Zeynep Selcen Öztunç
21902941
Section 03

CS 202-Assignment 01 Report

Question 1

Question 1

part a) By the big-Oh definition, $f(n)$ is $O(n^3)$ if $T(n) \leq cn^3$ for some $n \geq n_0$. Let us check this condition:

if $3n^3 + 4n^2 + 2n \leq cn^3$ then, $3 + \frac{4}{n} + \frac{2}{n^2} \leq c$

Therefore, the Big-Oh condition holds for $n \geq n_0 = 1$ and $c \geq 9$ ↳ (3+4+2)

part b)

1. $T(n) = T(n-1) + n^2$ (1) $(T(1)=1)$

$T(n-1) = T(n-2) + (n-1)^2$ (we put $n=n-1$ in (1))

Let's replace the $T(n-1)$ in (1) with $T(n-2) + (n-1)^2$

$T(n) = T(n-2) + n^2 + (n-1)^2$ (2)

$T(n-2) = T(n-3) + (n-2)^2$ (we put $n=n-2$ in (1))

Let's replace the $T(n-2)$ in 2 with $T(n-3) + (n-2)^2$

$T(n) = T(n-3) + (n-2)^2 + (n-1)^2 + n^2$ (3)

$\vdots$

$T(n) = T(n-k) + (n-(k-1))^2 + (n-(k-2))^2 \dots n^2$

Since $T(1)=1$, let's replace the k values with $n-1$ to obtain $T(1)$

$T(n) = T(1) + (n-(n-1-1))^2 + (n-(n-1-2))^2 \dots n^2$

$= T(1) + 2^2 + 3^2 \dots n^2 = 1^2 + 2^2 + 3 \dots n^2 = \frac{n(n+1)(2n+1)}{6}$

So $T(n) = \Theta(n^3)$

2. $T(n) = 2T(n/2) + n/2$ ① $(T(1) = 1)$

$T(n/2) = 2T(n/2^2) + n/2^2$ (we put $n = n/2$ in ①)

Let's replace the $T(n/2)$ in ① with $2T(n/2^2) + n/2^2$

$T(n) = 2\left(2T\left(\frac{n}{2^2}\right) + n/2^2\right) + n/2$

$T(n) = 2^2 T\left(\frac{n}{2^2}\right) + n/2 + n/2$ ②

$T(n/2^2) = 2T(n/2^3) + n/2^3$ (we put $n = n/2^2$ in ①)

Let's replace the $T(n/2^3)$ in ② with $2T(n/2^3) + n/2^3$

$T(n) = 2^2\left(2T(n/2^3) + n/2^3\right) + n/2 + n/2$

$T(n) = 2^3 T(n/2^3) + n/2 + n/2 + n/2 = 2^3 T(n/2^3) + 3(n/2)$ ③

$\vdots$

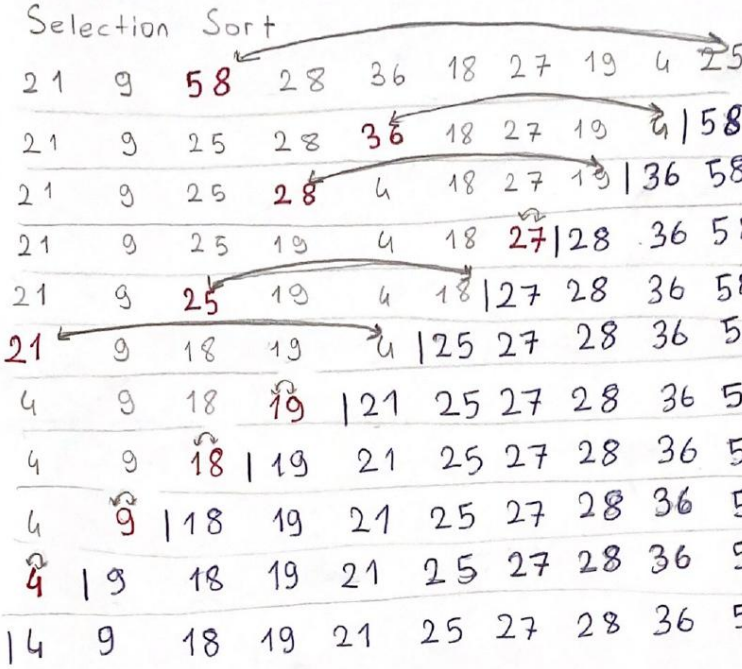
$T(n) = 2^k T(n/2^k) + k(n/2)$ Since $T(1) = 1$, let's try to obtain $T(1)$ by making $n/2^k$ equal to 1. If $n/2^k = 1$, $n \overset{(\star)}{=} 2^k \rightarrow k = \log n$. Then,

