
32	3003
33	2978
34	3266
35	3555
36	3451
37	3639



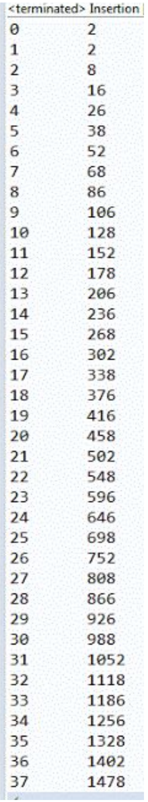
Array Size | Steps



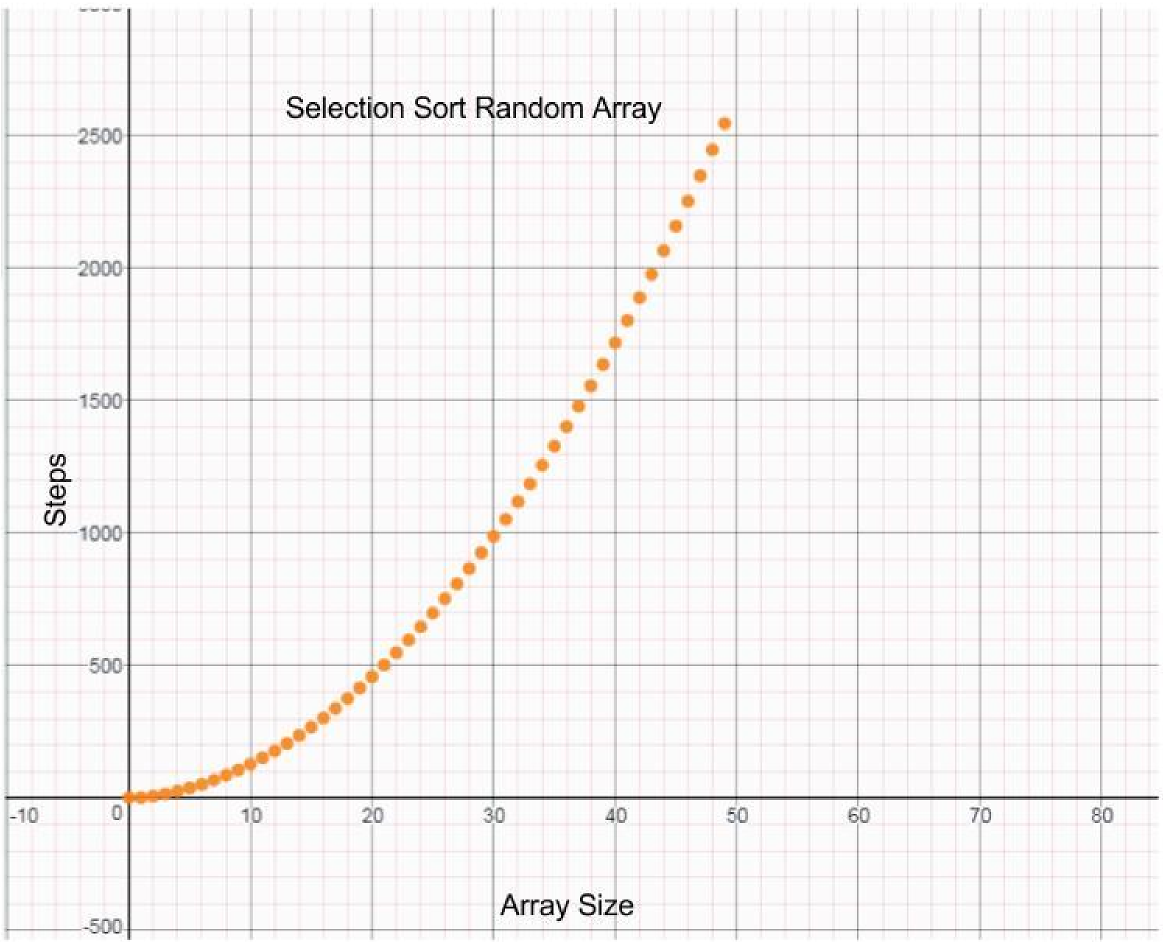
0	2
1	2
2	8
3	12
4	20
5	18
6	46
7	38
8	54
9	72
10	102
11	88
12	108
13	128
14	174
15	160
16	218
17	180
18	306
19	246
20	264
21	300
22	300
23	380
24	394
25	440
26	304
27	484
28	452
29	536
30	548
31	562
32	602
33	796
34	720
35	718
36	808
37	808



Array Size | Steps

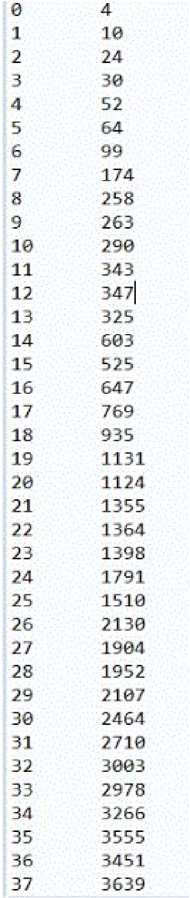


```
<terminated> Insertion
0          2
1          2
2          8
3          16
4          26
5          38
6          52
7          68
8          86
9          106
10         128
11         152
12         178
13         206
14         236
15         268
16         302
17         338
18         376
19         416
20         458
21         502
22         548
23         596
24         646
25         698
26         752
27         808
28         866
29         926
30         988
31         1052
32         1118
33         1186
34         1256
35         1328
36         1402
37         1478
```

