/**
* Title : Homework 1
* Author : Melike Demirci
* ID: 21702346
* Section : 2
* Assignment : 1
* Description : Question 1, 2 and 3
*/

Question 1:

(a) $f(n)$ = 20 n^4 + 20 n^2 + 5 is $O(n^5)$ by the Big-O definition, if $f(n) \leq cn^5$ for some $n \geq n_0$. If 20 n^4 + 20 n^2 + 5 ≤ cn^5 then $\frac{20}{n} + \frac{20}{n^3} + \frac{5}{n^5} \leq c$. Therefore, this condition holds for $n \geq n_0$ = 1 and c ≥ 45 = (20 + 20 + 5).

(b) Sort the array [18, 4, 47, 24, 15, 24, 17, 11, 31, 23] in ascending with selection and bubble sort.

Selection sort:

Note: "**|**" separates the sorted and unsorted part.

Initial array: [18, 4, 47, 24, 15, 24, 17, 11, 31, 23]

After 1st swap: [18, 4, **23**, 24, 15, 24, 17, 11, 31 **| 47**]

After 2nd swap: [18, 4, 23, 24, 15, 24, 17, 11 **| 31**, 47]

After 3rd swap: [18, 4, 23, **11**, 15, 24, 17 **| 24**, 31, 47]

After 4th swap: [18, 4, 23, 11, 15, **17 | 24**, 24, 31, 47]

After 5th swap: [18, 4, **17**, 11, 15**| 23**, 24, 24, 31, 47]

After 6th swap: [**15**, 4, 17, 11 **| 18**, 23, 24, 24, 31, 47]

After 7th swap: [15, 4, **11| 17**, 18, 23, 24, 24, 31, 47]

After 8th swap: [**11**, 4**| 15,** 17, 18, 23, 24, 24, 31, 47]

After 9th swap: [**4| 11**, 15, 17, 18, 23, 24, 24, 31, 47]

Sorted array: [4, 11, 15, 17, 18, 23, 24, 24, 31, 47]

Bubble sort:
Note: "|" separates the sorted and unsorted part.
Initial array: [18, 4, 47, 24, 15, 24, 17, 11, 31, 23]

Pass 1

Initial array: [**18, 4**, 47, 24, 15, 24, 17, 11, 31, 23|]

[4, **18, 47**, 24, 15, 24, 17, 11, 31, 23]

[4, 18, **24, 47**, 15, 24, 17, 11, 31, 23]

[4, 18, 24, **15, 47**, 24, 17, 11, 31, 23]

[4, 18, 24, 15, **24, 47**, 17, 11, 31, 23]

[4, 18, 24, 15, 24, **17, 47**, 11, 31, 23]

[4, 18, 24, 15, 24, 17, **11, 47**, 31, 23]

[4, 18, 24, 15, 24, 17, 11, **31, 47**, 23]

[4, 18, 24, 15, 24, 17, 11, 31, 23 |**47**]

Pass 2

[**4, 18**, 24, 15, 24, 17, 11, 31, 23 |47]

[4, **18, 24**, 15, 24, 17, 11, 31, 23 |47]

[4, 18, **24, 15**, 24, 17, 11, 31, 23 |47]

[4, 18, 15, **24, 24**, 17, 11, 31, 23 |47]

[4, 18, 15, 24, **24, 17**, 11, 31, 23 |47]

[4, 18, 15, 24, 17, **24, 11**, 31, 23 |47]

[4, 18, 15, 24, 17, 11, **24, 31**, 23 |47]

[4, 18, 15, 24, 17, 11, 24, **31, 23** |47]

[4, 18, 15, 24, 17, 11, 24, 23 | **31, 47**]

Pass 3

[**4, 18**, 15, 24, 17, 11, 24, 23 | 31, 47]

[4, **18, 15**, 24, 17, 11, 24, 23 | 31, 47]

[4, 15, **18, 24**, 17, 11, 24, 23 | 31, 47]

[4, 15, 18, **24, 17**, 11, 24, 23 | 31, 47]

[4, 15, 18, 17, **24, 11**, 24, 23 | 31, 47]

[4, 15, 18, 17, 11, **24, 24**, 23 | 31, 47]

[4, 15, 18, 17, 11, 24, **24, 23** | 31, 47]

[4, 15, 18, 17, 11, 24, 23| **24, 31, 47**]

Pass 4

[**4, 15**, 18, 17, 11, 24, 23| 24, 31, 47]

