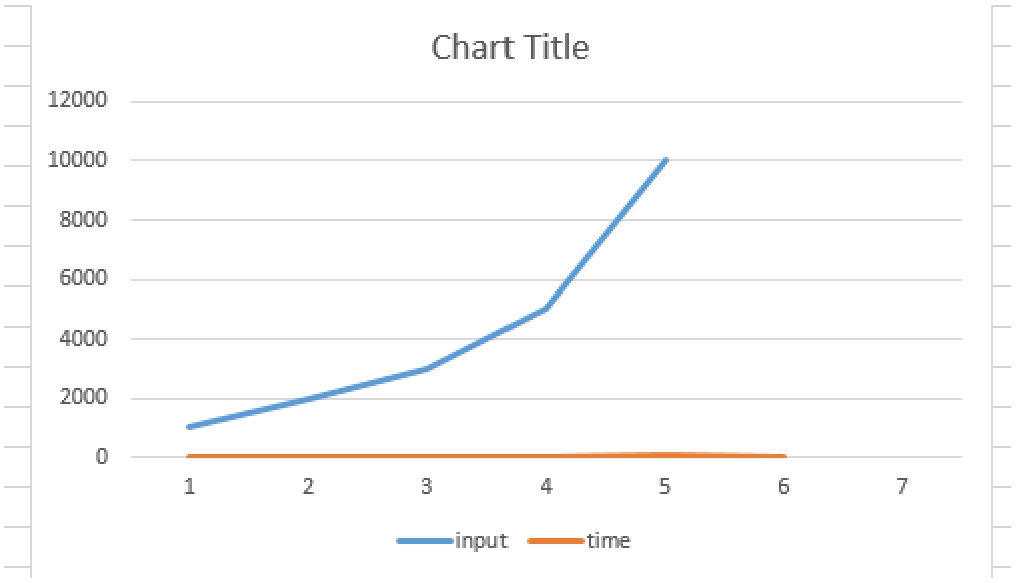
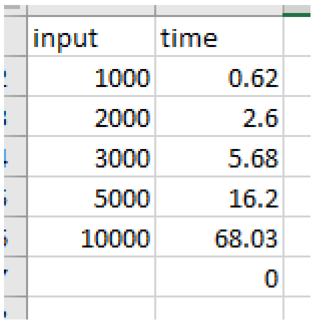
Steven Lopez 861106782 CS141 assignment1

Brute Force





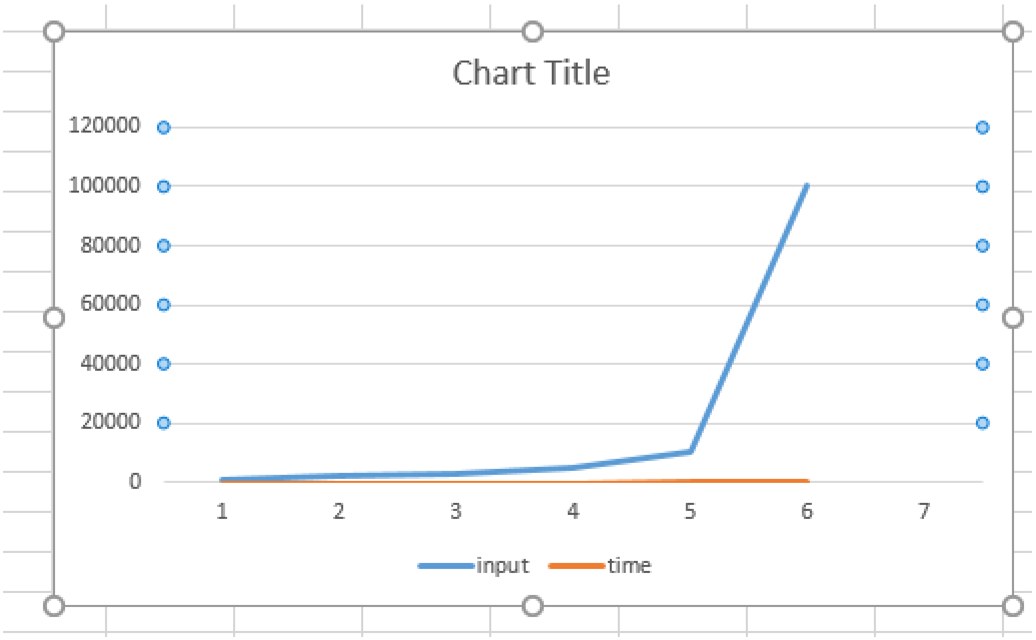
input	time
1000	0.62
2000	2.6
3000	5.68
5000	16.2
10000	68.03
	0

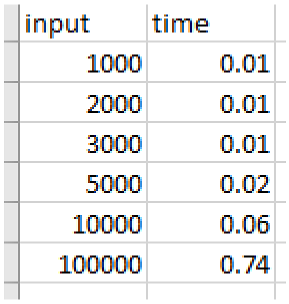
As the input gets larger from 1000 to 10000 we can see the increase from .62 seconds to 68.03 seconds, the time gets large quickly and I didn't want to test 100,000 but it would've been significantly slower.

My algorithm for this is to sort the points by their x and y values. We compare the first point(based on x values) to all the rest which is n times, then the same for the second which is n – 1 times. We do this for all the points and it's closer to .5(n^2) which is still n^2 run time.

NOTE: my function for this isn't 100% accurate, appears to be off by one with small sets(probably large as well but can't notice)

Divide and Conquer





input	time
1000	0.01
2000	0.01
3000	0.01
5000	0.02
10000	0.06
100000	0.74

Also tried 1,000,000(didn't put it) and runs in 11.68 seconds.

My algorithm sorts the list by x and y values, creates two new lists**, constant** for each. We then find the middle of the sorted x list, constant. We put half on the left side and the other half on the right side, which is **constant** as well. We call this until we get the minimum then compare them which takes **logn** time due to the recursion and **n** for the number of calls. This is for one function that returns "minOuter".

The second function is for the middle, we have our sorted x list and the min distance passed in. We check if any x's in the list are within min distance of our midpoint, if so create a new list and pass the qualifying x's in. This takes **constant** time since we go through the list once. I then called brute force on the points, minimal amount of points so not constant, but fast enough that doesn't add to the runtime. Overall the runtime is nlogn, due to the recursion and how many times it's called.