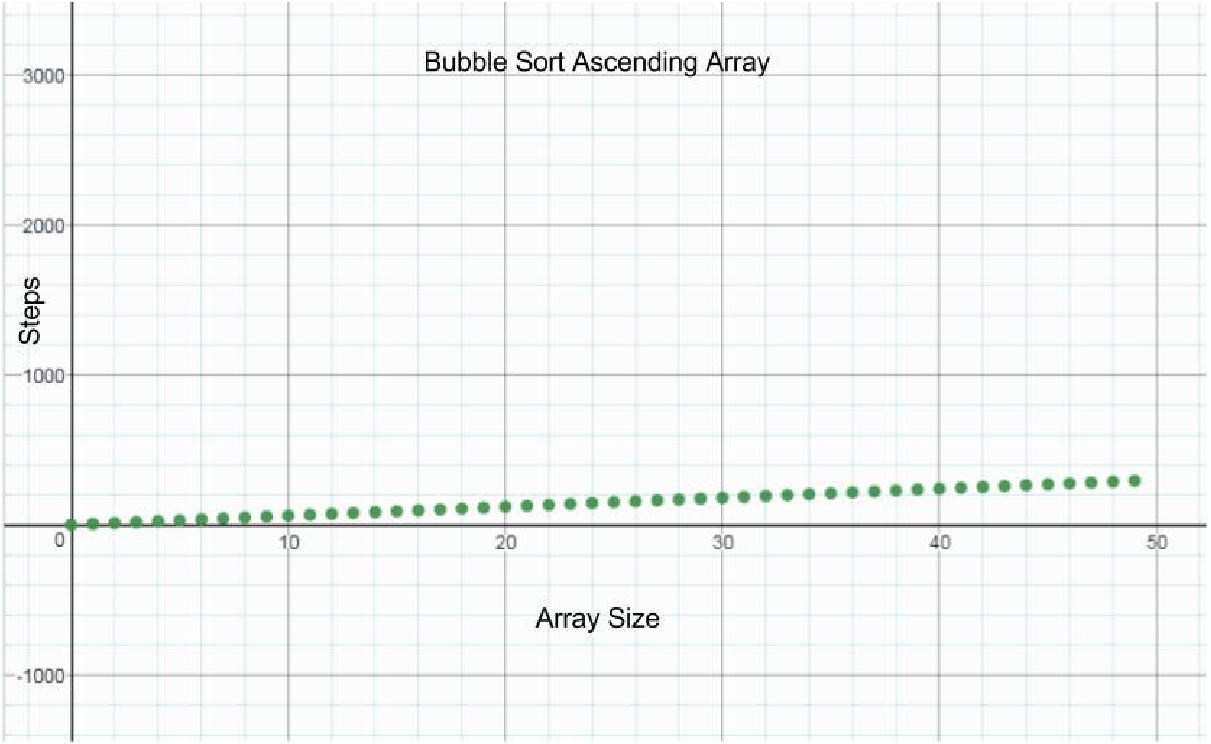


Ascending

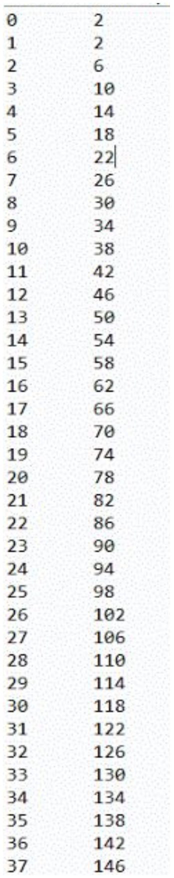
Array Size | Steps



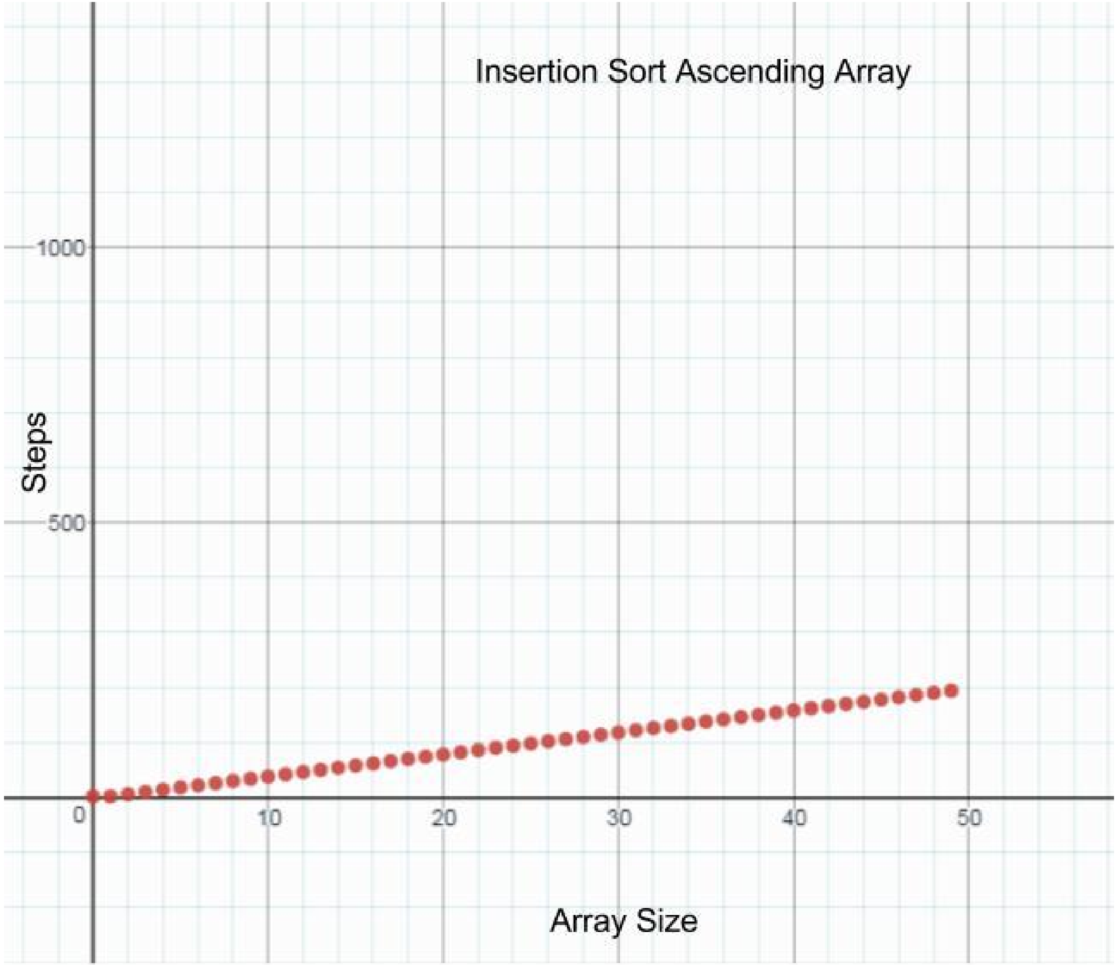
0	4
1	10
2	24
3	30
4	52
5	64
6	99
7	174
8	258
9	263
10	290
11	343
12	347
13	325
14	603
15	525
16	647
17	769
18	935
19	1131
20	1124
21	1355
22	1364
23	1398
24	1791
25	1510
26	2130
27	1904
28	1952
29	2107
30	2464
31	2710
32	3003
33	2978
34	3266
35	3555
36	3451
37	3639