$T(n) \overset{(\star)}{=} 2^k T(1) + \log n\,(n/2) = n \cdot 1 + \log n\,(n/2) = n + \frac{n \cdot \log n}{2}$

So $T(n) = \Theta(n \log n)$

Part c)

Selection Sort

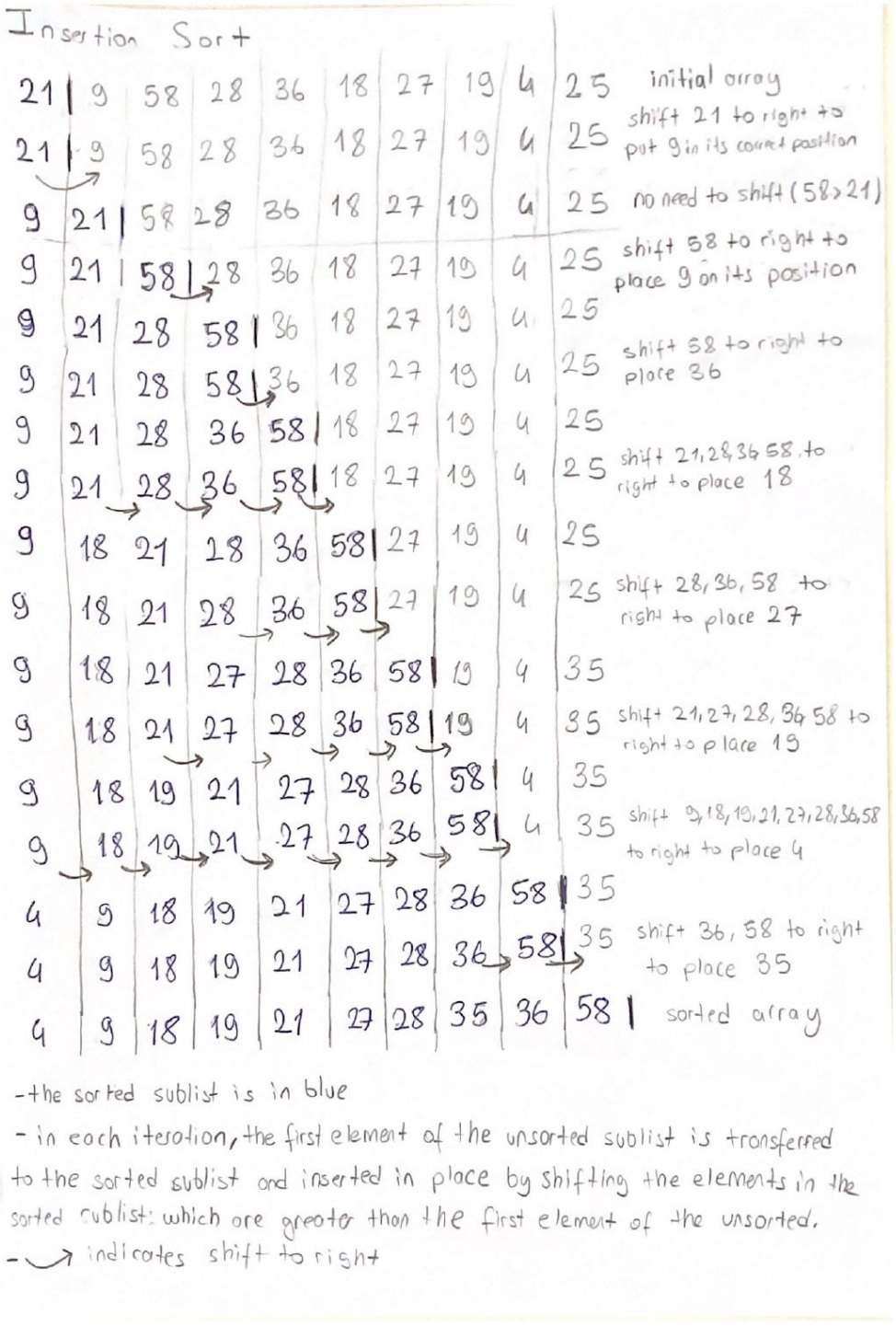


21	9	58	28	36	18	27	19	4	25	initial array:
21	9	25	28	36	18	27	19	4	58	after 1st swap:
21	9	25	28	4	18	27	19	36	58	after 2nd swap:
21	9	25	19	4	18	27	28	36	58	after 3rd swap:
21	9	25	19	4	18	27	28	36	58	after 4th swap:
21	9	18	19	4	25	27	28	36	58	after 5th swap:
4	9	18	19	21	25	27	28	36	58	after 6th swap:
4	9	18	19	21	25	27	28	36	58	after 7th swap:
4	9	18	19	21	25	27	28	36	58	after 8th swap:
4	9	18	19	21	25	27	28	36	58	after 9th swap:
4	9	18	19	21	25	27	28	36	58	after 10th swap: (sorted array)

- the sorted sublist is in blue
- the largest element in the unsorted sublist is in red

In each iteration, it swaps the largest element in the unsorted list with the last element in the unsorted list.

- ↷ indicates swapping



Question 2

2.c)

```
[selcen.oztunc@dijkstra ~]$ ./hw1
Bubble Sort:----------------------------------------------------
Number of key comparisons in the bubble sort algorithm is: 114
Number of data moves in the bubble sort algorithm is: 204
The contents of the array after bubble sort is:
12 23 24 25 26 27 29 31 32 33 35 37 38 40 56 79
Merge Sort:-----------------------------------------------------
Number of key comparisons in the merge sort algorithm is: 46
Number of data moves in the merge sort algorithm is: 128
The contents of the array after merge sort is:
12 23 24 25 26 27 29 31 32 33 35 37 38 40 56 79
Quick Sort:-----------------------------------------------------
Number of key comparisons in the quick sort algorithm is: 48
Number of data moves in the quick sort algorithm is: 114
The contents of the array after quick sort is:
12 23 24 25 26 27 29 31 32 33 35 37 38 40 56 79
```

2.d)

Analysis of Randomly Ordered Arrays

```
--------------------------------
Analysis of Merge Sort
Array size            Elapsed time              compCount             moveCount
4000                   1.365 ms                  42839                 95808

8000                   2.872 ms                  93669                207616

12000                  4.482 ms                 147708                327232

16000                  6.088 ms                 203293                447232

20000                  7.752 ms                 260776                574464

24000                  9.375 ms                 319373                702464

28000                 11.162 ms                 378647                830464

32000                 12.813 ms                 438668                958464

