Name: Zeynep Doğa Dellal

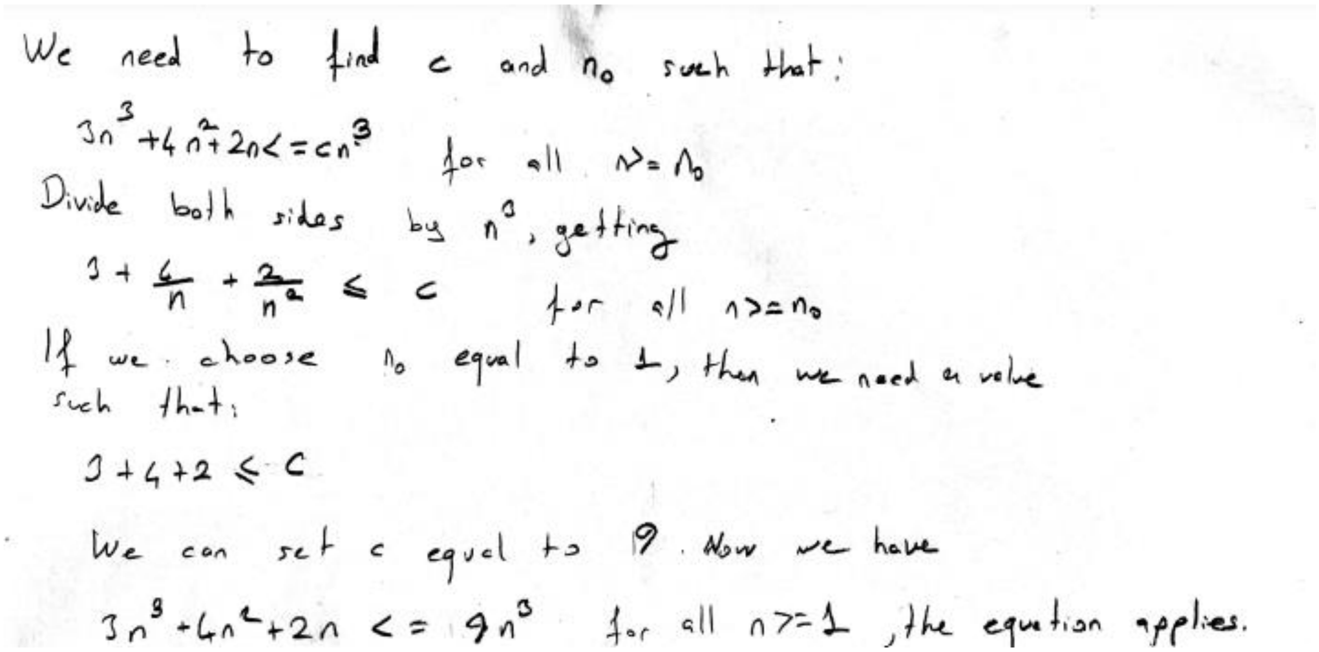
ID: 22002572

Section: 01

Homework: 1

Question 1:

a)



We need to find c and n_0 such that:

$3n^3+4n^2+2n <= cn^3$ for all $n >= n_0$

Divide both sides by n^3, getting

$3 + \frac{4}{n} + \frac{2}{n^2} \leq c$ for all $n >= n_0$

If we choose n_0 equal to 1, then we need a value such that:

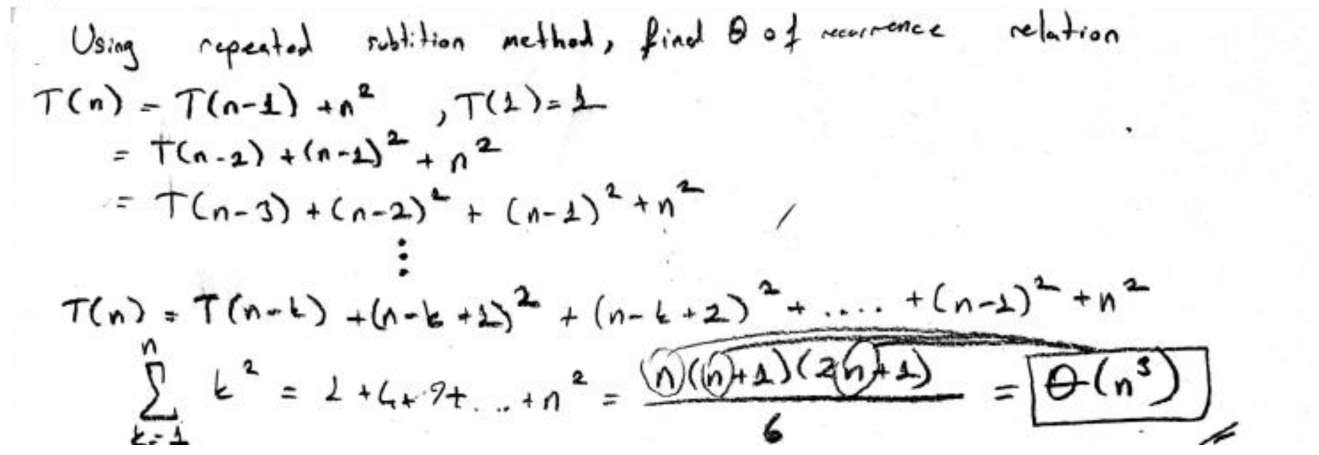
$3+4+2 \leq c$

We can set c equal to 9. Now we have

$3n^3+4n^2+2n <= 9n^3$ for all $n >= 1$, the equation applies.

b)