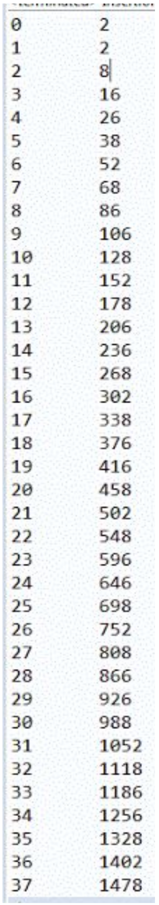
Array Size | Steps



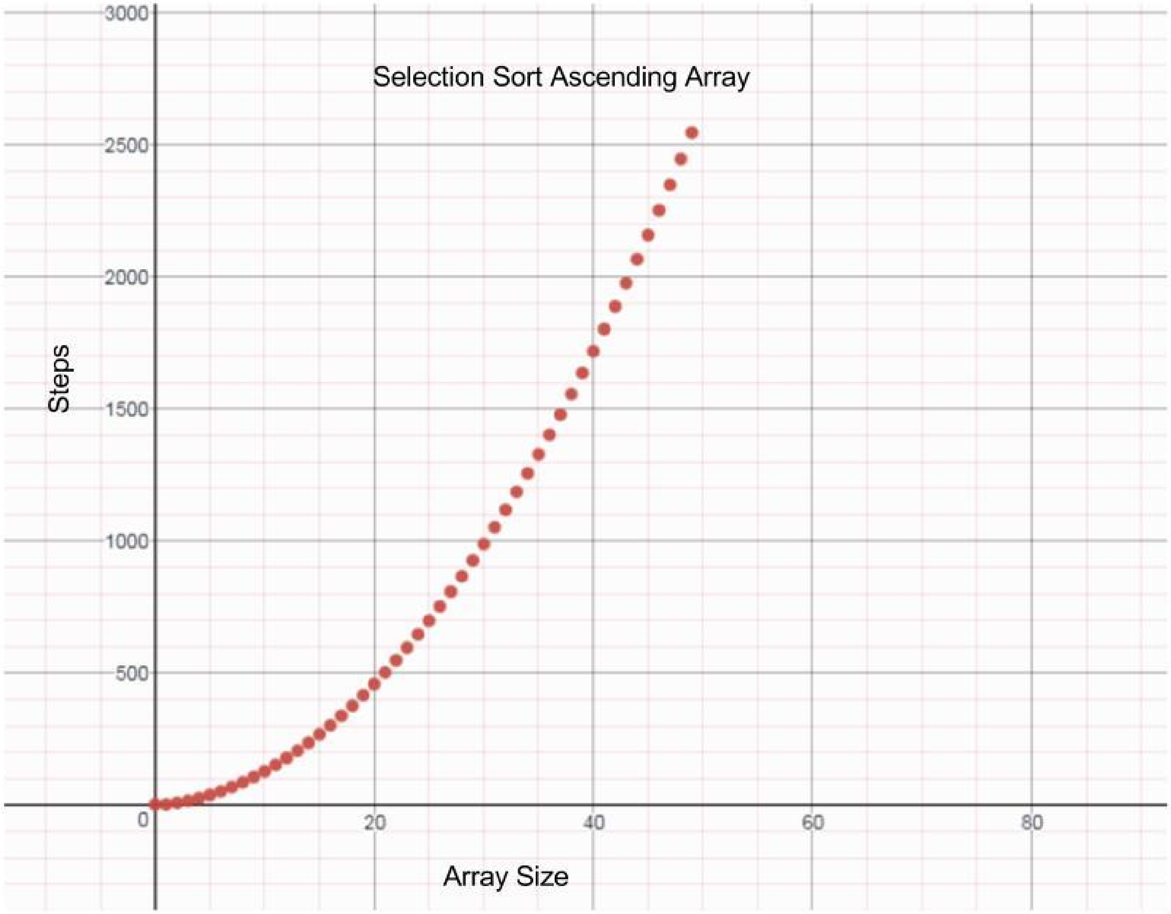
```
0         2
1         2
2         6
3         10
4         14
5         18
6         22
7         26
8         30
9         34
10        38
11        42
12        46
13        50
14        54
15        58
16        62
17        66
18        70
19        74
20        78
21        82
22        86
23        90
24        94
25        98
26        102
27        106
28        110
29        114
30        118
31        122
32        126
33        130
34        134
35        138
36        142
37        146
```



Array Size | Steps

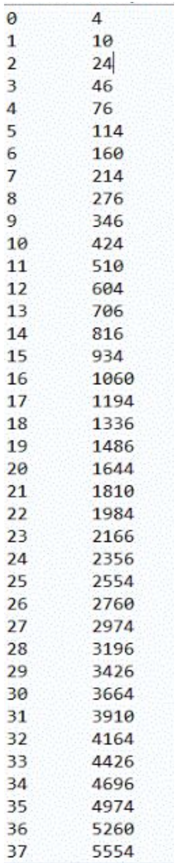


```
0           2
1           2
2           8
3           16
4           26
5           38
6           52
7           68
8           86
9           106
10          128
11          152
12          178
13          206
14          236
15          268
16          302
17          338
18          376
19          416
20          458
21          502
22          548
23          596
24          646
25          698
26          752
27          808
28          866
29          926
30          988
31          1052
32          1118
33          1186
34          1256
35          1328
36          1402
37          1478
```