[4, **15, 18**, 17, 11, 24, 23| 24, 31, 47]

[4, 15, **18, 17**, 11, 24, 23| 24, 31, 47]

[4, 15, 17, **18, 11**, 24, 23| 24, 31, 47]

[4, 15, 17, 11, **18, 24**, 23| 24, 31, 47]

[4, 15, 17, 11, 18, **24, 23**| 24, 31, 47]

[4, 15, 17, 11, 18, 23| **24, 24, 31, 47**]

Pass 5

[**4, 15**, 17, 11, 18, 23| 24, 24, 31, 47]

[4, **15, 17**, 11, 18, 23| 24, 24, 31, 47]

[4, 15, **17, 11**, 18, 23| 24, 24, 31, 47]

[4, 15, 11, **17, 18**, 23| 24, 24, 31, 47]

[4, 15, 11, 17, **18, 23**| 24, 24, 31, 47]

[4, 15, 11, 17, 18 | **23, 24, 24, 31, 47**]

Pass 6

[**4, 15**, 11, 17, 18 | 23, 24, 24, 31, 47]

[4, **15, 11**, 17, 18 | 23, 24, 24, 31, 47]

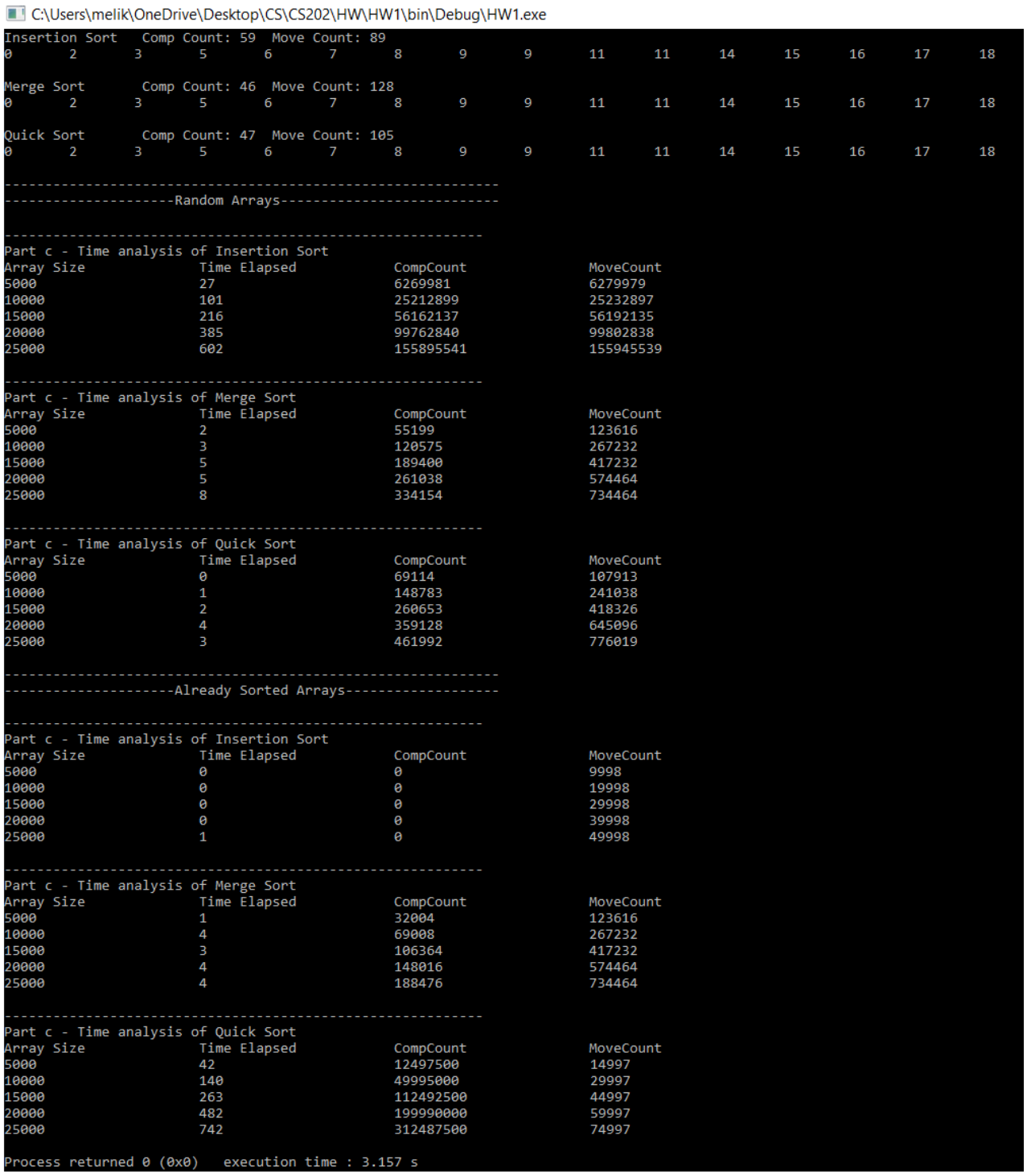
[4, 11, **15, 17**, 18 | 23, 24, 24, 31, 47]