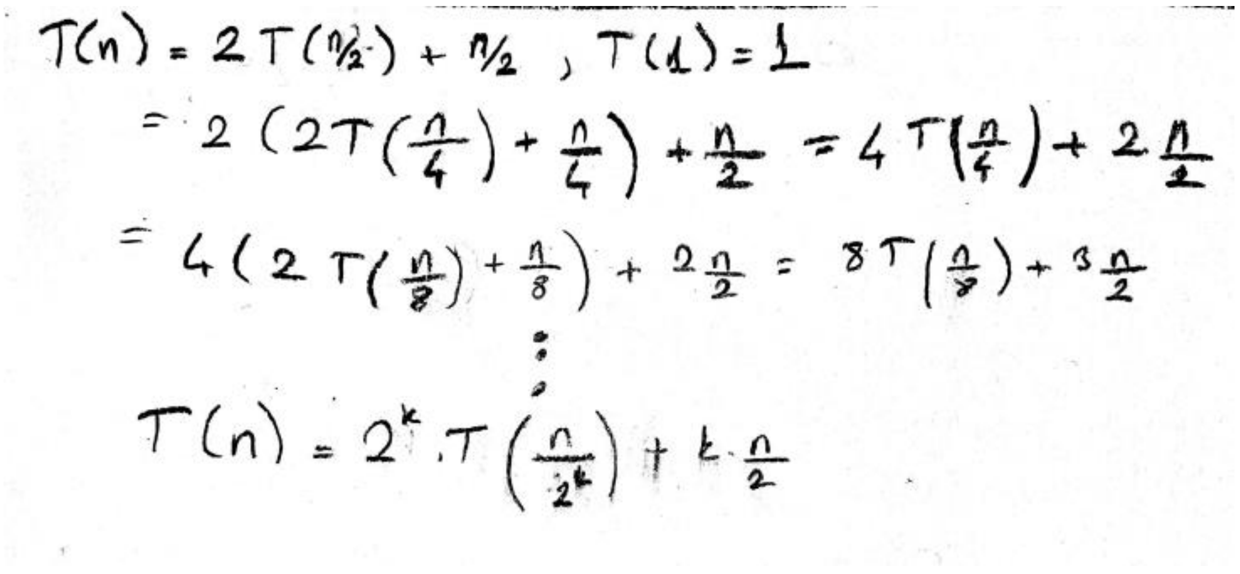
Part 1:



Using repeated subtition method, find Θ of recurrence relation

$$T(n) = T(n-1) + n^2 \quad , T(1) = 1$$
$$= T(n-2) + (n-1)^2 + n^2$$
$$= T(n-3) + (n-2)^2 + (n-1)^2 + n^2$$
$$\vdots$$
$$T(n) = T(n-k) + (n-k+1)^2 + (n-k+2)^2 + \dots + (n-1)^2 + n^2$$
$$\sum_{k=1}^{n} k^2 = 1 + 4 + 9 + \dots + n^2 = \frac{n(n+1)(2n+1)}{6} = \boxed{\Theta(n^3)}$$

Part 2:



$T(n) = 2T(n/2) + n/2 \quad , \quad T(1) = 1$

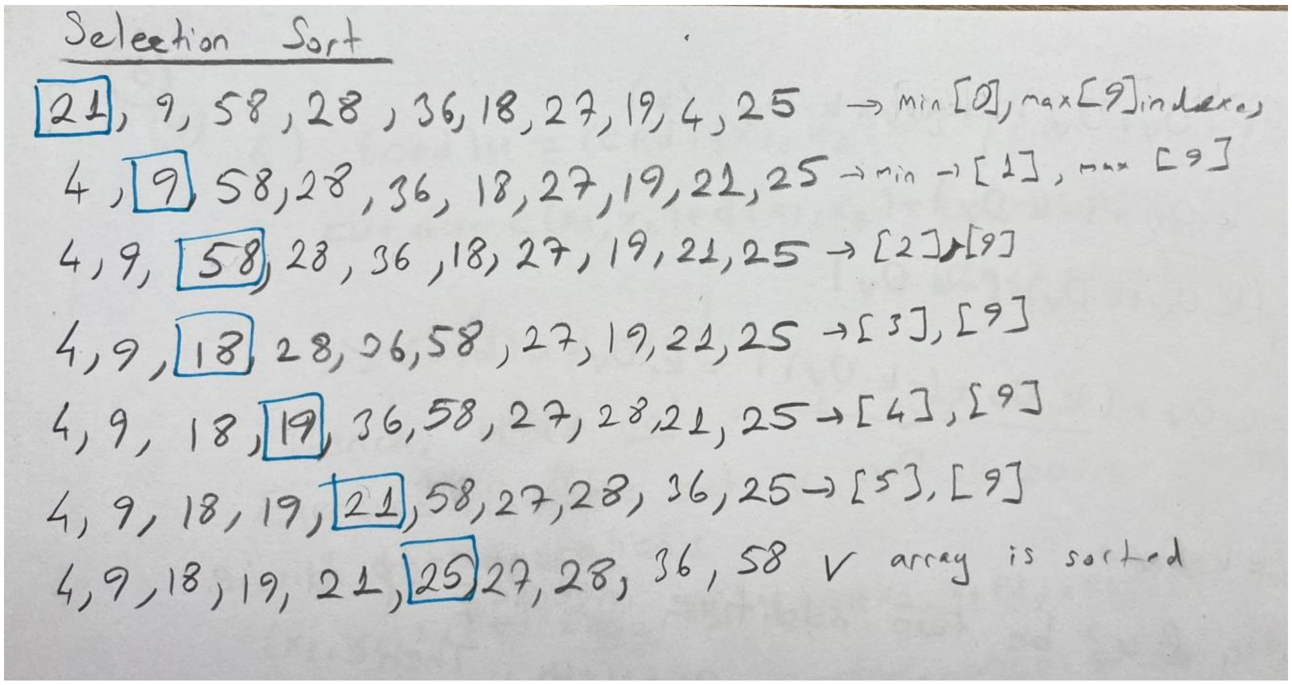
$= 2\left(2T\left(\frac{n}{4}\right) + \frac{n}{4}\right) + \frac{n}{2} = 4T\left(\frac{n}{4}\right) + 2\frac{n}{2}$

