ACIT 3896
Assignment 1

By: Caleb Seeman A01315336
Submitted to: Jeremey Holman

Alpha Sorting Raw Data:

```
PS C:\Users\Owner\Documents\Python-Assignment\Assignment-1> python .\alpha_testing.py
Size            Count     Median              Mean                StDev
16              10        0.087               0.089               0.008
32              10        0.248               0.242               0.017
64              10        0.581               0.608               0.061
128             10        2.132               2.367               0.432
256             10        8.933               8.886               0.994
512             10        37.999              37.183              2.484
1024            10        146.834             146.544             2.225
2048            10        576.744             590.168             44.953
4096            10        2307.242            2310.044            15.395
8192            10        9216.712            9224.905            51.891
16384           10        36751.177           36771.744           95.296
32768           10        142626.022          139134.832          7968.085
65536           10        515280.188          515482.254          1456.489
```

x_1	y_1
16	0.087
32	0.248
64	0.581
128	2.132
256	8.886
512	37.183
1024	146.544
2048	590.168
4096	2310.44
8192	9224.905
16384	36771.744
32768	139134.832
65536	515482.254

Bravo Sorting Raw Data:

```
PS C:\Users\Owner\Documents\Python-Assignment\Assignment-1> python .\bravo_testing.py
Size            Count     Median              Mean                StDev
16              10        0.05                0.052               0.007
32              10        0.078               0.078               0.002
64              10        0.13                0.132               0.004
128             10        0.232               0.233               0.004
256             10        0.446               0.47                0.077
512             10        0.971               1.069               0.218
1024            10        1.731               1.899               0.308
2048            10        3.87                3.896               0.635
4096            10        6.956               7.303               0.768
8192            10        15.095              14.903              0.783
16384           10        29.465              29.307              1.189
32768           10        57.029              56.426              2.904
65536           10        113.995             115.3               2.589
131072          10        247.56              261.299             56.183
262144          10        457.019             456.469             8.924
524288          10        910.878             912.857             12.026
1048576         10        1833.585            1831.419            12.173
2097152         10        3673.008            3676.336            27.889
4194304         10        7322.595            7290.534            113.085
```

x_1	y_1
16	0.052
32	0.078
64	0.132
128	0.233
256	0.47
512	1.069
1024	1.899
2048	3.896
4096	7.303
8192	14.903
16384	29.307
32768	56.426
65536	115.3
131072	261.299
262144	456.469
524288	912.857
1048576	1831.419
2097152	3676.336
4194304	7290.534

Charlie sorting Raw Data:

```
PS C:\Users\Owner\Documents\Python-Assignment\Assignment-1> python .\charlie_testing.py
Size            Count     Median              Mean                StDev
10              10        0.077               0.079               0.007
20              10        0.664               0.663               0.005
30              10        3.29                3.703               0.692
40              10        13.246              12.918              0.916
50              10        36.085              35.901              1.672
60              10        85.669              86.824              5.104
70              10        196.595             195.558             3.606
80              10        396.146             396.304             6.464
90              10        745.25              744.545             9.307
100             10        1335.213            1332.306            10.57
110             10        2282.729            2285.819            20.87
120             10        3762.689            3759.908            20.746
130             10        5985.916            5992.9              22.373
140             10        9286.716            9283.569            45.145
150             10        14109.174           14107.611           41.678
160             10        20784.626           20809.166           98.314
170             10        30212.718           30217.893           105.475
180             10        43117.122           43107.306           103.494
190             10        60592.993           60589.996           140.394
200             10        83981.088           83991.306           175.926
```