[4, 11, 15, **17, 18** | 23, 24, 24, 31, 47]

[4, 11, 15, 17 |**18, 23, 24, 24, 31, 47**]

Sorted array: [4, 11, 15, 17, 18, 23, 24, 24, 31, 47]

Question 2:



```
C:\Users\melik\OneDrive\Desktop\CS\CS202\HW\HW1\bin\Debug\HW1.exe
Insertion Sort   Comp Count: 59  Move Count: 89
0       2       3       5       6       7       8       9       9       11      11      14      15      16      17      18

Merge Sort       Comp Count: 46  Move Count: 128
0       2       3       5       6       7       8       9       9       11      11      14      15      16      17      18

Quick Sort       Comp Count: 47  Move Count: 105
0       2       3       5       6       7       8       9       9       11      11      14      15      16      17      18

------------------------------------------------------------
--------------------Random Arrays---------------------------

------------------------------------------------------------
Part c - Time analysis of Insertion Sort
Array Size              Time Elapsed            CompCount               MoveCount
5000                    27                      6269981                 6279979
10000                   101                     25212899                25232897
15000                   216                     56162137                56192135
20000                   385                     99762840                99802838
25000                   602                     155895541               155945539

------------------------------------------------------------
Part c - Time analysis of Merge Sort
Array Size              Time Elapsed            CompCount               MoveCount
5000                    2                       55199                   123616
10000                   3                       120575                  267232
15000                   5                       189400                  417232
20000                   5                       261038                  574464
25000                   8                       334154                  734464

------------------------------------------------------------
Part c - Time analysis of Quick Sort
Array Size              Time Elapsed            CompCount               MoveCount
5000                    0                       69114                   107913
10000                   1                       148783                  241038
15000                   2                       260653                  418326
20000                   4                       359128                  645096
25000                   3                       461992                  776019

------------------------------------------------------------
--------------------Already Sorted Arrays-------------------

------------------------------------------------------------
Part c - Time analysis of Insertion Sort
Array Size              Time Elapsed            CompCount               MoveCount
5000                    0                       0                       9998
10000                   0                       0                       19998
15000                   0                       0                       29998
20000                   0                       0                       39998
25000                   1                       0                       49998

------------------------------------------------------------
Part c - Time analysis of Merge Sort
Array Size              Time Elapsed            CompCount               MoveCount
5000                    1                       32004                   123616
10000                   4                       69008                   267232
15000                   3                       106364                  417232
20000                   4                       148016                  574464
25000                   4                       188476                  734464

------------------------------------------------------------
Part c - Time analysis of Quick Sort
Array Size              Time Elapsed            CompCount               MoveCount
5000                    42                      12497500                14997
10000                   140                     49995000                29997
15000                   263                     112492500               44997
20000                   482                     199990000               59997
25000                   742                     312487500               74997

Process returned 0 (0x0)   execution time : 3.157 s
```