$= 4\left(2T\left(\frac{n}{8}\right) + \frac{n}{8}\right) + 2\frac{n}{2} = 8T\left(\frac{n}{8}\right) + 3\frac{n}{2}$

$\vdots$

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

c)



Selection Sort

$\boxed{21}$, 9, 58, 28, 36, 18, 27, 19, 4, 25 → min [0], max [9] indexes

4, $\boxed{9}$, 58, 28, 36, 18, 27, 19, 21, 25 → min → [1], max [9]

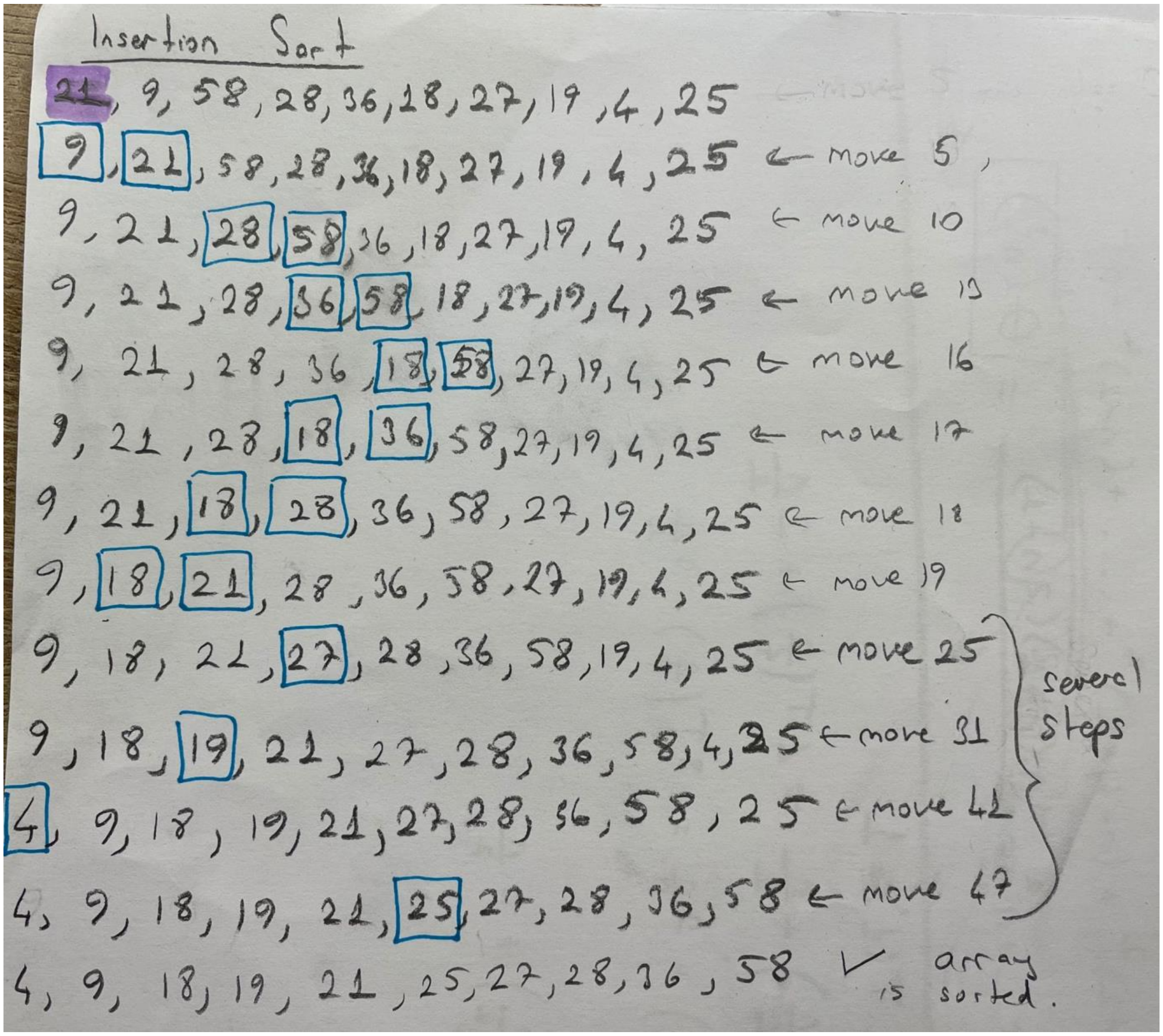
4, 9, $\boxed{58}$, 28, 36, 18, 27, 19, 21, 25 → [2] → [9]

4, 9, $\boxed{18}$, 28, 36, 58, 27, 19, 21, 25 → [3], [9]

4, 9, 18, $\boxed{19}$, 36, 58, 27, 28, 21, 25 → [4], [9]

4, 9, 18, 19, $\boxed{21}$, 58, 27, 28, 36, 25 → [5], [9]

