Leminur Çelik - 150190085

BLG 335E – ANALYSIS OF ALGORITHMS 1

HOMEWORK 1

Part 2 - Report

a. How to Run

First of all, the code I wrote must be compiled. Then, the code is run with the csv file taken as input. In this homework, the books.csv file is taken as input. It is saved in another text file after sorting by the books' average rating values. According to the code I wrote, the newly sorted books are saved in the new_books.txt file. As output, the average rating values before sorting and after sorting are printed. In addition to this, the execution time of the algorithm and the number of partitionings are printed.

b. After reading the data from the books.csv file and sorting it with the ascending oder according to the average rating values, the sorted result was written into the sorted_books.txt file.

c. User is actual CPU time used in executing the process. Therefore, we look at the user part.
The number of partitions and swaps and execution time of full data

```
Number of partitionings: 10945
Number of swaps: 91813

real    0m3.388s
user    0m0.376s
sys     0m0.031s
```

The number of partitions and execution time of half of the data. Since the number of data was halved, the number of iterations decreased accordingly. At the same time, the execution time was less than half.

```
Number of partitionings: 5399
Number of swaps: 49183

real    0m3.022s
user    0m0.126s
sys     0m0.019s
```

The number of partitions and execution time of quarter of the data. Since the number of data decreased to a quarter, the number of iterations decreased accordingly. At the same time, execution time was less than a quarter.

```
Number of partitionings: 2633
Number of swaps: 23001

real    0m2.282s
user    0m0.062s
sys     0m0.000s
```

As the number of iterations decreases, there is a significant change in execution time.

d. Best case: O(nlog(n))

Average case: O(nlog(n))

Worst case: O(n^2)

In order to be the best case, the middle value is selected as the pivot. Therefore, in the partition part, the array is divided into exactly two, so the recurrence equation of the best case is T(n) = 2T(n/2) + n.

Using substitution method,

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

By substituting T(n/2) in the first relationship, we obtain:

T(n) = 2^2T(n/2^2) + 2 * n/2 + n = 2^2T(n/2) + n

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

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

…

T(n) = 2^kT(n/2^k) + k * n

T(n/2^k) = 2T(n/2^(k+1)) + n/2^k

By substituting T(n/2^(k+1)) in the relationship above, we obtain:

T(n) = 2^(k+1) * T(n/2^(k+1)) + (k+1) * n

T(n) = 2^kT(n/2^k) + k * n → when n/2^k = 1, k = logn

T(n) = n * T(1) + n * logn → T(n) = O(nlogn)

In order to be the average case, pivot should be selected except for small, large and medium values. For example, when partition puts O(n/4) elements in one set and O(3n/4) elements in other set, the recurrence equation is T(n) = T(n/4) + T(3n/4) + n. After the next steps, it becomes T(n) = O (nlog(n)) as in the best case.

In order to be the worst case, the smallest or largest value is selected as the pivot. Therefore, in the partition part array splits 1 value on one side and n-1 values on the other side, so the recurrence equation of the worst case is T(n) = T(n-1) + n.

Using substitution method,

T(n-1) = T(n-2) + n – 1

By substituting T(n-1) in the first relationship, we obtain:

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

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

By substituting T(n-2) in the relationship above, we obtain:

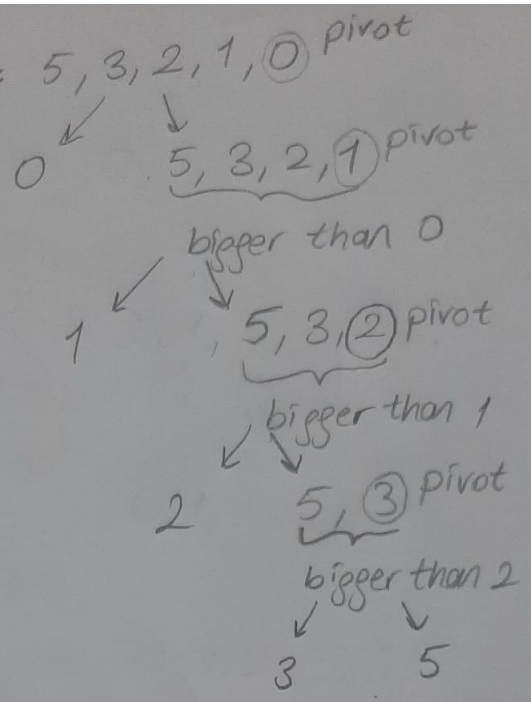
$T(n) = T(n-3) + n - 2 + n - 1 + n$

$T(n) = T(n - k) + n - (k - 1) + n - (k - 2) + \ldots + n \rightarrow$ when $n - k = 1$, $k = n - 1$

$T(n) = T(1) + 2 + \ldots + n - 1 + n \rightarrow T(n) = n (n + 1) / 2 - 1 \rightarrow T(n) = O(n^2)$

e. If the pivot is selected as the smallest or the largest value, only one element on one side is left in the partitioning part, and all the elements except that one element remain on the other side. Therefore, n-1 elements are dealt with again which makes the worst case n^2. By doing random pivoting, we can reduce the execution time to nlogn in some cases. Besides, merge sort can be used to reduce time complexity. Merge sort has nlogn time complexity in all cases.

e. First Scenario : 5, 3, 2, 1, (0) pivot



Second Scenario : 3, 2, 1, 0, (5) pivot