36000                 14.606 ms                 499882               1092928

40000                 16.356 ms                 561750               1228928

44000                  18.11 ms                 624122               1364928

48000                 19.769 ms                 686786               1500928
```

```
--------------------------------
Random Arrays
--------------------------------
Analysis of Bubble Sort
Array size             Elapsed time               compCount              moveCount
4000                  156.019 ms                  7995299               12111072

8000                  654.569 ms                 31987354               47533452

12000                   1500.1 ms                 71987330              107609346

16000                  2682.18 ms                127976424              191072775

20000                  4215.54 ms                199989955              300482910

24000                  6051.55 ms                287968497              426177774

28000                  8282.57 ms                391845285              588483678

32000                  10799.3 ms                511915365              766674468

36000                  13764.6 ms                647978679              972677208

40000                    16946 ms                799948875             1199154543

44000                    20540 ms                967906369             1459460853

48000                    24463 ms               1151938325             1735511643
```

```
-----------------------------------
Analysis of Quick Sort
Array size            Elapsed time              compCount              moveCount
4000                   1.234 ms                  52723                  86017

8000                   2.782 ms                 119932                 200495

12000                   4.178 ms                 184367                 290731

16000                   6.126 ms                 265623                 466729

20000                   7.686 ms                 327681                 572263

24000                   9.335 ms                 401081                 697128

28000                  11.567 ms                 493224                 902983

32000                   12.99 ms                 568592                 966471

36000                  14.598 ms                 621007                1081615

40000                  17.159 ms                 744132                1324426

44000                  18.571 ms                 853434                1385343

48000                   20.38 ms                 915666                1547535
```