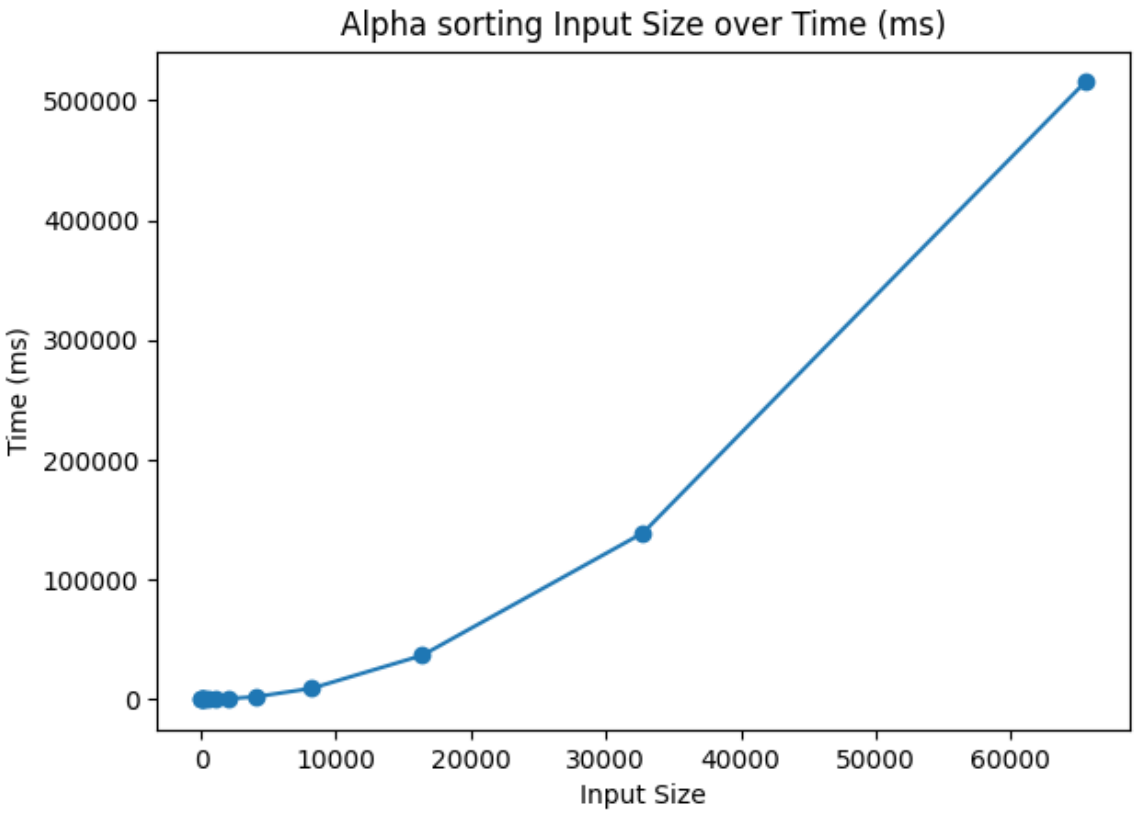
x_2	y_2
10	0.079
20	0.663
30	3.703
40	12.918
50	35.901
60	86.824
70	195.558
80	396.304
90	744.545
100	1332.306
110	2285.819
120	3759.908
130	5992.9
140	9283.569
150	14107.611
160	20809.166
170	30217.893
180	43107.306
190	60589.996
200	83991.306

Delta sorting Raw Data:

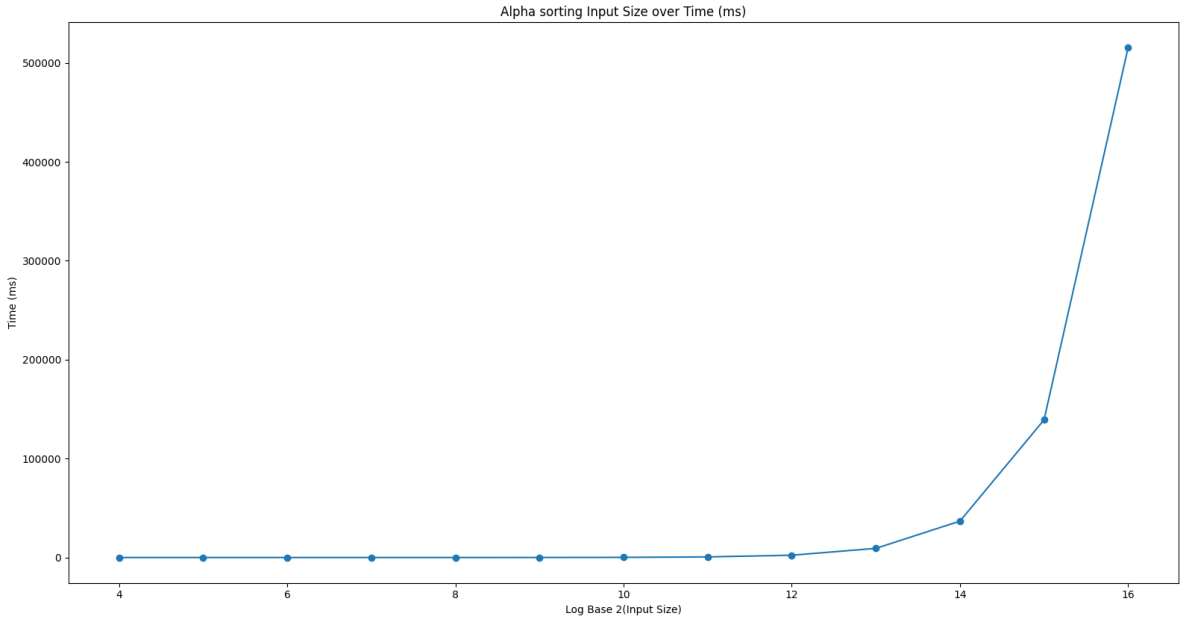
```
PS C:\Users\Owner\Documents\Python-Assignment\Assignment-1> python .\delta_testing.py
Size            Count     Median              Mean                StDev
2               10        0.013               0.015               0.006
4               10        0.018               0.019               0.002
8               10        0.033               0.034               0.003
16              10        0.1                 0.104               0.011
32              10        0.354               0.382               0.055
64              10        1.528               1.553               0.216
128             10        8.643               8.421               1.709
256             10        40.341              40.934              3.324
512             10        223.115             223.845             26.074
1024            10        1294.57             1306.56             54.578
2048            10        8366.483            8461.372            389.171
4096            10        61465.688           61462.496           1120.304
8192            10        461100.872          461133.86           7356.492
```