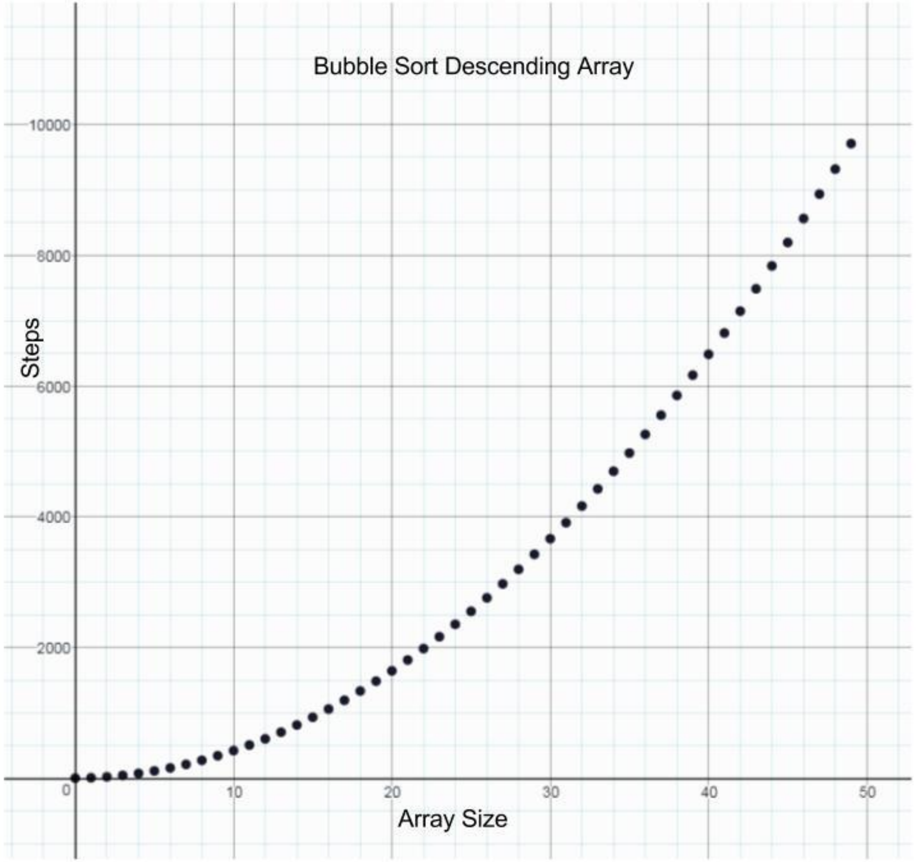


Descending

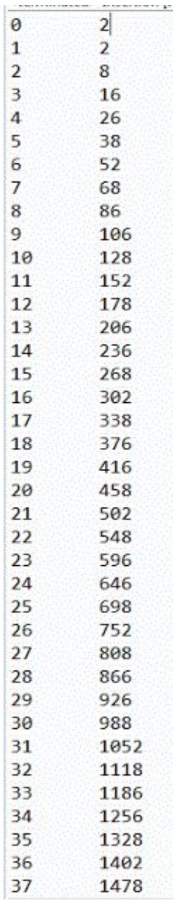
Array Size | Steps



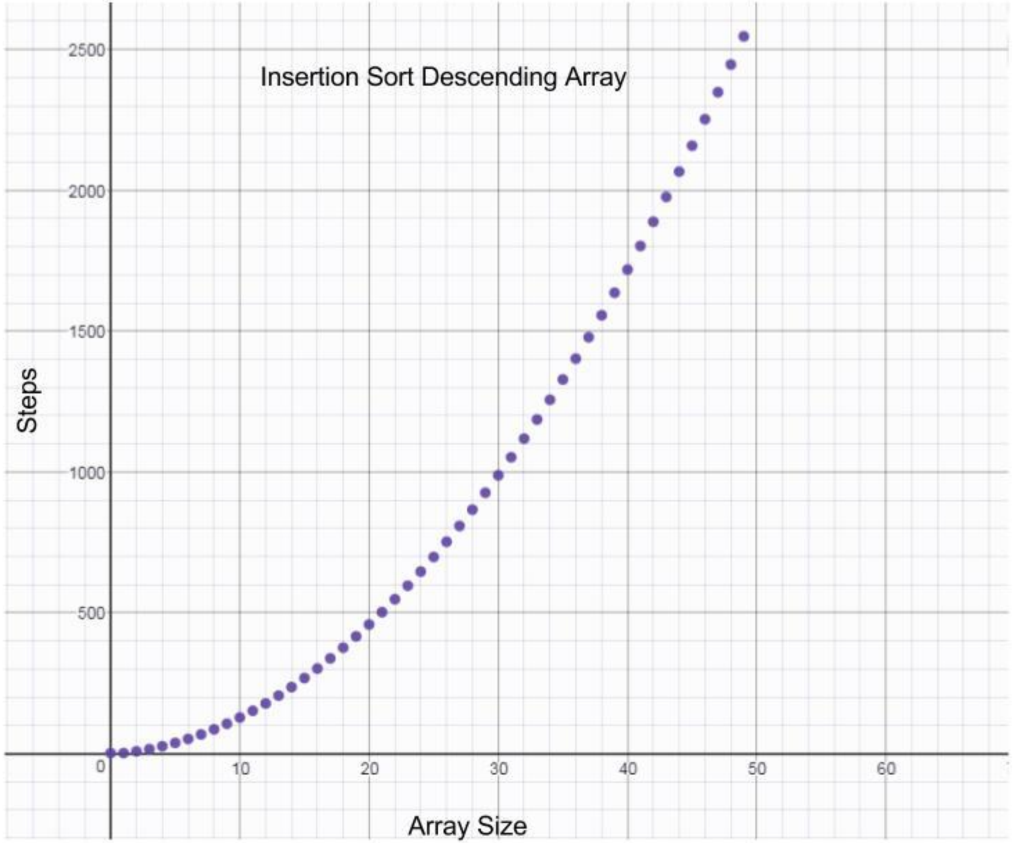
0	4
1	10
2	24
3	46
4	76
5	114
6	160
7	214
8	276
9	346
10	424
11	510
12	604
13	706
14	816
15	934
16	1060
17	1194
18	1336
19	1486
20	1644
21	1810
22	1984
23	2166
24	2356
25	2554
26	2760
27	2974
28	3196
29	3426
30	3664
31	3910
32	4164
33	4426
34	4696
35	4974
36	5260
37	5554



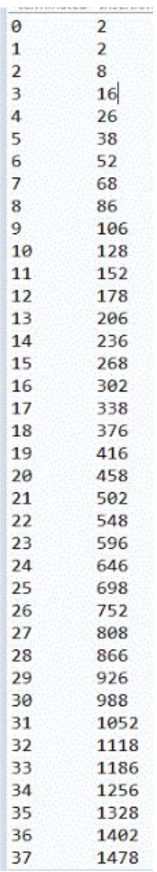
Array Size | Steps



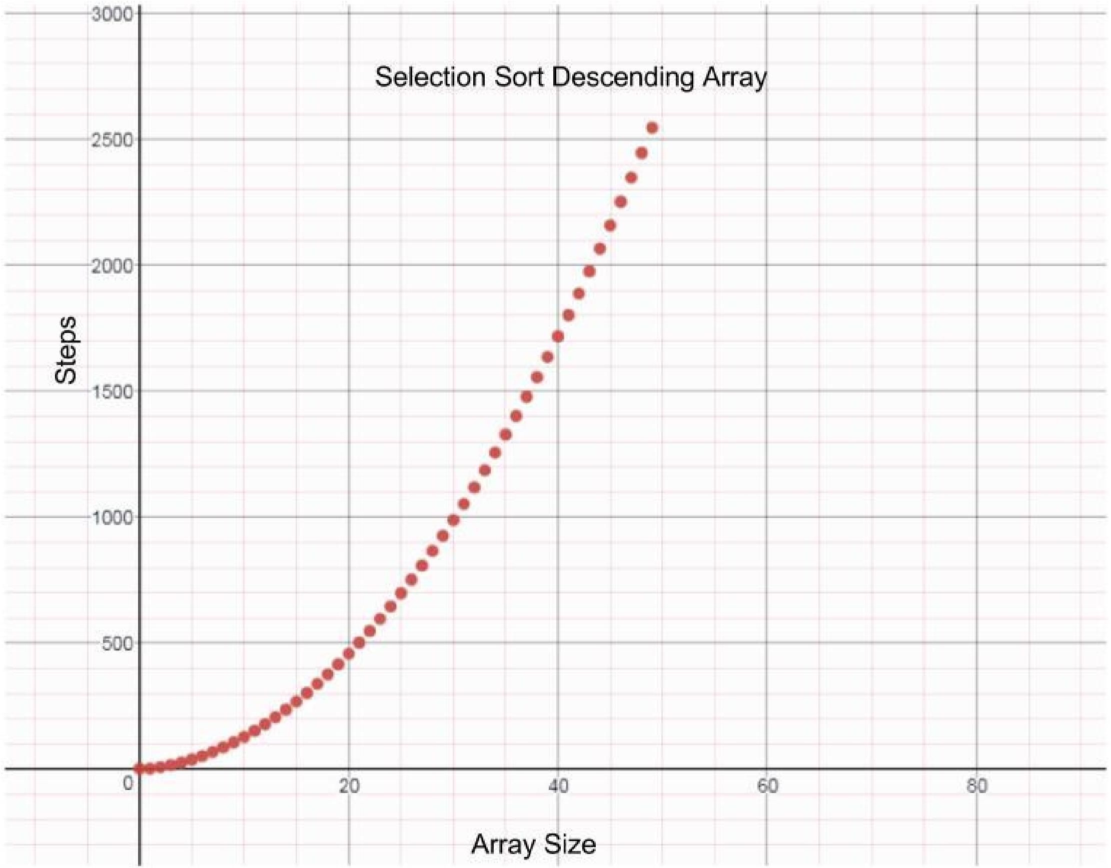
0	2
1	2
2	8
3	16
4	26
5	38
6	52
7	68
8	86
9	106
10	128
11	152
12	178
13	206
14	236
15	268
16	302
17	338
18	376
19	416
20	458
21	502
22	548
23	596
24	646
25	698
26	752
27	808
28	866
29	926
30	988
31	1052
32	1118
33	1186
34	1256
35	1328
36	1402
37	1478