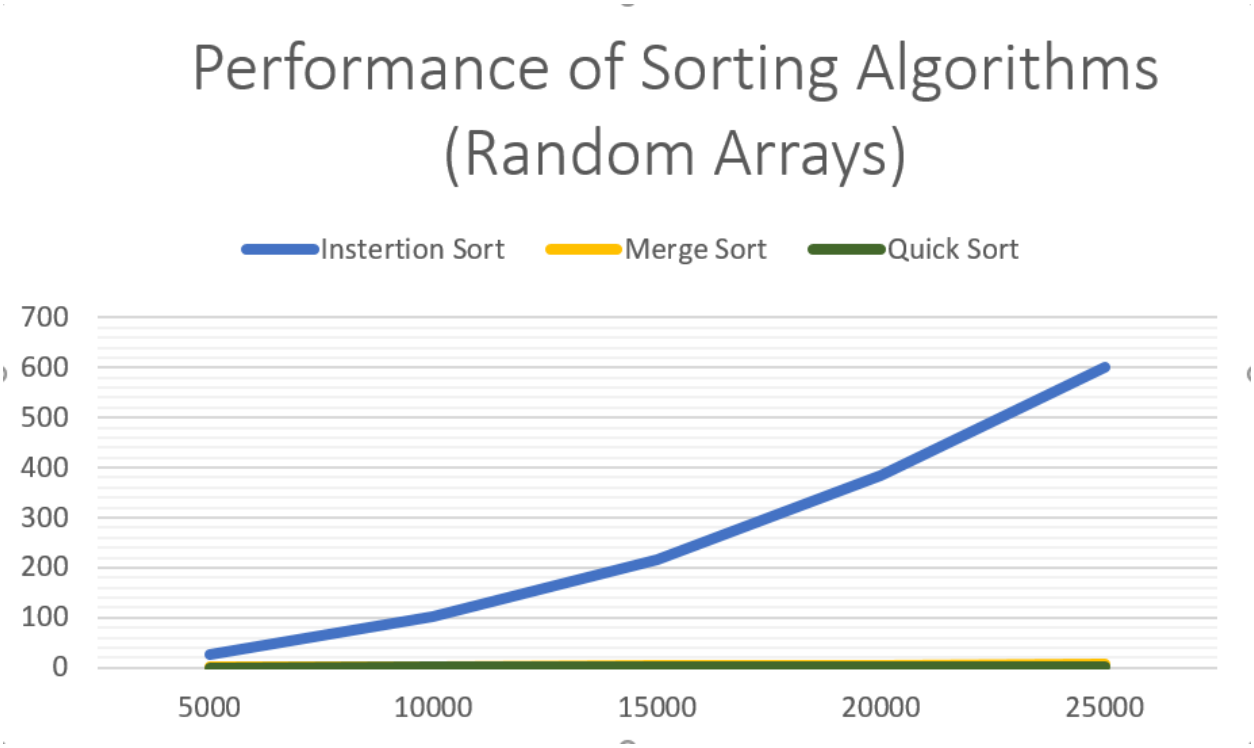


Figure 1: Performance of the Sorting Algorithms (Random Arrays)