x_1	y_1
2	0.015
4	0.019
8	0.034
16	0.104
32	0.382
64	1.553
128	8.421
256	40.934
512	223.845
1024	1306.56
2048	8461.372
4096	61462.496
8192	461133.86

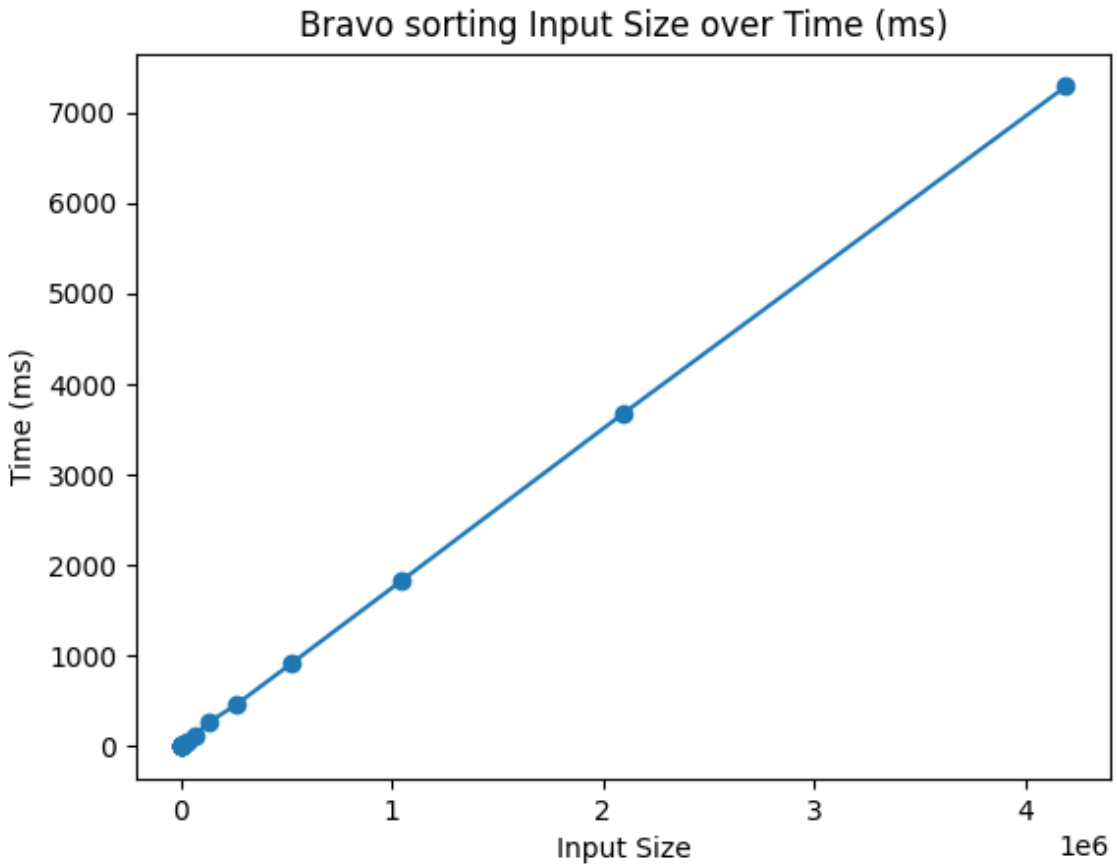
Visualization:

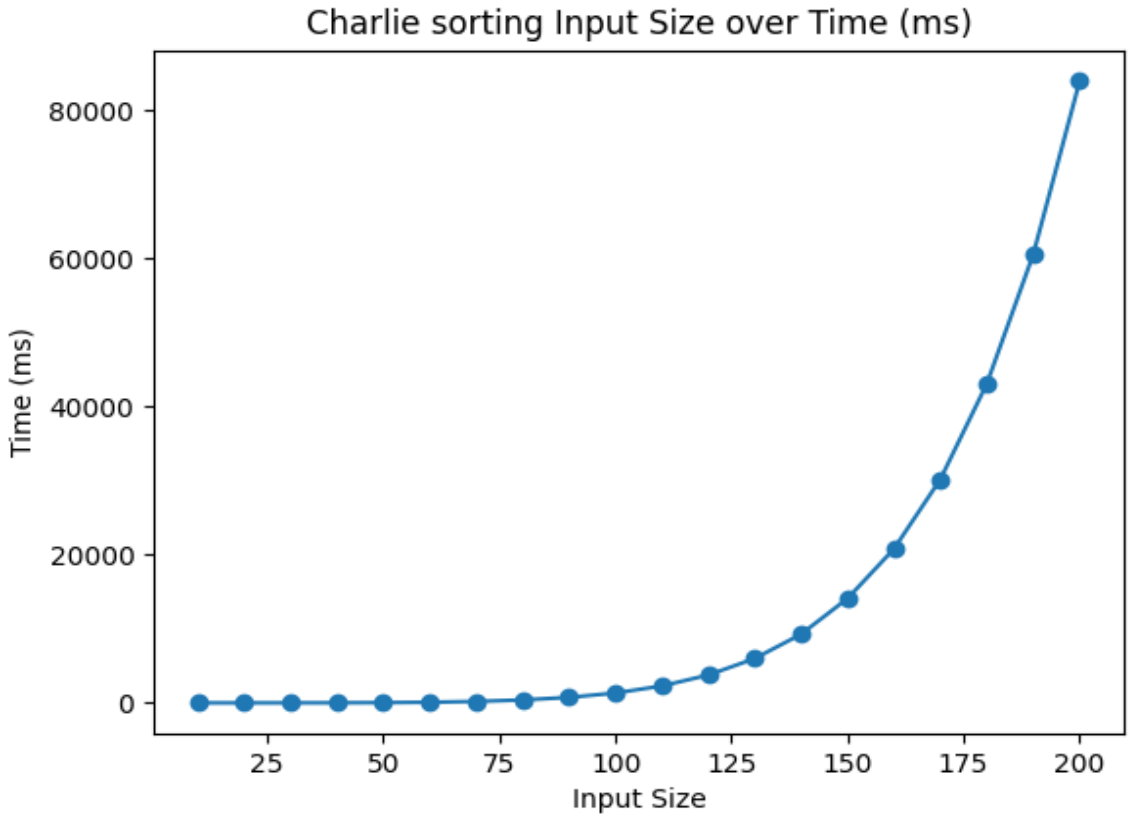


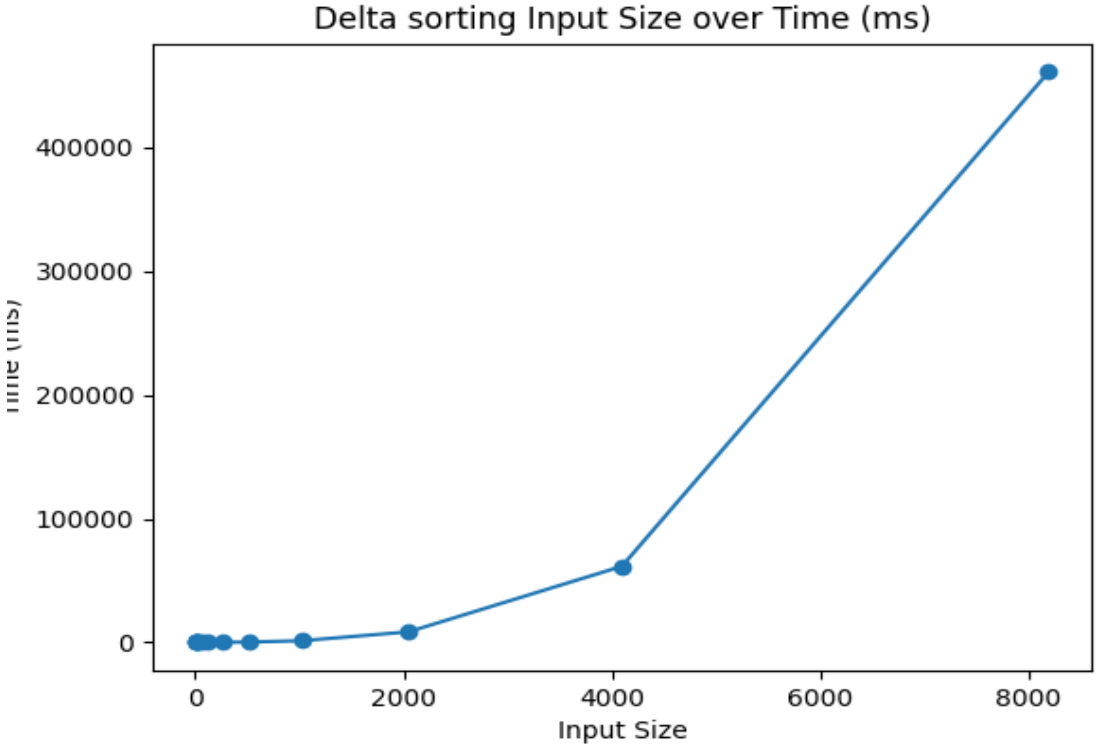
Note: For this first algorithm I attempted to graph the x axis using log base 2(Input Size) however when doing that I ended up with the following graph:



I concluded that this data type wasn't helpful in determining the final mathematical calculation for the slope and was less helpful for visualizing the data. The first graph has more condensed points when at the beginning it gives an estimate of what the slope looks like in between data sizes whereas the second graph doesn't.