Analysis of Ascending Arrays

```
--------------------------------
Ascending Arrays
--------------------------------
Analysis of Bubble Sort
Array size            Elapsed time              compCount              moveCount
4000                   0.023 ms                  3999                      0

8000                   0.104 ms                  7999                      0

12000                   0.067 ms                 11999                      0

16000                   0.089 ms                 15999                      0

20000                   0.112 ms                 19999                      0

24000                    0.18 ms                 23999                      0

28000                   0.155 ms                 27999                      0

32000                   0.178 ms                 31999                      0

36000                   0.198 ms                 35999                      0

40000                    0.22 ms                 39999                      0

44000                   0.242 ms                 43999                      0

48000                   0.268 ms                 47999                      0

-----------------------------------
Analysis of Merge Sort
```

```
-----------------------------------
Analysis of Merge Sort
Array size            Elapsed time              compCount              moveCount
4000                    0.879 ms                  24176                  95808

8000                    1.972 ms                  52352                 207616

12000                    2.886 ms                  84304                 327232

16000                    3.919 ms                 112704                 447232

20000                    5.014 ms                 148016                 574464

24000                    6.207 ms                 180608                 702464

28000                    7.119 ms                 212720                 830464

32000                    8.449 ms                 241408                 958464

36000                    9.386 ms                 279185                1092928

40000                    10.41 ms                 316033                1228928

44000                   11.624 ms                 352048                1364928

48000                   12.677 ms                 385217                1500928

-----------------------------------
```

```
-----------------------------------
Analysis of Quick Sort
Array size             Elapsed time              compCount              moveCount
4000                   34.176 ms                 7998000                 15996

8000                  135.892 ms                31996000                 31996

12000                  304.857 ms                71994000                 47996

16000                  541.747 ms               127992000                 63996

20000                  846.346 ms               199990000                 79996

24000                  1218.72 ms               287988000                 95996

28000                   1658.3 ms               391986000                111996

32000                  2165.57 ms               511984000                127996

36000                  2740.38 ms               647949307                143994

40000                  3383.16 ms               799965069                159995

44000                  4094.12 ms               967969002                175995

48000                  4871.81 ms              1151934656                191995
```