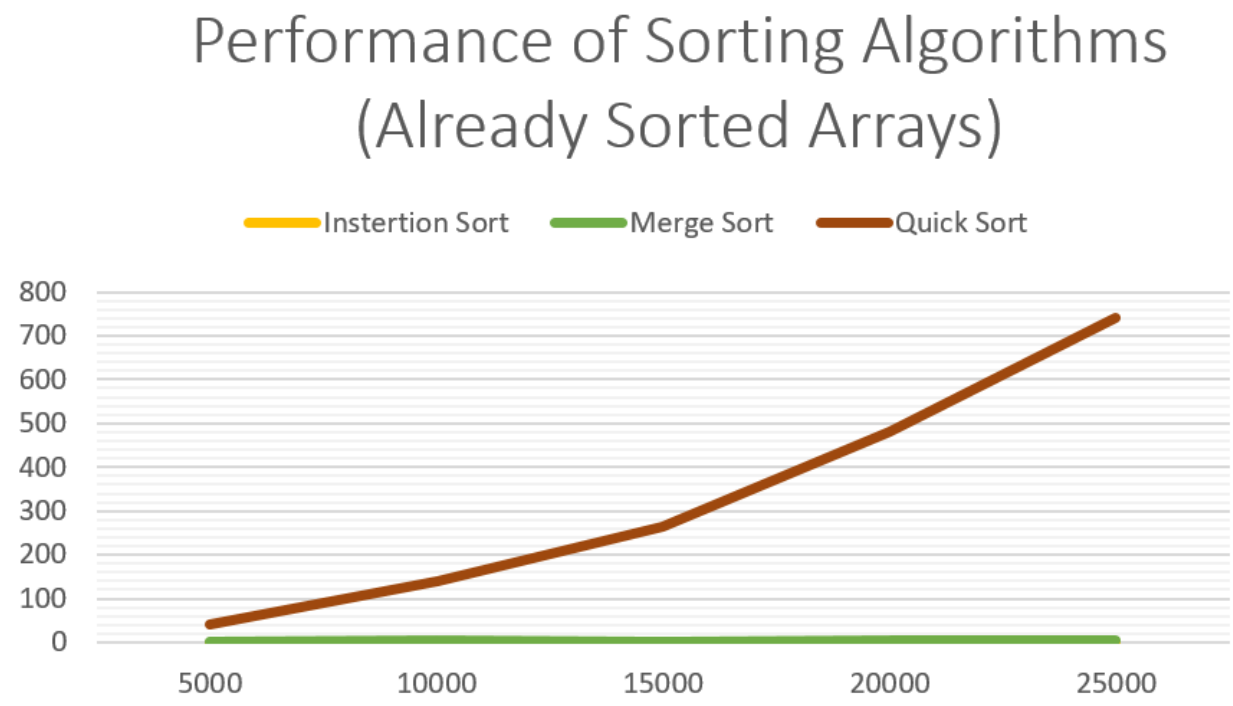


Figure 2: Performance of the Sorting Algorithms (Already Sorted Arrays)

Algorithm	Time Complexities		
	Best	Average	Worst
Insertion Sort	$\Omega(n)$	$\theta(n^2)$	$O(n^2)$
Merge Sort	$\Omega(nlog(n))$	$\theta(nlog(n))$	$O(nlog(n))$
Quick Sort	$\Omega(nlog(n))$	$\theta(nlog(n))$	$O(n^2)$

Table 1: Time Complexities of the Sorting Algorithms

From the data it can be observed that insertion sort worked in $O(n^2)$ for random arrays but worked in o(n) for the already sorted arrays. Time elapsed is not clear for the insertion sort on already sorted arrays but it can be observed from the number of moves. So the experiment shows that for the insertion sort the best case is when array is already sorted.

Another observation is that merge sort worked in O(nlog(n)) for both random and already sorted arrays. It can said that the empirical result and the theoretical result are same.

Quick sort worked in $O(n^2)$ for already sorted array and worked in O(nlog(n)) for random arrays. Theoretically, it is obvious that already sorted arrays are the worst case of the quick sort when pivot is the first element. In quick sort selecting the pivot is very important, when pivot is selected near to the average it works in O(nlog(n)) however in our example pivot is always the first element and in the already sorted array first element is the smallest. The smallest element should not be selected as a pivot because it prohibit dividing the array two balanced sub arrays.

Question 3:

When the experiment results observed it can be seen that elapsed time of insertion sort is directly proportional to the K. This result is reasonable because when K increases, the move and comparison counts also increases. Move count and comparison count are the two things that determines the elapsed time.

For the merge and quick sort, it looks like the change of K has no significant effect on elapsed time. Both of them works in O(nlog(n)) as always. When K = 0, which means that array is already sorted, elapsed time for quick sort is very high because first element is being selected as the pivot. This cause unbalanced subarrays for the quick sort and it cause the algorithm works in $O(n^2)$.

```
**************************************Question 3**********************************
Size: 5000
K               Insertion Sort          Merge Sort              Quick Sort
0                       0                       4                       30
500                     4                       4                       1
1000                    7                       2                       1
1500                    9                       0                       0
2000                    18                      1                       0
2500                    20                      4                       0
3000                    26                      0                       0
3500                    42                      0                       4
4000                    45                      0                       4
4500                    38                      4                       0
Size: 10000
K               Insertion Sort          Merge Sort              Quick Sort
0                       0                       1                       138
1000                    11                      7                       0
2000                    33                      4                       4
3000                    83                      4                       0
4000                    92                      0                       4
5000                    58                      0                       4
6000                    76                      2                       2
7000                    129                     4                       0
8000                    171                     4                       0
9000                    197                     8                       4
Size: 15000
K               Insertion Sort          Merge Sort              Quick Sort
0                       0                       10                      304
1500                    56                      11                      4
3000                    56                      1                       6
4500                    91                      4                       0
6000                    178                     2                       5
7500                    138                     6                       2
9000                    166                     4                       2
10500                   186                     4                       3
12000                   247                     2                       4
13500                   254                     2                       4
Size: 20000
K               Insertion Sort          Merge Sort              Quick Sort
0                       0                       7                       496
2000                    73                      7                       4
4000                    102                     6                       4
6000                    148                     7                       3
8000                    179                     4                       4
10000                   204                     4                       4
12000                   236                     4                       4
14000                   314                     4                       6
16000                   355                     4                       4
```