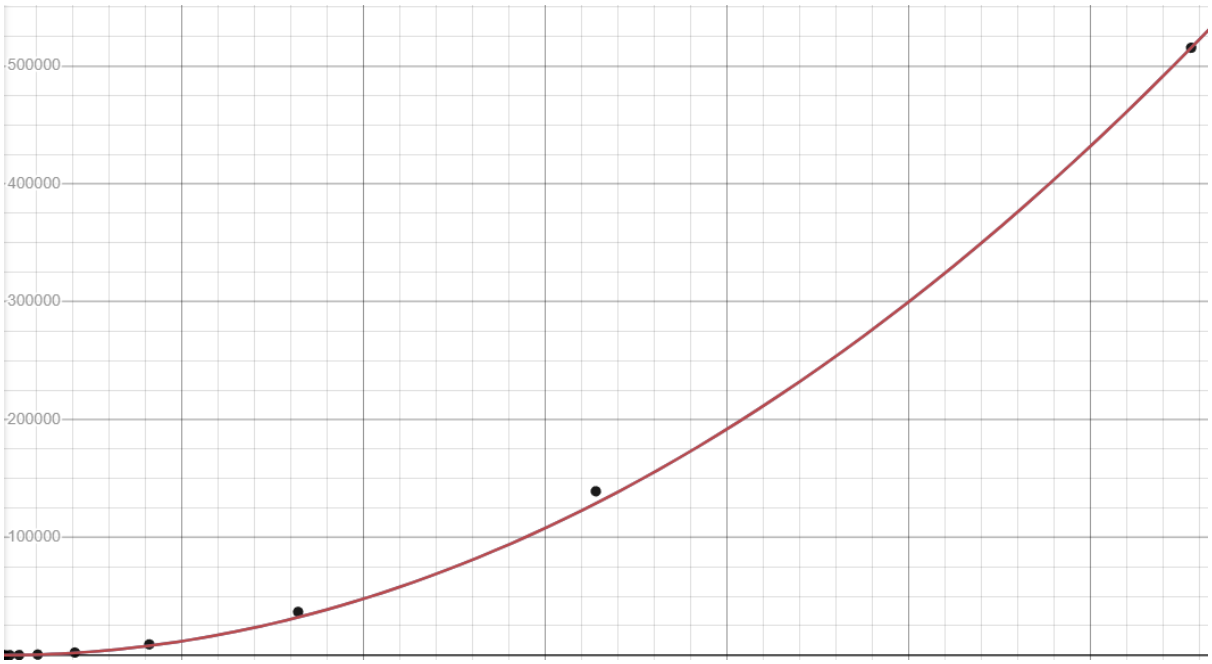



Charlie sorting Input Size over Time (ms)
Time (ms)
80000
60000
40000
20000
0
25
50
75
100
125
150
175
200
Input Size


Delta sorting Input Size over Time (ms)
400000
300000
200000
100000
0
0
2000
4000
6000
8000
Input Size

Alpha Hypothesis and Conjecture:

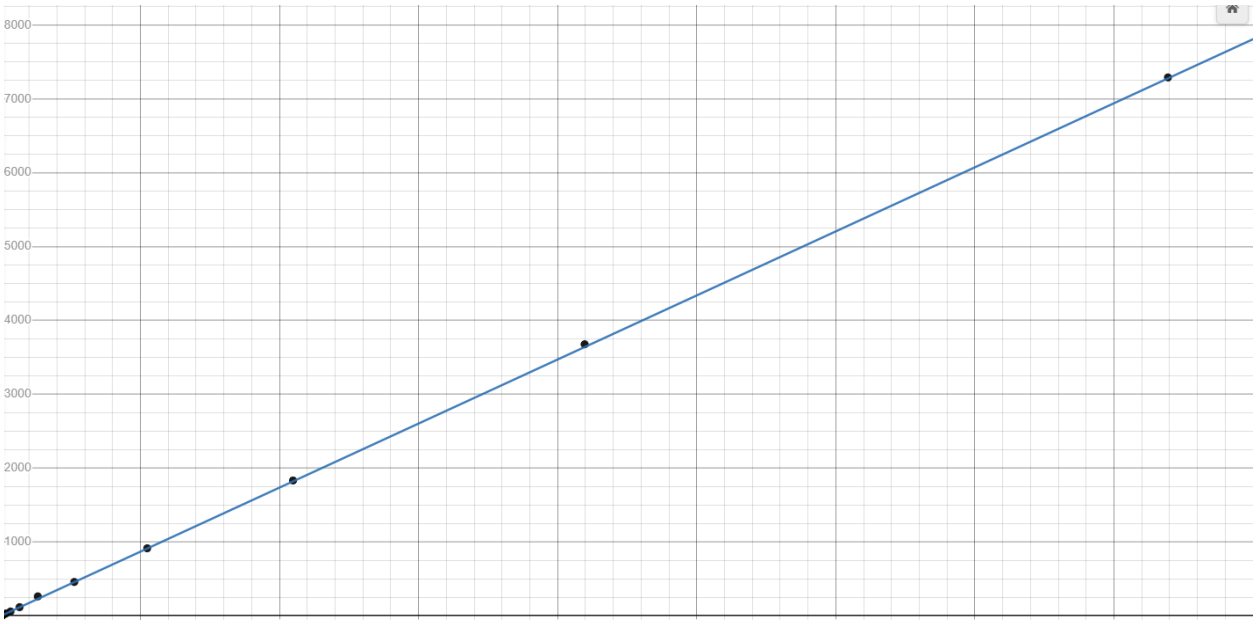
My hypothesis for the alpha sorting method is: It looks like it takes around 0.00012*n^2. The proof for my hypothesis can be seen in the following graph which shows the data points compared to a line of 0.00012*n^2.



The R-squared value for this line is R^2 = 0.9995. With this data and R-squared value we can be confident that with scaling this value will hold up to a very high accuracy based on our current data. With this taking the conjecture of attempting to find the run time over 48 hours we can reverse plug into our formula having 48 hours = 172800000 milliseconds. With this if we plug 172800000=0.00012*n^2 we can rearrange this to be n=sqrt(172800000/0.00012). Solving this equation we get that n = 1200000.