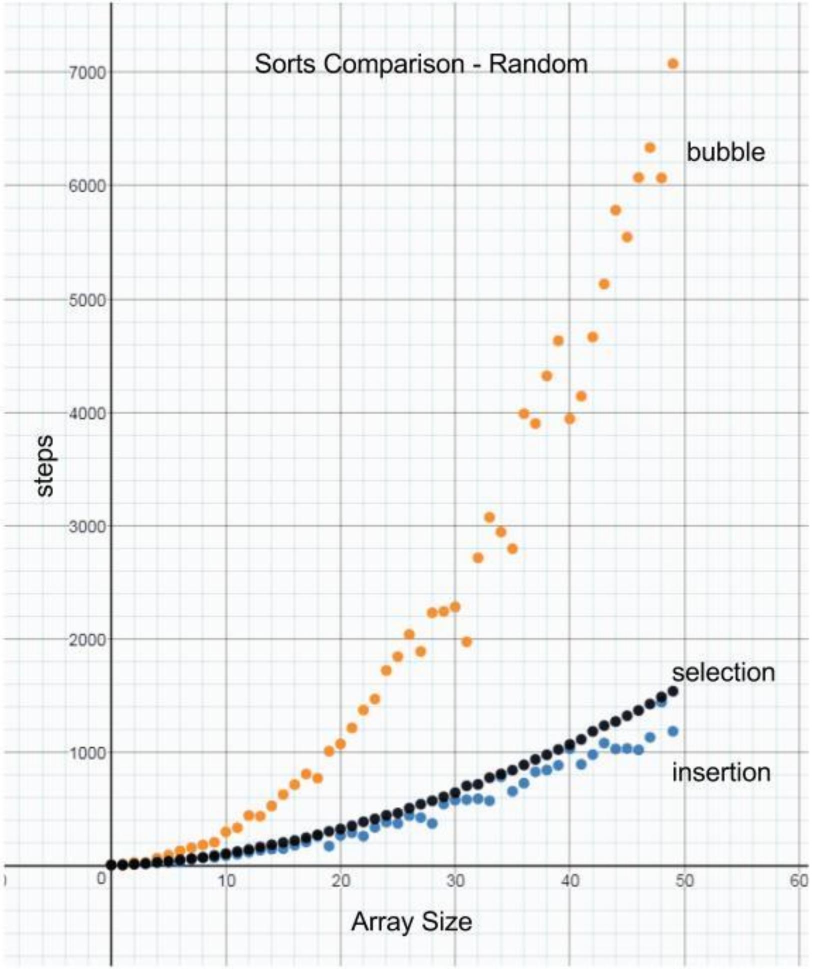
Array Size | Steps

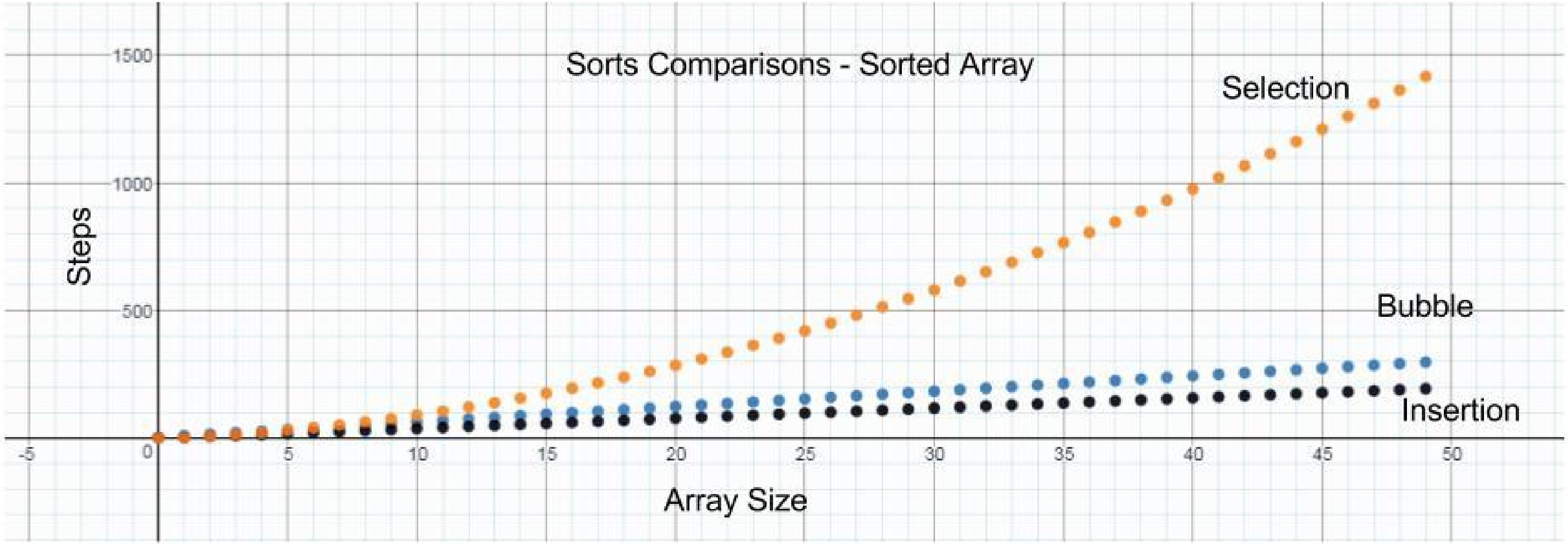


0	2
1	2
2	8
3	16
4	26
5	38
6	52
7	68
8	86
9	106
10	128
11	152
12	178
13	206
14	236
15	268
16	302
17	338
18	376
19	416
20	458
21	502
22	548
23	596
24	646
25	698
26	752
27	808
28	866
29	926
30	988
31	1052
32	1118
33	1186
34	1256
35	1328
36	1402
37	1478