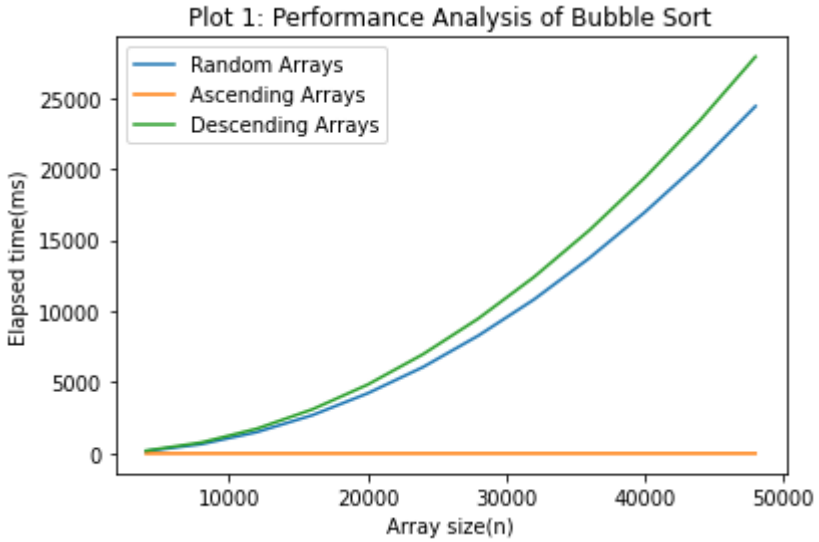
Analysis of Descending Arrays

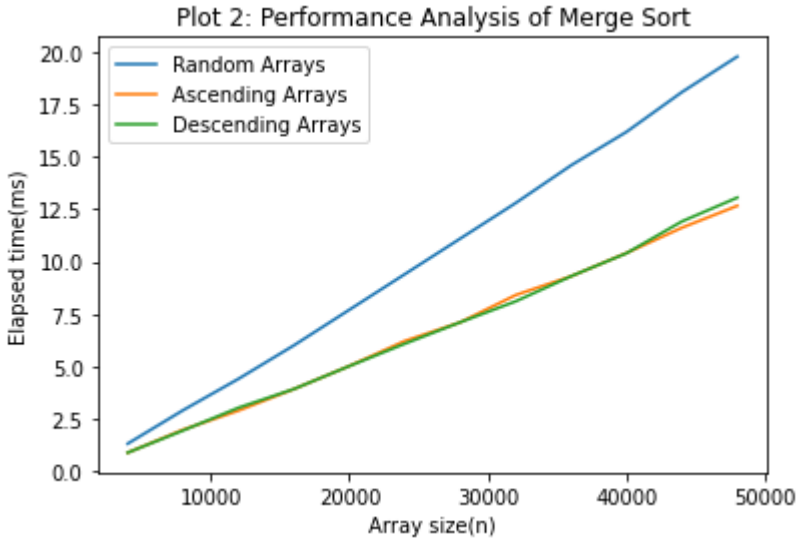
```
--------------------------------
Descending Arrays
--------------------------------
Analysis of Bubble Sort
Array size             Elapsed time              compCount               moveCount
4000                  193.607 ms                 7998000                23994000

8000                  774.334 ms                31996000                95988000

12000                  1744.16 ms                71994000               215982000

16000                   3101.5 ms               127992000               383976000

20000                  4844.32 ms               199990000               599970000

24000                  6977.99 ms               287988000               863964000

28000                  9496.21 ms               391986000              1175958000

32000                  12395.7 ms               511984000              1535952000

36000                    15705 ms               647982000              1943946000

40000                  19368.3 ms               799980000             -1895027299

44000                  23449.3 ms               967978000             -1391033299

48000                  27889.8 ms              1151976000              -839039299

-----------------------------------
```

```
-----------------------------------
Analysis of Merge Sort
Array size             Elapsed time              compCount              moveCount
4000                    0.879 ms                  23728                  95808

8000                    1.939 ms                  51456                 207616

12000                   3.018 ms                  79312                 327232

16000                   3.911 ms                 110912                 447232

20000                   5.008 ms                 139216                 574464

24000                   6.076 ms                 170624                 702464

28000                    7.14 ms                 202512                 830464

32000                   8.153 ms                 237824                 958464

36000                   9.354 ms                 267280                1092928

40000                  10.441 ms                 298432                1228928

44000                  11.928 ms                 330416                1364928

48000                  13.059 ms                 365248                1500928

-----------------------------------
```