Bravo Hypothesis and Conjecture:

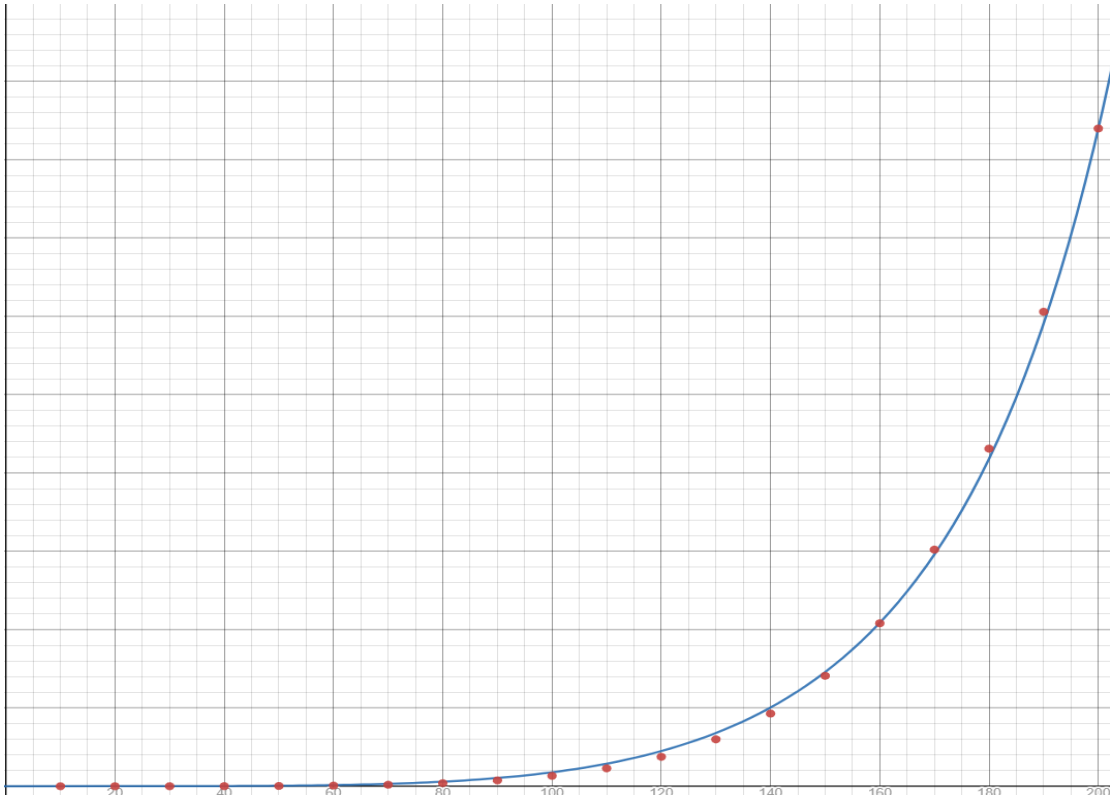
My hypothesis for the bravo sorting method is: It looks to take around 0.001735*n. The proof for my hypothesis can be seen in the following graph which shows the data points compared to a line of 0.001735*n.



The R-squared value for this line was R^2 = 1. With this data and the R-squared value we can be extremely confident that this will hold up with larger data points. With this taking the conjecture of attempting to find the run time over 48 hours we can reverse plug into our formula having 48 hours = 172800000 milliseconds. With this if we plug 172800000=0.001735*n we can rearrange this to be n=172800000/0.001735. Solving this equation we get that n = 99596541786.7.

Charlie Hypothesis and Conjecture:

My hypothesis for the bravo sorting method is: It looks to take around 2^0.075n+0.00000016n^5. The proof for my hypothesis can be seen in the following graph which shows the data points compared to a line of 2^0.075n+0.00000016n^5.