4, 9, 18, 19, 21, $\boxed{25}$ 27, 28, 36, 58 ✓ array is sorted



Insertion Sort

21, 9, 58, 28, 36, 18, 27, 19, 4, 25

$\boxed{9}$, $\boxed{21}$, 58, 28, 36, 18, 27, 19, 4, 25 ← move 5,

9, 21, $\boxed{28}$, $\boxed{58}$, 36, 18, 27, 19, 4, 25 ← move 10

9, 21, 28, $\boxed{36}$, $\boxed{58}$, 18, 27, 19, 4, 25 ← move 13

9, 21, 28, 36, $\boxed{18}$, $\boxed{58}$, 27, 19, 4, 25 ← move 16

9, 21, 28, $\boxed{18}$, $\boxed{36}$, 58, 27, 19, 4, 25 ← move 17

9, 21, $\boxed{18}$, $\boxed{28}$, 36, 58, 27, 19, 4, 25 ← move 18

9, $\boxed{18}$, $\boxed{21}$, 28, 36, 58, 27, 19, 4, 25 ← move 19

9, 18, 21, $\boxed{27}$, 28, 36, 58, 19, 4, 25 ← move 25

9, 18, $\boxed{19}$, 21, 27, 28, 36, 58, 4, 25 ← move 31

$\boxed{4}$, 9, 18, 19, 21, 27, 28, 36, 58, 25 ← move 42

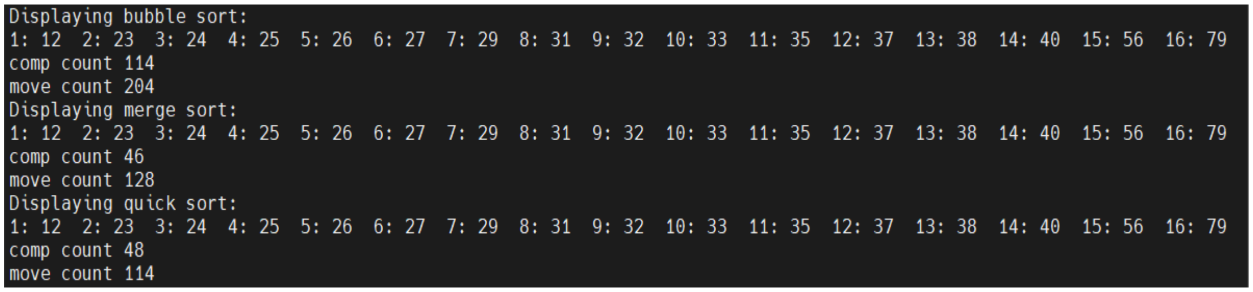
4, 9, 18, 19, 21, $\boxed{25}$, 27, 28, 36, 58 ← move 47

several steps

4, 9, 18, 19, 21, 25, 27, 28, 36, 58 ✓ array is sorted.

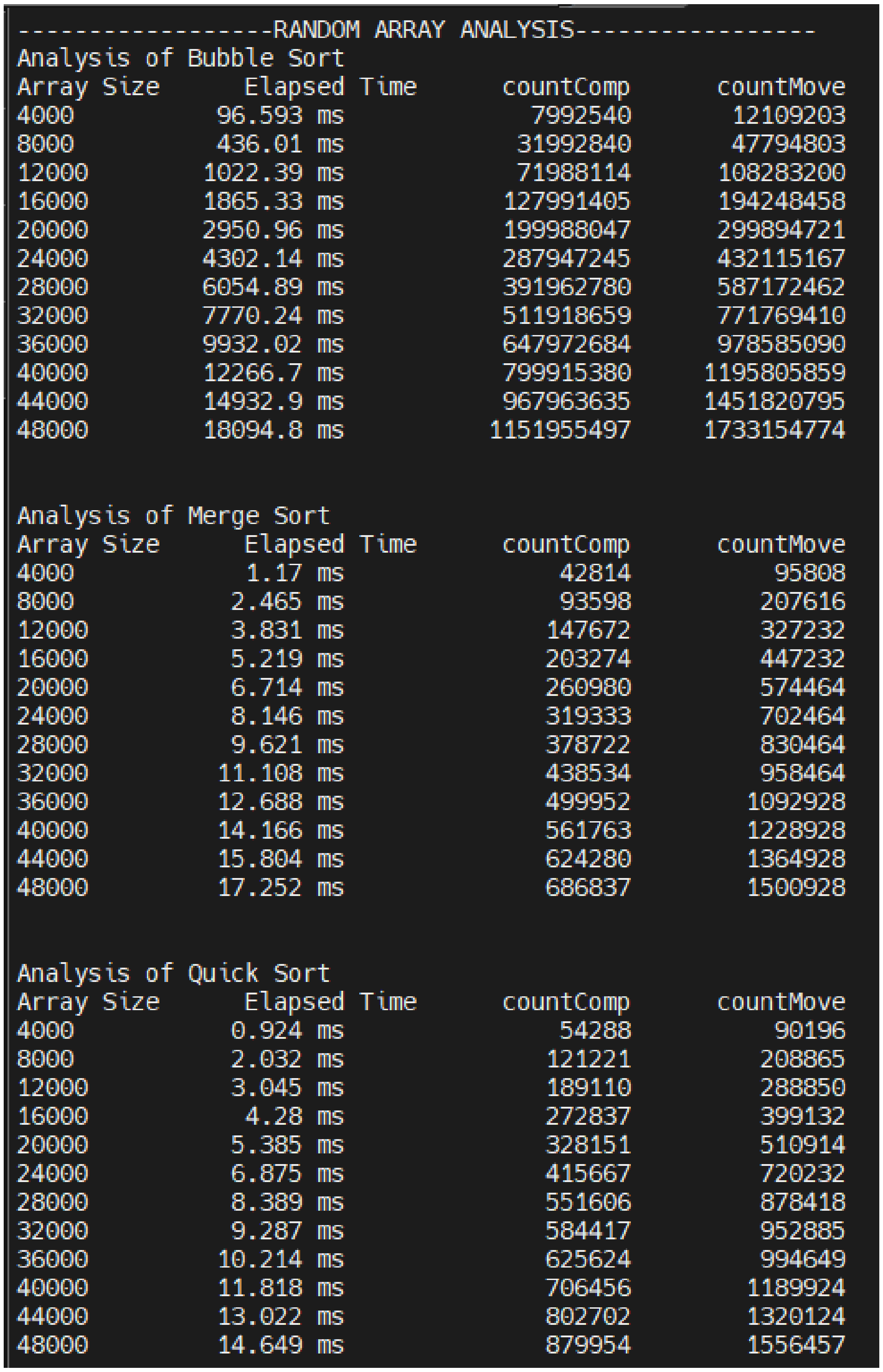
Question 2:

Displaying bubble, merge and quick sort according to given array:



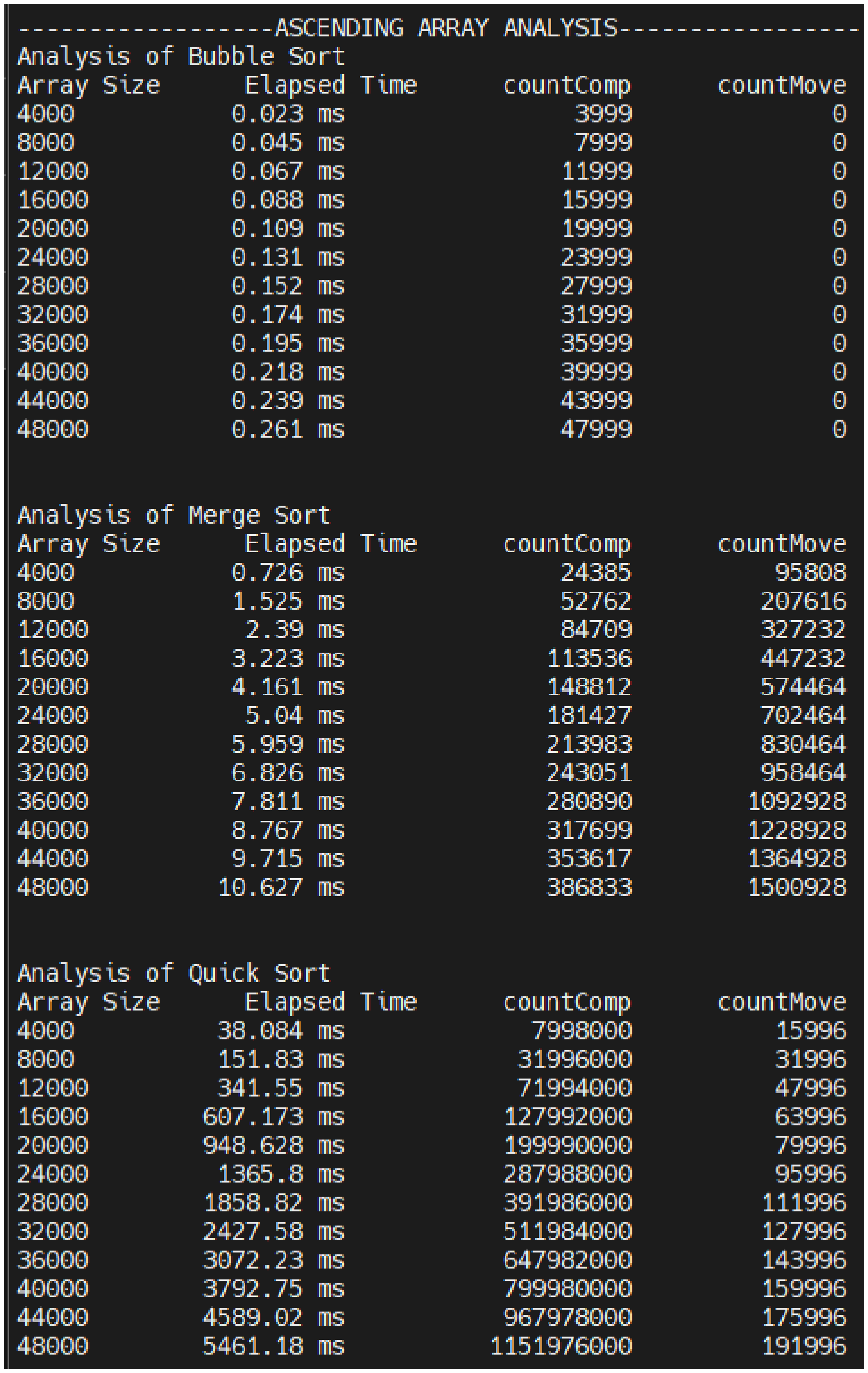
```
Displaying bubble sort:
1: 12  2: 23  3: 24  4: 25  5: 26  6: 27  7: 29  8: 31  9: 32  10: 33  11: 35  12: 37  13: 38  14: 40  15: 56  16: 79
comp count 114
move count 204
Displaying merge sort:
1: 12  2: 23  3: 24  4: 25  5: 26  6: 27  7: 29  8: 31  9: 32  10: 33  11: 35  12: 37  13: 38  14: 40  15: 56  16: 79
comp count 46
move count 128
Displaying quick sort:
1: 12  2: 23  3: 24  4: 25  5: 26  6: 27  7: 29  8: 31  9: 32  10: 33  11: 35  12: 37  13: 38  14: 40  15: 56  16: 79
comp count 48
move count 114
```

Displaying analysis of random arrays:



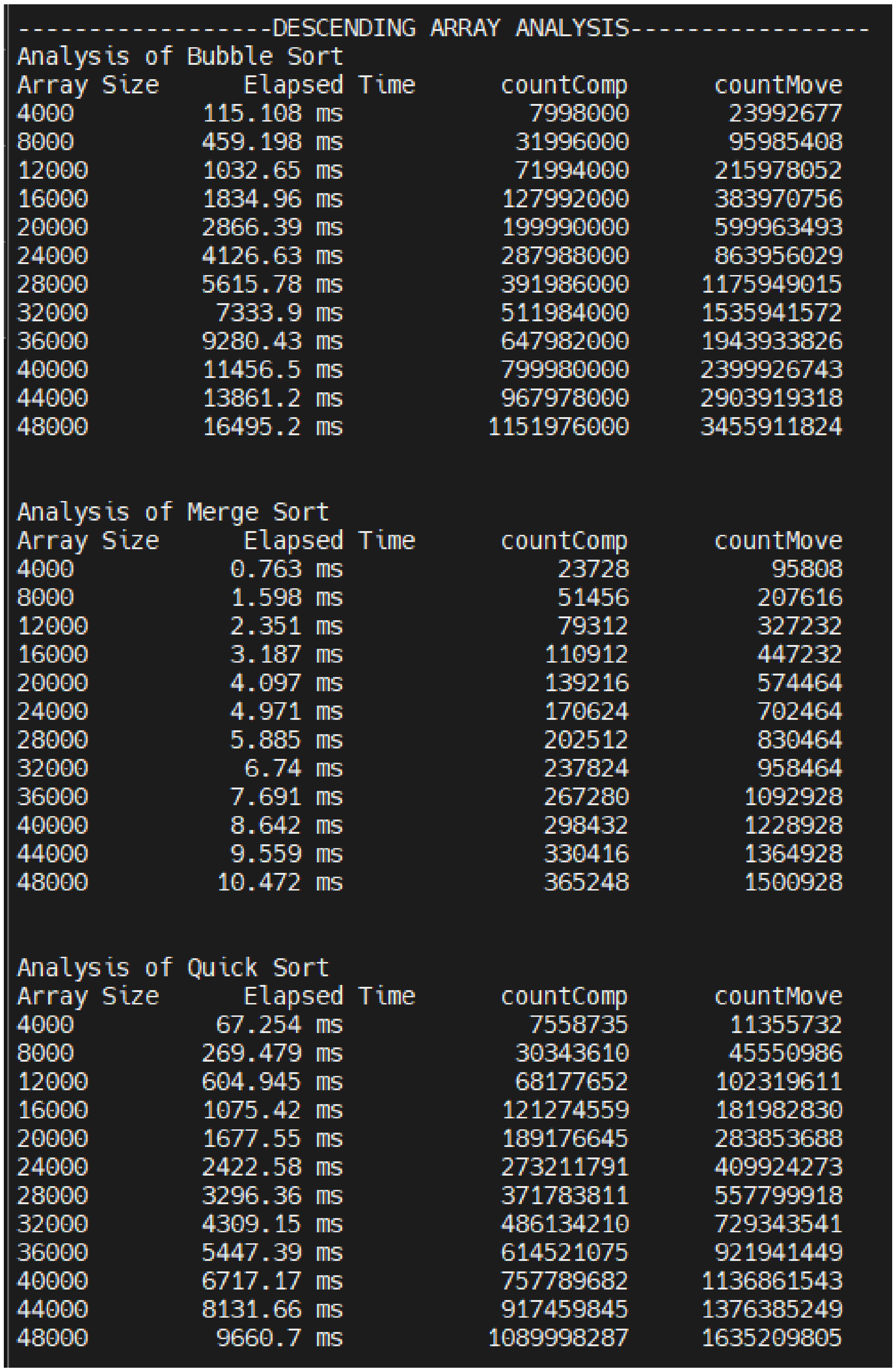
```
-------------------RANDOM ARRAY ANALYSIS-----------------
Analysis of Bubble Sort
Array Size      Elapsed Time      countComp      countMove
4000          96.593 ms           7992540       12109203
8000          436.01 ms          31992840       47794803
12000        1022.39 ms          71988114      108283200
16000        1865.33 ms         127991405      194248458
20000        2950.96 ms         199988047      299894721
24000        4302.14 ms         287947245      432115167
28000        6054.89 ms         391962780      587172462
32000        7770.24 ms         511918659      771769410
36000        9932.02 ms         647972684      978585090
40000        12266.7 ms         799915380     1195805859
44000        14932.9 ms         967963635     1451820795
48000        18094.8 ms        1151955497     1733154774


Analysis of Merge Sort
Array Size      Elapsed Time      countComp      countMove
4000            1.17 ms             42814          95808
8000           2.465 ms             93598         207616
12000          3.831 ms            147672         327232
16000          5.219 ms            203274         447232
20000          6.714 ms            260980         574464
24000          8.146 ms            319333         702464
28000          9.621 ms            378722         830464
32000         11.108 ms            438534         958464
36000         12.688 ms            499952        1092928
40000         14.166 ms            561763        1228928
44000         15.804 ms            624280        1364928
48000         17.252 ms            686837        1500928


Analysis of Quick Sort
Array Size      Elapsed Time      countComp      countMove
4000           0.924 ms             54288          90196
8000           2.032 ms            121221         208865
12000          3.045 ms            189110         288850
16000           4.28 ms            272837         399132
20000          5.385 ms            328151         510914
24000          6.875 ms            415667         720232
28000          8.389 ms            551606         878418
32000          9.287 ms            584417         952885
36000         10.214 ms            625624         994649
40000         11.818 ms            706456        1189924
44000         13.022 ms            802702        1320124
48000         14.649 ms            879954        1556457
```

