Michael Smith

Part 3:

a.

Merge sort

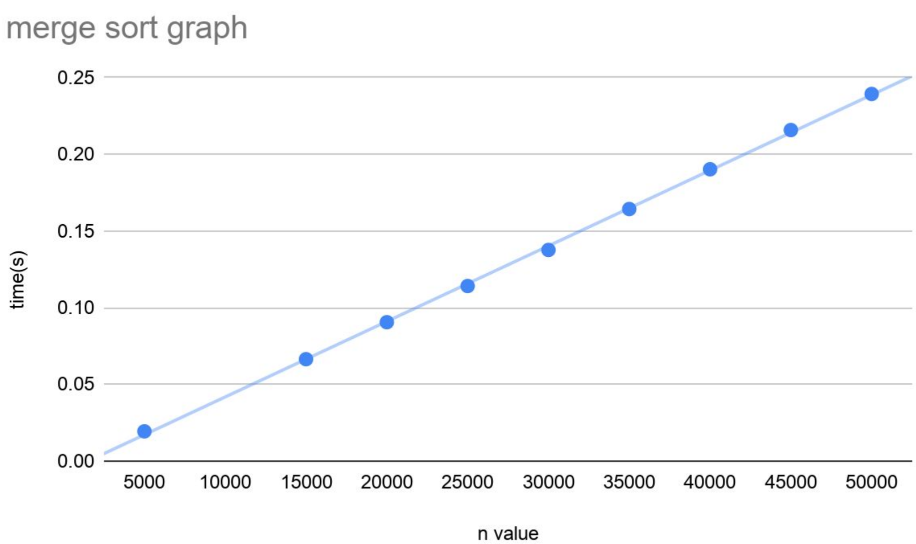
n value	time(s)
5000	0.0194110870361
10000	0.044319152832'
15000	0.066447019577
20000	0.0905520915985
25000	0.114130020142
30000	0.137635946274
35000	0.164332151413
40000	0.190243959427
45000	0.215818166733
50000	0.239320039749

Insert sort

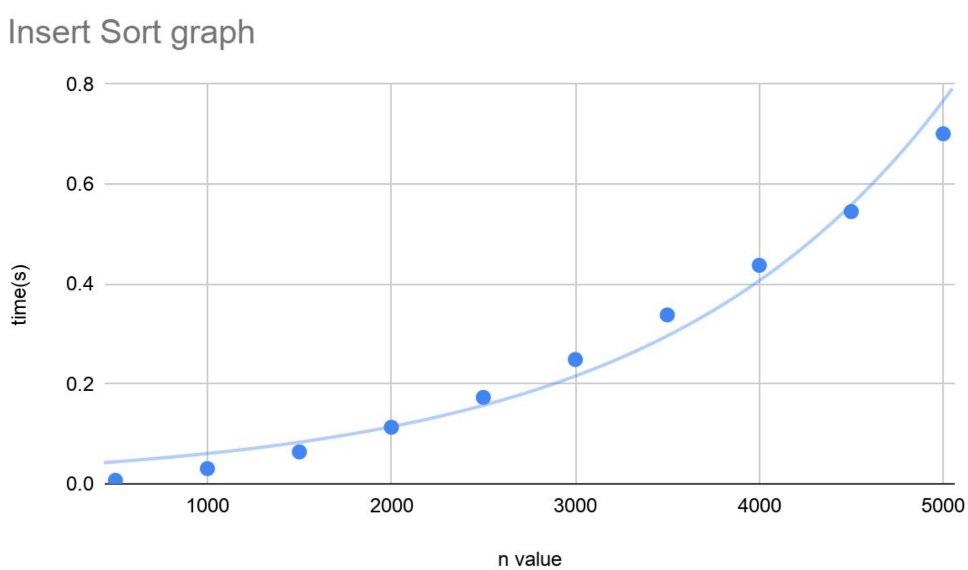
n value	time(s)
500	0.00702500343323
1000	0.0304479598999
1500	0.0639569759369
2000	0.113276004791
2500	0.173090934753
3000	0.24889588356
3500	0.338148117065
4000	0.437569141388
4500	0.545073986053
5000	0.700849056244

In order to generate the random arrays, I created an array of ranges that picks a random number from that range to set the soon to be sorted array's size. I had to modify the n values for insert sort when taking my data because it would have a runtime longer than a minute.

b.