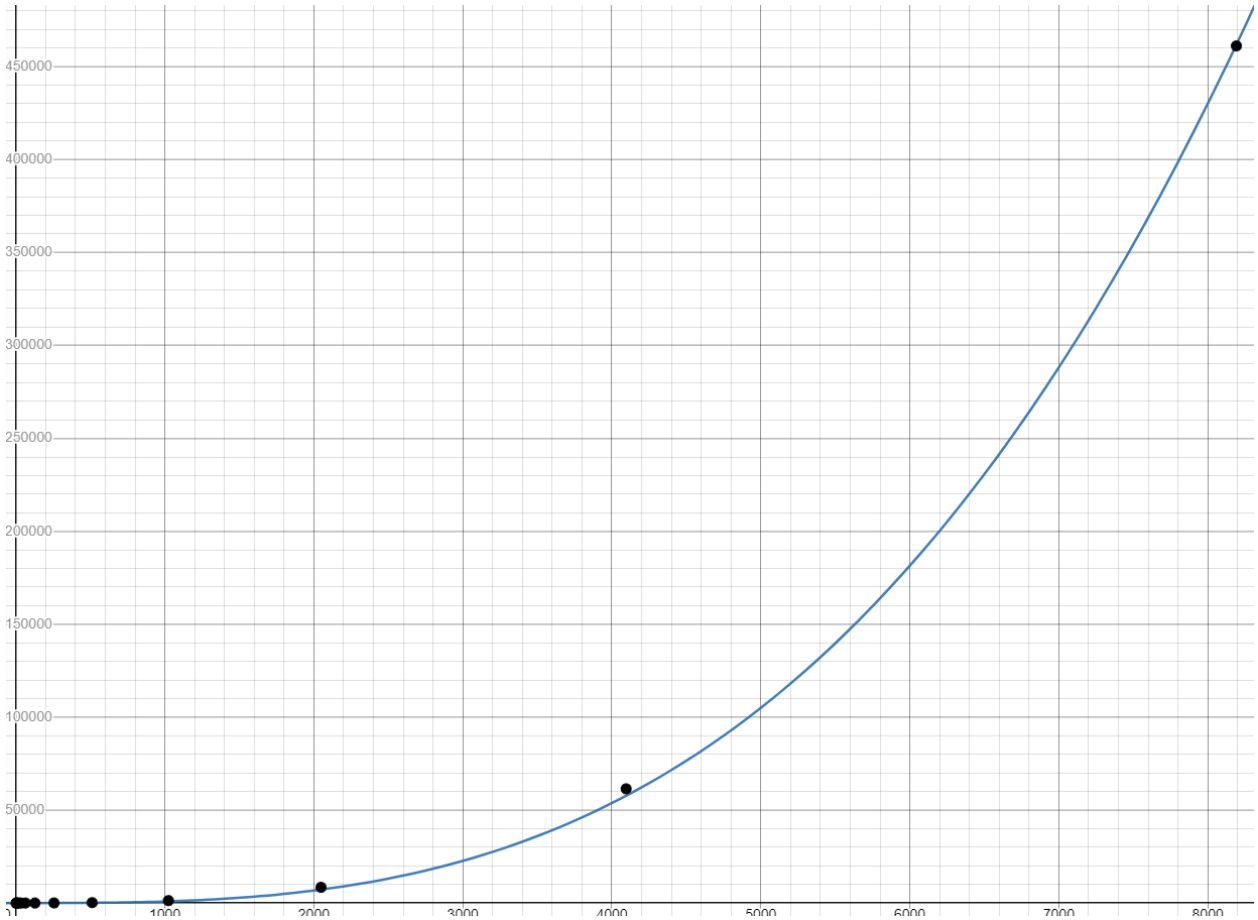


Since this required two variables I wasn't able to calculate the R-squared value for this line. However based on its accuracy to the points I am confident in saying that this will hold up as data gets larger. With this taking the conjecture of attempting to find the run time over 48 hours we can reverse plug into our formula having 48 hours = 172800000 milliseconds. With this if we plug 172800000=2^0.075n+0.00000016n^5 we can rearrange this to be n=log base 2(172800000)/0.075 we can do this because as n approaches a very large number n^5 can be discarded as it will not affect the value much. Solving this equation we get that n = 321.138318739. As additional proof that my conjecture is true I plugged in what would be the value given my function and it came to ~320.5.

321.12	17800000
320.554	17800000

Delta sorting Hypothesis and Conjecture:

My hypothesis for the bravo sorting method is: It looks to take around 0.00000084n^3. The proof for my hypothesis can be seen in the following graph which shows the data points compared to a line of 0.00000084n^3.



The R-squared value for this line was R^2 = 0.9999. With this data and the R-squared value we can be extremely confident that this will hold up with larger data points. With this taking the conjecture of attempting to find the run time over 48 hours we can reverse plug into our formula having 48 hours = 172800000 milliseconds. With this if we plug 172800000=0.00000084n^3 we can rearrange this to be n=cuberoot(172800000/0.00000084). Solving this equation we get that n=59032.