CS301 Assignment 2

Ataollah Hosseinzadeh Fard (ID: 28610)

9 November 2022

1 Problem 1

1.1 Part a)

As algorithm we can use merge sort which has average run-time of $O(n * logn)$.
Then we will have a set of sorted numbers so to get list k smallest numbers, we have to get a subset of sorted set so it will take $O(k)$.
In total it will take $O(n * logn + k)$ which is $O(n * logn)$.

1.2 Part b)

if we use an order-statistics algorithm to find Kth smallest number in set, it will take $O(n)$. then we can chose that Kth smallest number as pivot and partition the set around pivot which will cost $O(n)$. then we can take the pivot and partition in its leftside as a new set and then with using merge sort, we can sort the set. This merge operation will take $O(k * logk)$.
In total it will cost us as $O(n) + O(n) + O(k * logk)$ which will be $O(k * logk + 2n)$.

1.3 Which method would you use? Please explain why.

The second approach, because seems faster since in the second approach, the number of elements in set to be merge is less than first approach, so in total less workload.

2 Problem 2

2.1 Part a)

Normally in Radix sorting algorithm while sorting numbers in each step we compare digits of same index, we can do the same but there is 2 new things, first in number sorting each digit is [0, 10] but in sorting strings each character is between ['A', 'Z'] we consider only upper-cases, so our rage is extended. Also in numbers comparing 100 and 10 is easy but in strings we should make sure that all strings have same size, so first we have to find the longest string and then add extra '-' to end of other strings until all other strings have same size as longest string. Then, we start the algorithm, in each step we compare characters based in their order in Alphabet or we can consider their ASCII codes, the remaining is same as Radix sort, and at the end we will have a sorted list of strings.

2.2 Part b)

input: [“BATURAY”, “GORKEM”, “GIRAY”, “TAHIR”, “BARIS”]

we find longest string in O(n), which is “BATURAY” with size of 7.
Adding ’-’ to ends, [“BATURAY”, “GORKEM-”, “GIRAY–”, “TAHIR–”, “BARIS–”]
Sorting by 6th index [“GORKEM-”, “GIRAY-”, “TAHIR-”, “BARIS-”, “BATURAY”]
Sorting by 5th index [“GIRAY-”, “TAHIR-”, “BARIS-”, “BATURAY”, “GORKEM-”]
Sorting by 4th index [“GORKEM-”, “TAHIR–”, “BATURAY”, “BARIS–”, “GIRAY–”]
Sorting by 3th index [“GIRAY–”, “TAHIR–”, “BARIS–”, “GORKEM-”, “BATURAY”]
Sorting by 2th index [“TAHIR–”, “GIRAY–”, “BARIS–”, “GORKEM-”, “BATURAY”]
Sorting by 1th index [“TAHIR–”, “BARIS–”, “BATURAY”, “GIRAY–”, “GORKEM-”]
Sorting by 0th index [“BARIS–”, “BATURAY”, “GIRAY–”, “GORKEM-”, “TAHIR–”]
Remove extra ’-’s from ends, [“BARIS”, “BATURAY”, “GIRAY”, “GORKEM”, “TAHIR”]

2.3 Part c)

Finding the longes string will cost as $O(df)$ and d is size of longest string and f is algorithm. The run time of Radix sort which is a type of counting sort is $O(n + k)$ which k is the base number of each step, in sorting numbers it is 10 since we have 10 digits but in sorting strings we have to consider whole alphabet so k is 27. With our modification run time of new algorithm will be $O(d(n + k))$. In this questions input, program will run in $O(7(n + 27))$.