All


Sorts Comparison - Random
bubble
selection
insertion
steps
Array Size
7000
6000
5000
4000
3000
2000
1000
0
10
20
30
40
50
60


Sorts Comparisons - Sorted Array
Selection
Bubble
Insertion
Steps
1500
1000
500
-5
0
5
10
15
20
25
30
35
40
45
50
Array Size

Explain why these are called quadratic sorts. O(n2)

- Number of comparisons increases as a quadratic relationship with array size
- Ex: Insertion sort
    - Outer loop runs N times
    - Inner loop runs N/2 times
    - N * N/2 = N^2

Which is the most efficient sort of a random array? Why?

- Insertion Sort
  - Insertion sort provides a O(n^2) worst case algorithm that adapts to O(n) tir data is nearly sorted.
  - requires less memory
    - Fewer comparisons

Which is the least efficient sort of a reverse ordered array? Why?

- Bubble Sort
  - Because of a large amount of possible swaps.
  - It compares every element to a lot of other elements. That slows it down
  - Worst case: O(n^2)
  - Average Case: O(n^2)
  - Best Case: O(n)

Which of these sort situations will produce a linear relationship O(n). Why?

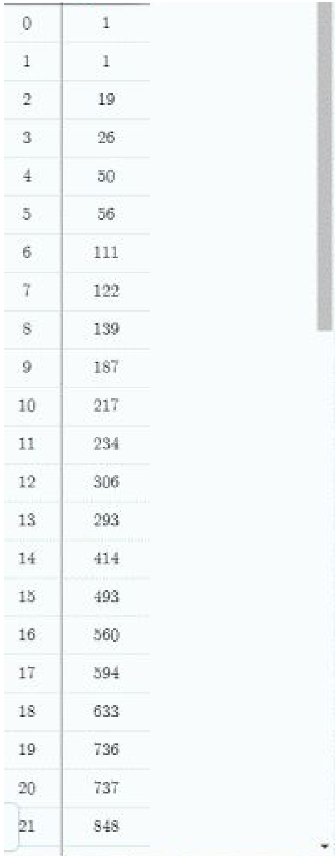
- Bubble and Insertion Sort with ascending ordered array
    - The array is already sorted
    - So the methods only pass through once (O(n))
    - do not swap anything, everything is already in order.

