Homework 4 (problem set)

Requirements:

- Submit a .pdf file, all the solutions must be typed. Hand-written solutions will not be graded.
- Students can either work alone or in pairs. If you have a teammate, then only one of you need to submit.

Student name(s):

1. Illustrate the operation of COUNTING-SORT on the array A = 6, 7, 2, 6, 1, 3, 4, 6, 1, 3, 2

   Use the algorithm in the lectures/textbook. Especially, please show how the values in the array C are filled and how the values are "dispatched" from A to B.

2. In the counting sort we have seen in the lecture, there 4th iteration is:
   `for j=n downto 1`

   If we change it to:
   `for j=1 upto n`

   Does this algorithm still provide the correct output? Illustrate your reasoning with this example: ***A={4 1 3 4 3}, k = 4.***

   What problem (if any) will be introduced? Can the new version of counting sort be used in the Radix sort?

3. If there is a max-heap and we want to search for the smallest element, what should be the searching strategy if we want to find it faster? You can assume that all the elements are distinct. Your algorithm may or may not improve the asymptotic performance, but it should be better than simple sequential search.

4. Illustrate the operation of MAX-HEAPIFY(A, 3) on the array A = 27, 17, 3, 16, 13, 10, 1, 5, 7, 12, 4, 8, 9, 0. Show your work step by step with necessary graph.

5. Illustrate the operation of (the entire) HEAPSORT on the array A = 5, 13, 2, 25, 7, 17, 20, 8, 4. Show work.

6. What are the **best case** and **worst case** for BUILD-MAX-HEAP(A) algorithm? What will the running time be in the best and worst case? Consider input array A of length n that is already sorted in increasing order and decreasing order.

7. What is the running time of QUICKSORT when all elements of array A have the same values? You can assume that we use the quicksort algorithm in Python presented in the lecture notes. Is this the best case, worst case or average? Explain.

8. Use the Python code of QUICKSORT on the lecture notes, illustrate the steps for the first `Partition` on the input 13, 19, 9, 5, 12, 8, 7, 4, 11, 2, 6, 21. Remember that the pivot will be moved to the middle before we finish the `Partition` function.

9. Assume that you have a linear time median function `linear_time_median(A)` already defined. It takes an array A as input and returns the value of the median. Make use of this function and define anther function that returns the 10^{th} smallest value (10^{th} order statistics) of a given array A. Your function can be either iterative or recursive.

   Your function should call the `linear_time_median(A)` whenever needed. You can assume there is a `sort(A)` function, but you cannot call `sort(A)` unless the size of A is no more than 5. For each time you call `linear_time_median(A)`, you can assume there is one value returned (the median of [3,1,2] will be 2, and the median of [3,1,0,2] will be 1 which is the lower of the middle two values). Provide either pseudo code or code in Python.