Displaying analysis of ascending arrays:



```
------------------ASCENDING ARRAY ANALYSIS----------------
Analysis of Bubble Sort
Array Size      Elapsed Time      countComp      countMove
4000           0.023 ms              3999              0
8000           0.045 ms              7999              0
12000          0.067 ms             11999              0
16000          0.088 ms             15999              0
20000          0.109 ms             19999              0
24000          0.131 ms             23999              0
28000          0.152 ms             27999              0
32000          0.174 ms             31999              0
36000          0.195 ms             35999              0
40000          0.218 ms             39999              0
44000          0.239 ms             43999              0
48000          0.261 ms             47999              0


Analysis of Merge Sort
Array Size      Elapsed Time      countComp      countMove
4000           0.726 ms             24385          95808
8000           1.525 ms             52762         207616
12000           2.39 ms             84709         327232
16000          3.223 ms            113536         447232
20000          4.161 ms            148812         574464
24000           5.04 ms            181427         702464
28000          5.959 ms            213983         830464
32000          6.826 ms            243051         958464
36000          7.811 ms            280890        1092928
40000          8.767 ms            317699        1228928
44000          9.715 ms            353617        1364928
48000         10.627 ms            386833        1500928


Analysis of Quick Sort
Array Size      Elapsed Time      countComp      countMove
4000          38.084 ms           7998000          15996
8000          151.83 ms          31996000          31996
12000         341.55 ms          71994000          47996
16000        607.173 ms         127992000          63996
20000        948.628 ms         199990000          79996
24000         1365.8 ms         287988000          95996
28000        1858.82 ms         391986000         111996
32000        2427.58 ms         511984000         127996
36000        3072.23 ms         647982000         143996
40000        3792.75 ms         799980000         159996
44000        4589.02 ms         967978000         175996
48000        5461.18 ms        1151976000         191996
```

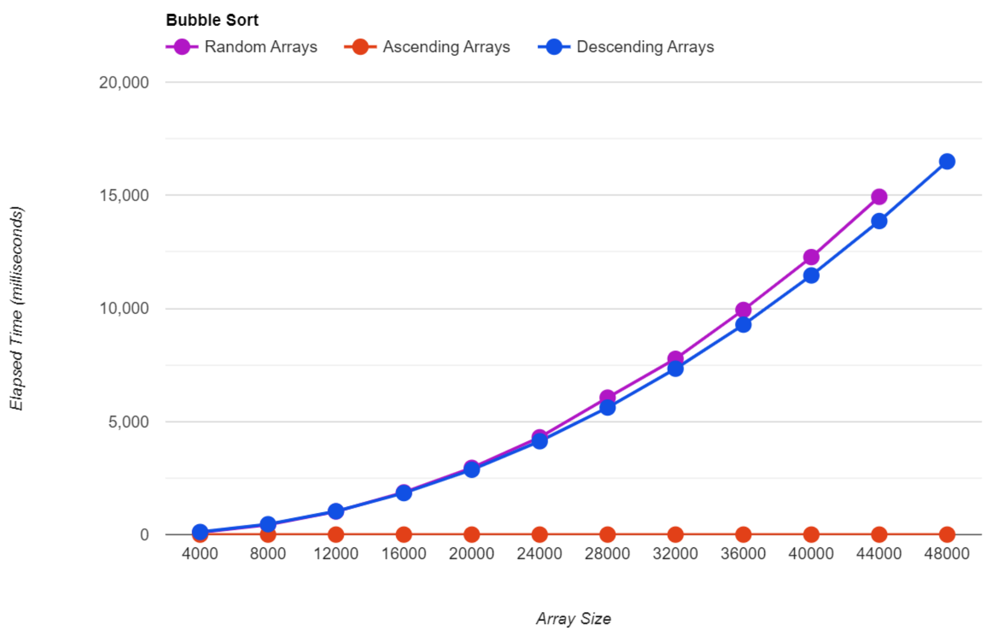
Displaying analysis of descending arrays:



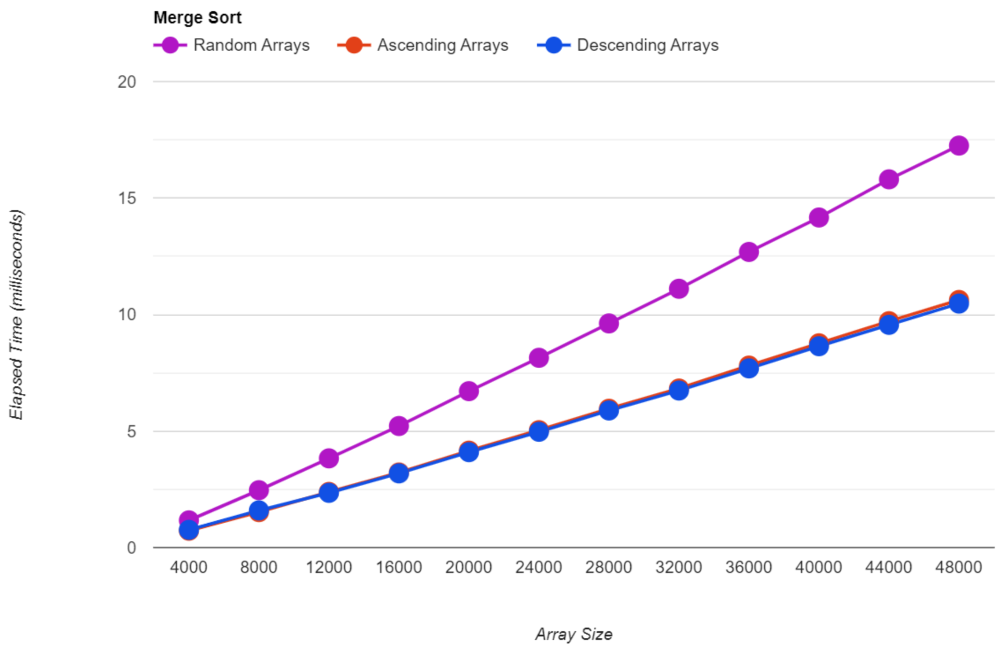
```
------------------DESCENDING ARRAY ANALYSIS-----------------
Analysis of Bubble Sort
Array Size      Elapsed Time      countComp       countMove
4000         115.108 ms             7998000        23992677
8000         459.198 ms            31996000        95985408
12000        1032.65 ms            71994000       215978052
16000        1834.96 ms           127992000       383970756
20000        2866.39 ms           199990000       599963493
24000        4126.63 ms           287988000       863956029
28000        5615.78 ms           391986000      1175949015
32000         7333.9 ms           511984000      1535941572
36000        9280.43 ms           647982000      1943933826
40000        11456.5 ms           799980000      2399926743
44000        13861.2 ms           967978000      2903919318
48000        16495.2 ms          1151976000      3455911824


Analysis of Merge Sort
Array Size      Elapsed Time      countComp       countMove
4000           0.763 ms               23728           95808
8000           1.598 ms               51456          207616
12000          2.351 ms               79312          327232
16000          3.187 ms              110912          447232
20000          4.097 ms              139216          574464
24000          4.971 ms              170624          702464
28000          5.885 ms              202512          830464
32000           6.74 ms              237824          958464
36000          7.691 ms              267280         1092928
40000          8.642 ms              298432         1228928
44000          9.559 ms              330416         1364928
48000         10.472 ms              365248         1500928


Analysis of Quick Sort
Array Size      Elapsed Time      countComp       countMove
4000          67.254 ms             7558735        11355732
8000         269.479 ms            30343610        45550986
12000        604.945 ms            68177652       102319611
16000        1075.42 ms           121274559       181982830
20000        1677.55 ms           189176645       283853688
24000        2422.58 ms           273211791       409924273
28000        3296.36 ms           371783811       557799918
32000        4309.15 ms           486134210       729343541
36000        5447.39 ms           614521075       921941449
40000        6717.17 ms           757789682      1136861543
44000        8131.66 ms           917459845      1376385249
48000         9660.7 ms          1089998287      1635209805
```

Question 3:

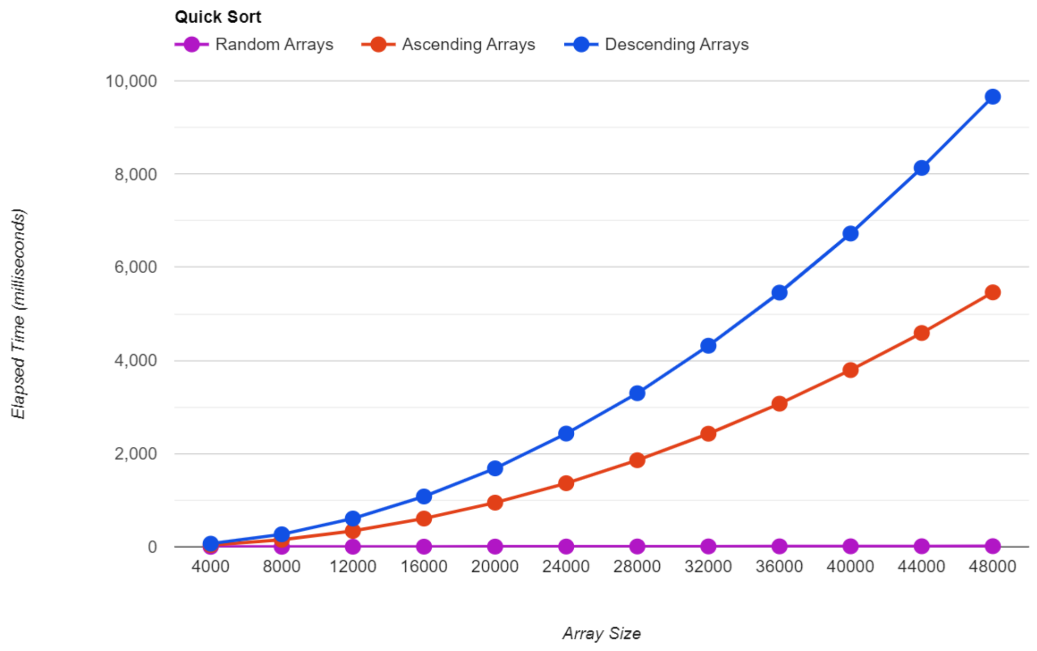
Bubble Sort Comparison



Merge Sort Comparison



Quick Sort Comparison



Bubble Sort: Bubble sort's best case is O(n) and worst case is O(n²). It is clearly observable that this theorem is true in my experiment. Best case scenario (already sorted) runs with O(n) time complexity with no moves and worst and average case scenarios runs with O(n²) time complexity.

Merge Sort: Merge sort's time complexity is O(n*logn) for all cases. Theoretically time complexities of all cases are the same. The results I found are a proof for that theorem. There is no notable difference between random, ascending and descending arrays.

Quick Sort: Quick sort's time complexity is O(n*logn) for average case and O(n²) for worst case. When array is descending, time needed for the function is maximum in my experiment. The experiment and theorem give similar results.

General Observation: Due to its rapidity, determining its duration was problematic because the calculations used to determine its lifetime were not very precise, and the data occasionally fluctuated. Also, random arrays some array random every time. Therefore, duration changes from run to run. For accurate results, we need to test the program more.