Part II

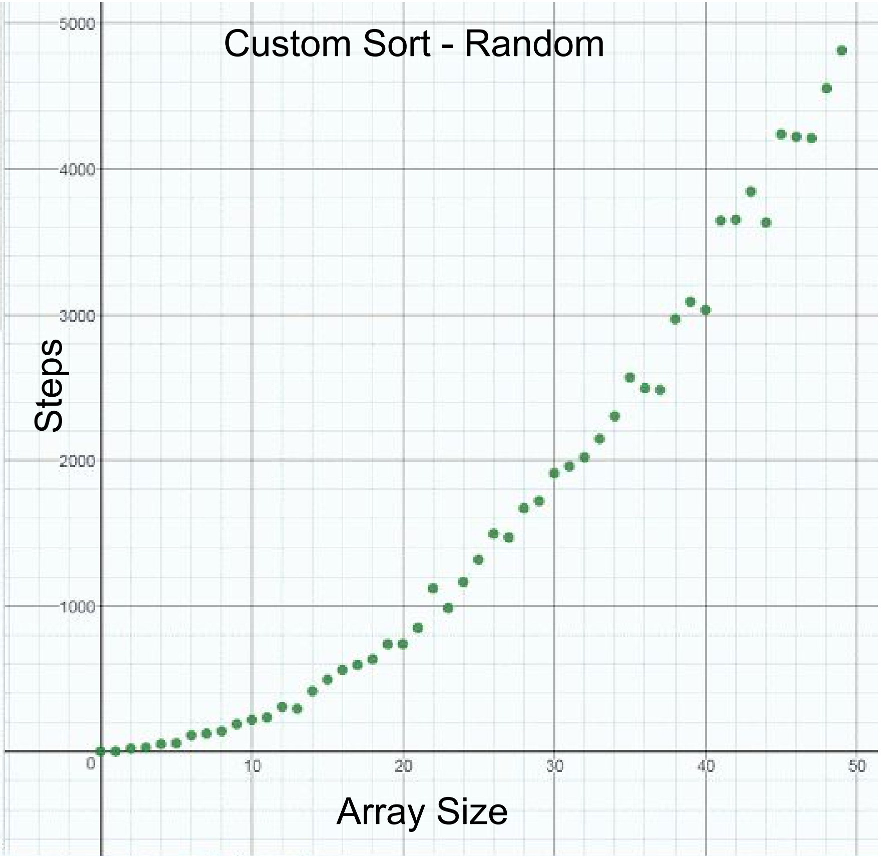
** Explained in CustomSort.java

Part II: Random

Array Size | Steps

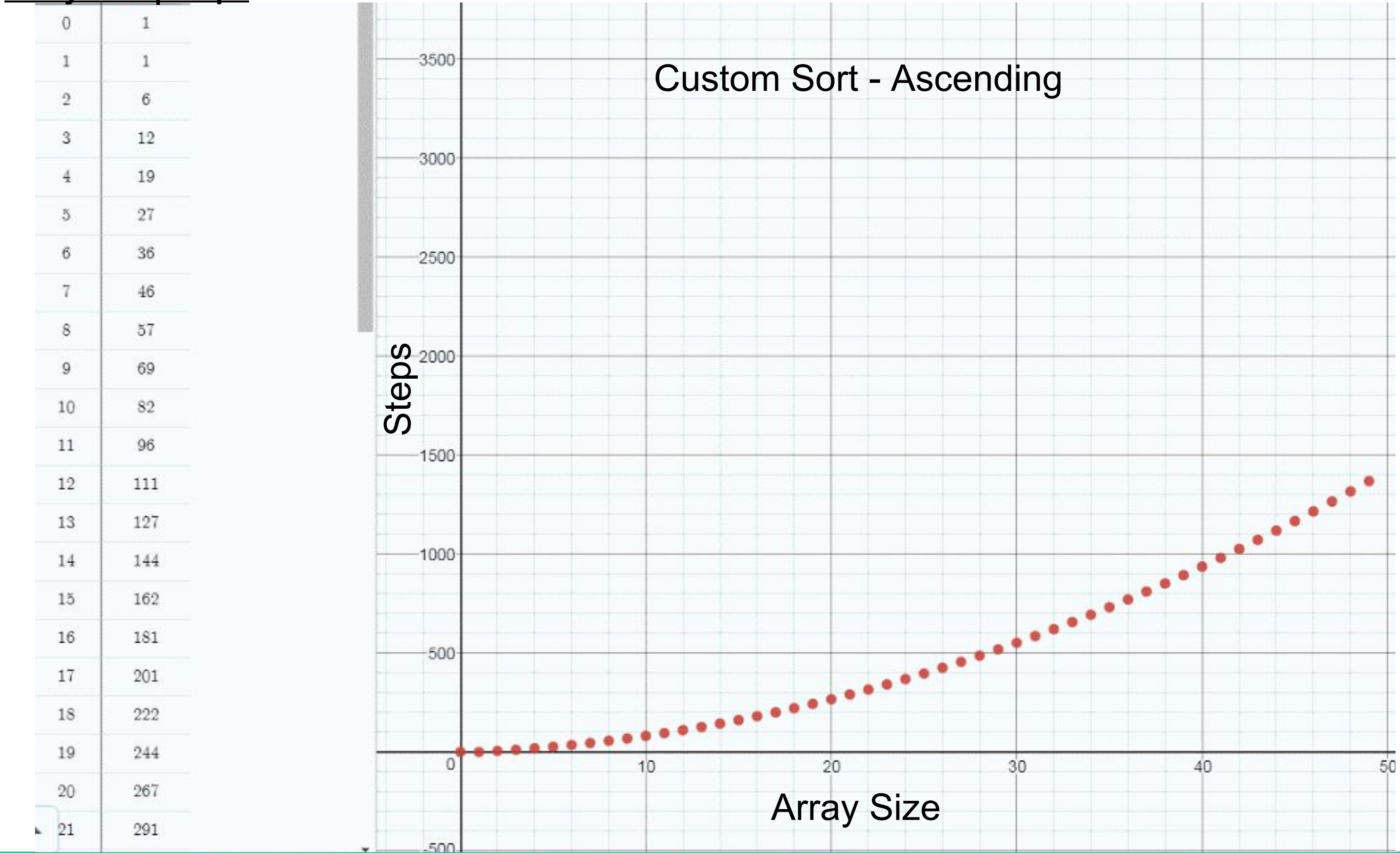


0	1
1	1
2	19
3	26
4	50
5	56
6	111
7	122
8	139
9	187
10	217
11	234
12	306
13	293
14	414
15	493
16	560
17	594
18	633
19	736
20	737
21	848



Part II: Ascending

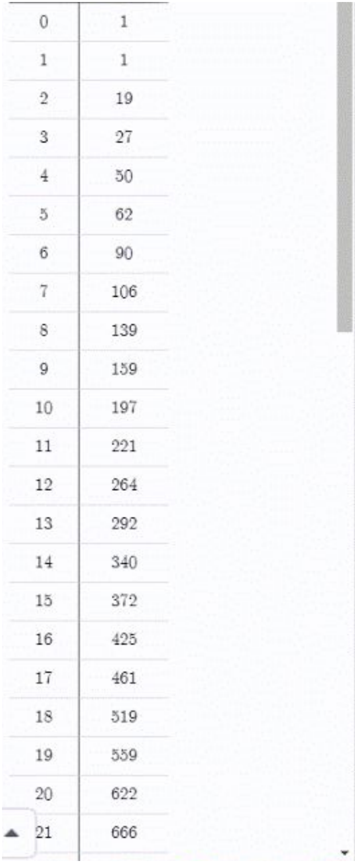
Array Size | Steps



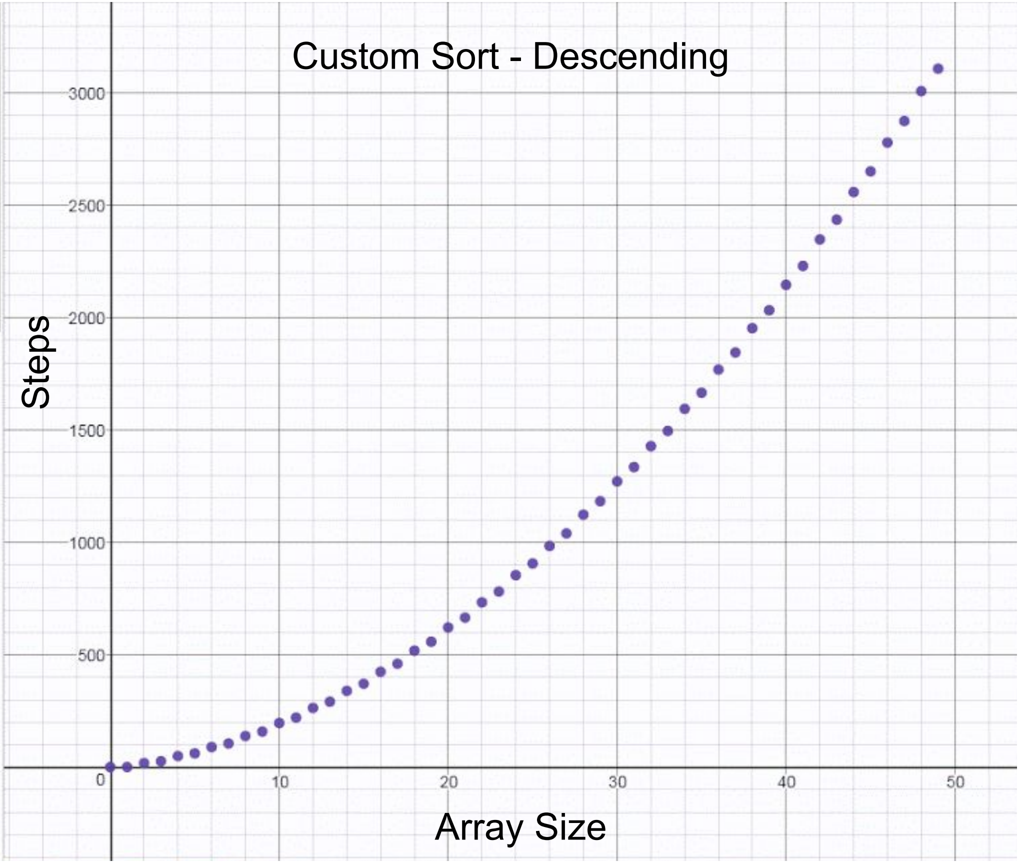
0	1
1	1
2	6
3	12
4	19
5	27
6	36
7	46
8	57
9	69
10	82
11	96
12	111
13	127
14	144
15	162
16	181
17	201
18	222
19	244
20	267
21	291