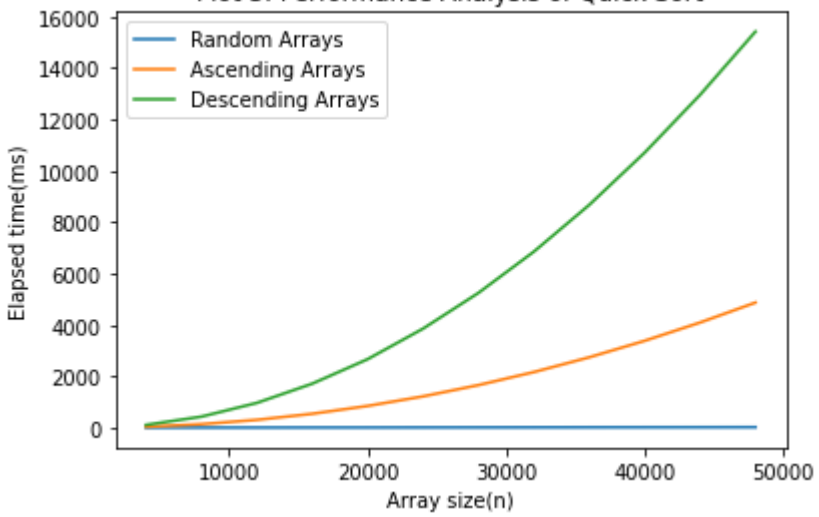
```
-----------------------------------
Analysis of Quick Sort
Array size            Elapsed time              compCount              moveCount
4000                 107.276 ms                 7998000                12015996

8000                 429.071 ms                31996000                48031996

12000                 964.252 ms                71994000               108047996

16000                1713.91 ms                127992000               192063996

20000                2677.94 ms                199990000               300079996

24000                3856.05 ms                287988000               432095996

28000                5248.39 ms                391986000               588111996

32000                6854.53 ms                511984000               768127996

36000                 8675.1 ms                647982000               972143996

40000                10710.7 ms                799980000              1200159996

44000                12959.7 ms                967978000              1452175996

48000                15423.3 ms               1151976000              1728191996

[selcen.oztunc@dijkstra ~]$
```

Question 3

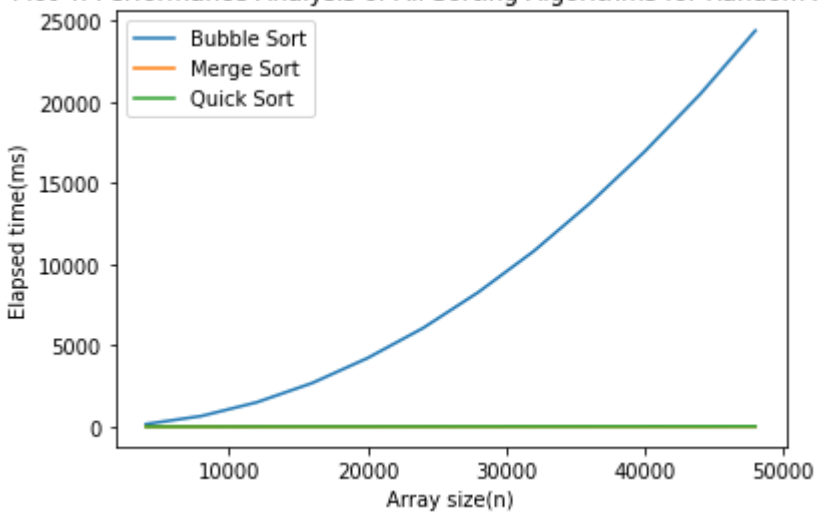




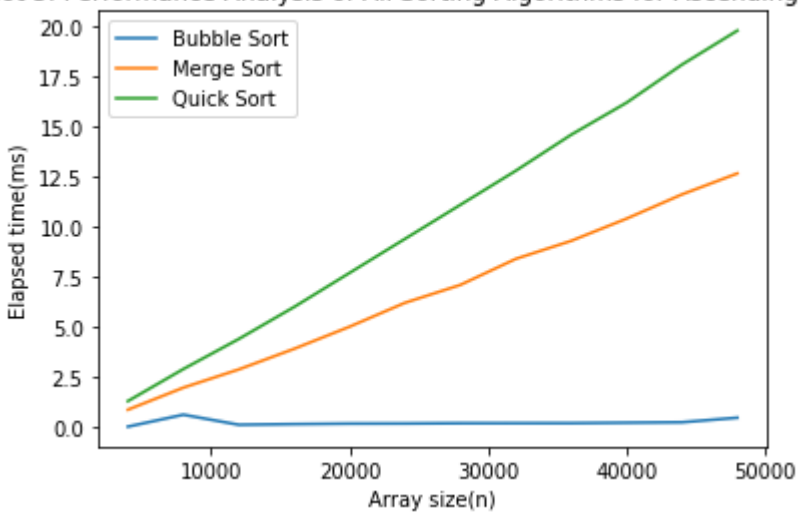
Plot 3: Performance Analysis of Quick Sort