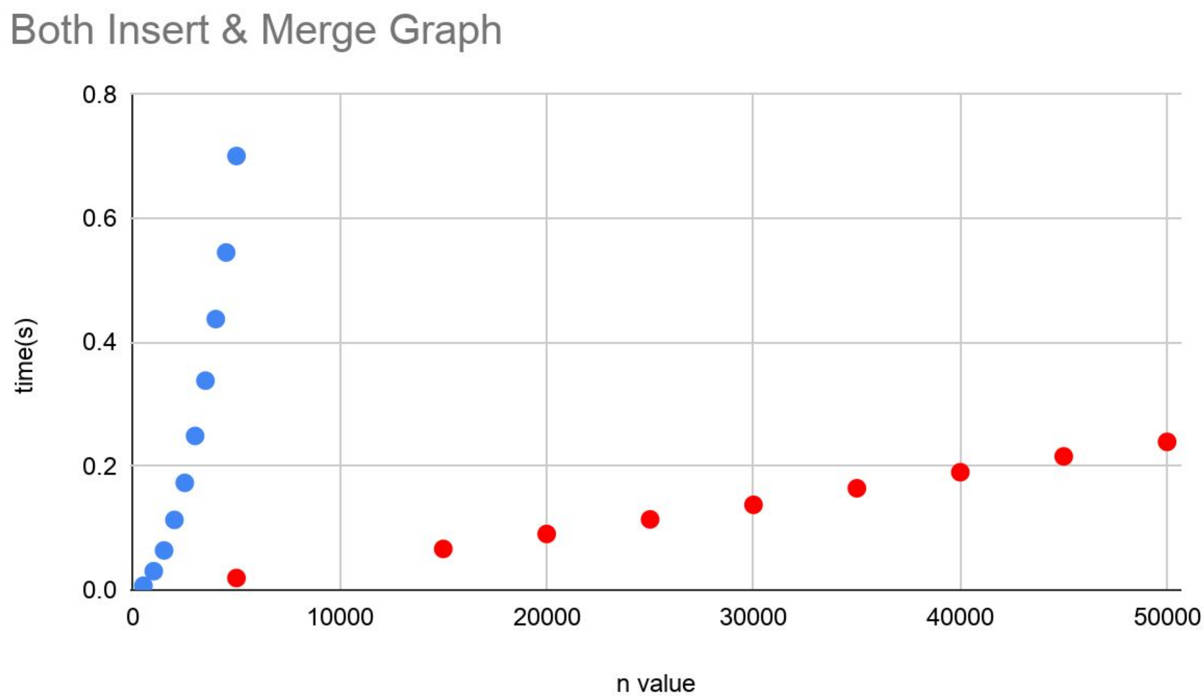


I would assume from the reading that this scatter plot follows the nlog(n) curve.



I assumed the insert sort follows an exponential curve judging from the data.

c.



I wasn't able to make a key but the blue represents the Insert sort times and the red represents the merge sort times.

d. Initially, I was not getting a believable answer when trying to predict what the runtime would be for both algorithms so I'm not entirely sure how accurate these guesses are. I did more research and figured I was missing multiple scalar values for each algorithm so I used exel to get these answers.

I predict that Merge Sort runtime when n = 200,000 is .65213271 seconds using the algorithm a*n*log(n) and I also predict that Insert Sort runtime when n = 200,000 would be extremely long, I was not able to calculate it on my calculator.

e. My comparison of the two algorithms isn't too complex. Each graph follows the respective algorithms closely and when put together you can see the major differences. The Insert Sorting experimental data I collected follows an exponential curve of n^2 and the Merge Sorting experimental data follows a logarithmic n*log(n) curve respectively.