Part II: Descending

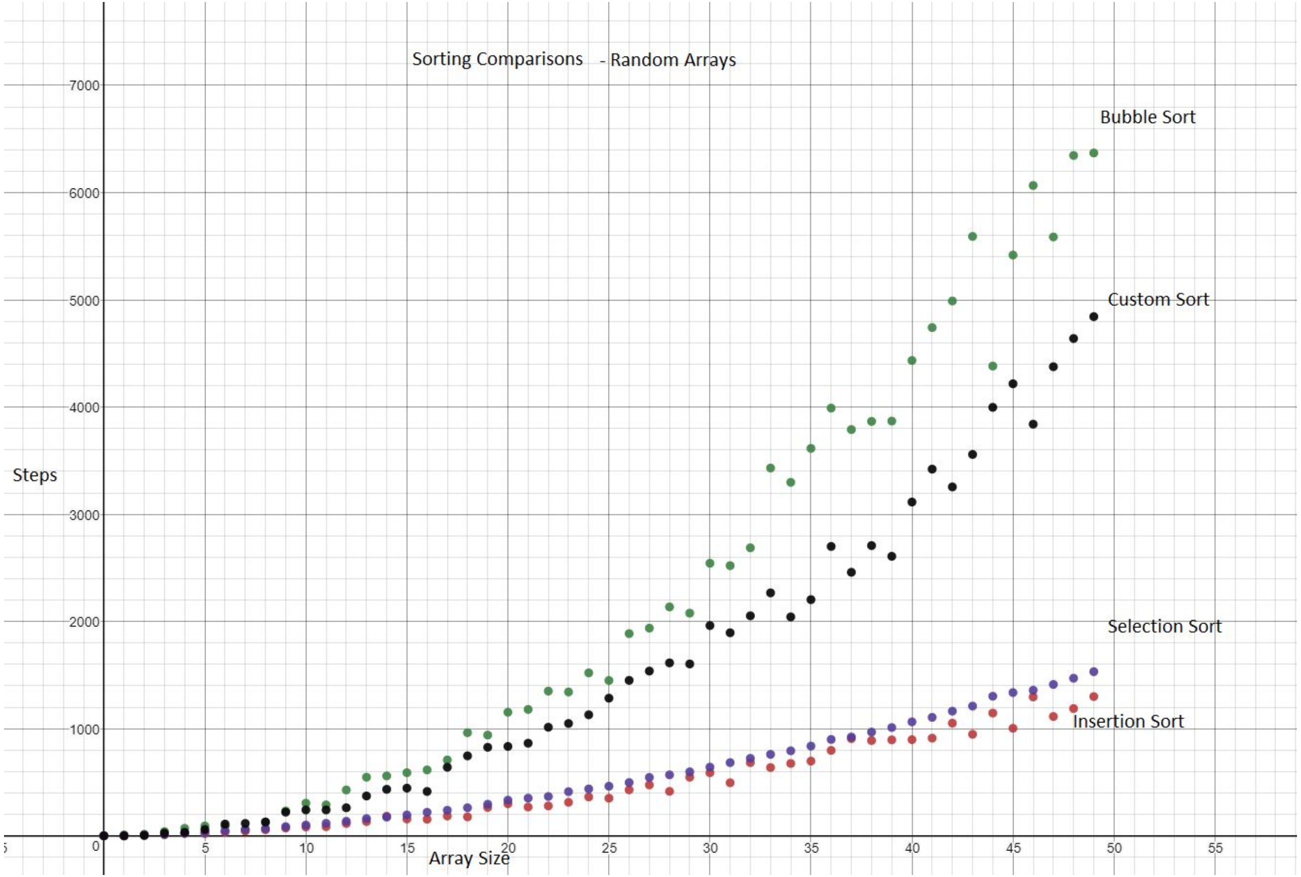
Array Size | Steps



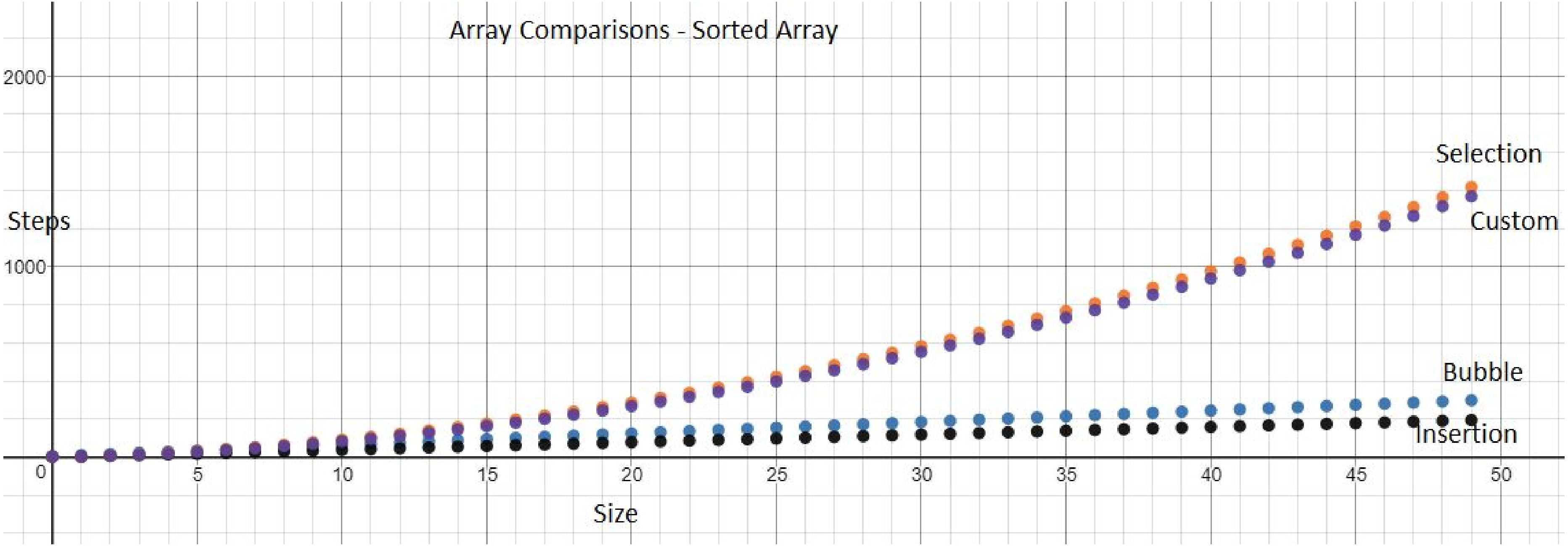
0	1
1	1
2	19
3	27
4	50
5	62
6	90
7	106
8	139
9	159
10	197
11	221
12	264
13	292
14	340
15	372
16	425
17	461
18	519
19	559
20	622
21	666



Comparison: Random



Comparison Sorted Array


Array Comparisons - Sorted Array
Steps
Size
Selection
Custom
Bubble
Insertion