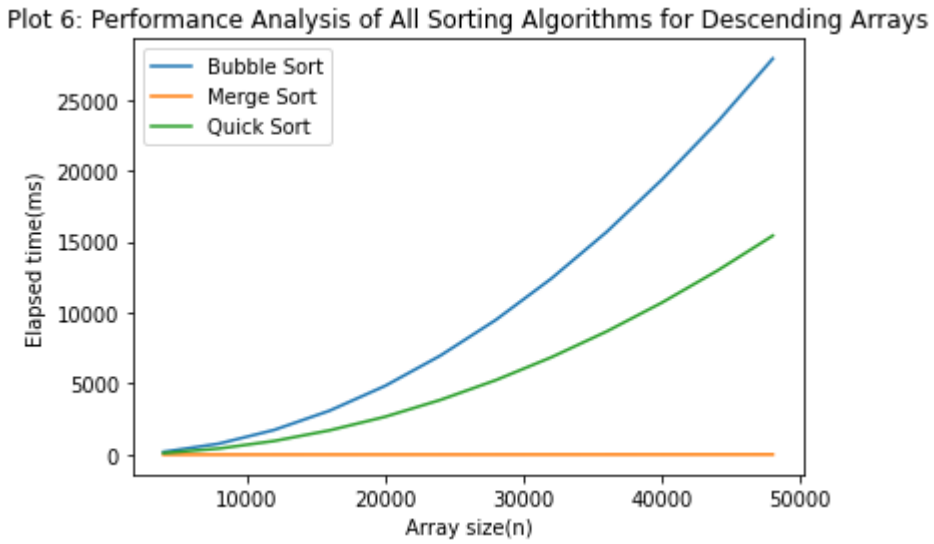


Plot 4: Performance Analysis of All Sorting Algorithms for Random Arrays



Plot 5: Performance Analysis of All Sorting Algorithms for Ascending Arrays





In terms of performance analysis, the random arrays in the homework represent the average case, the descending arrays represent the worst case and the ascending arrays represent the best case. Quick sort has O(nlogn) running time for its best and average cases and O(n^2) for its worst case. It can be seen from Plot 3 that when the array is in descending order, the elapsed time is much higher compared to the other orders of the arrays. This is because the first element of the array is chosen as pivot in the implementation so the whole array needs to be reversed, which is the worst case and its time complexity is O(n^2). In ascending and random arrays the empirical results also satisfy the theoretical ones, the running time is O(nlogn) which is almost a linear line. Merge sort has the O(nlogn) time complexity for the best, worst and average cases. The empirical results are also in line with this, as it can be seen from Plot 2. Bubble sort runs O(n^2) in the worst and average cases and O(n) in the best case. As expected, it is fastest when the array is already sorted (ascending arrays) and slowest when the array is in descending order which can be seen from Plot 1.

From Plot 4, it can be seen that the bubble sort is the slowest algorithm compared to the other two. This corresponds to the theoretical results since it has O(n^2) running time in the average case. Since merge sort and quick sort have very similar elapsed times, they overlap in the plot. We also know this is true from the theoretical results; they both have O(n logn) running times. Plot 6 which compares all algorithms in descending order also clearly shows that the bubble sort is the slowest. The empirical and theoretical results are usually the same but they slightly differ because of hardware differences, the IDE that I have used as well as the limited test cases that I have tried.