Sabancı University
Faculty of Engineering and Natural Sciences

CS301 – Algorithms

Homework 2

Due: November 03, 2020 @ 23.59
(Upload to SUCourse+ - **no late submission**)

PLEASE NOTE:

- Provide only the requested information and nothing more. Unreadable, unintelligible and irrelevant answers will not be considered.
- You can collaborate with your `TA/INSTRUCTOR ONLY` and discuss the solutions of the problems. However you have to write down the solutions on your own.
- Plagiarism will not be tolerated.
- Submit your answers as a single pdf file named as username-yourname-surname-hw2.pdf
- Example submission file name: furkanreha-furkanreha-tutas-hw2.pdf
- Late submission is allowed for only 10 hours. Each hour late submission costs 10% of your grade. For example, if you submit 5 minutes late, your grade will be multiplied by 0.9, if you submit 61 minutes late, your grade will be multiplied by 0.8.

1. Consider the Radix sort that is used to sort numbers. (55pt)
   (a) Explain how would you modify the Radix sort in order to sort the list of words in alphabetical order. (20pt)
   (b) Sort the following list of words using the modified algorithm and give the form of the array after every iteration of the main loop of the radix sort. (20pt)
   $[HUSNU, BERK, FURKAN, REHA]$
   (c) Find the time complexity of the modified algorithm. (15pt)

2. Consider the following algorithm, which is exactly the same algorithm we considered in the class for the selection problem with one difference; instead of dividing the input into groups of 5 elements, we divide the input into groups of k elements, where k is a parameter of the algorithm. (45pt)

```
1  float WCL_Select (A, first, last, i, k) {
2      if (first == last) { // i=1 in this case
3          return A[first];
4          }
5
6      Divide n elements into groups of k elements; //g_1 g_2 ... g_{n/k}
7      Find the median m_i each group g_i;
8      Use WCL_Select to find the median m_m of the medians of all the groups;
9      mid = NewPartition(A, first, last, m_m); // it partitions around m_m
10     mid_and_less = mid - first + 1
11
12     if (mid_and_less == i) { // we may be lucky
13         return A[mid];
14         }
15     if (i < mid_and_less) { // it is in the left subarray
16         return (WCL_Select(A, first, mid-1, i, k));
17         }
18
19     return (WCL_Select(A, mid+1, last, i-mid_and_less, k));
20     }
```

(a) Derive the recurrence for the running time of the algorithm given above. Note that the recurrence will be parametric in k as well. After you derive this recurrence, substituting $k = 5$ in the recurrence should give the recurrence we had in the class. (20pt)

(b) Write the concrete recursion for $k = 7$. Show that running time for this recursion is still linear. (10pt)

(c) Find the largest odd integer value for k, for which the running time of this algorithm is super-linear (i.e. grows faster than linear). Write the concrete recursion for the value of k you suggest and show that running time is super-linear in this case. (15pt)