CSCI 570 - Fall 2019 - HW 3
Due September 18th

1 Graded Problems

1. Design a data structure that has the following properties (assume n elements in the data structure, and that the data structure properties need to be preserved at the end of each operation):

   - Find median takes $O(1)$ time
   - Extract-Median takes $O(\log n)$ time
   - Insert takes $O(\log n)$ time
   - Delete takes $O(\log n)$ time

   Do the following:

   (a) Describe how your data structure will work

   (b) Give algorithms that implement the Extract-Median() and Insert() functions.

   Hint: Read this only if you really need to. Your Data Structure should use a min-heap and a max-heap simultaneously where half of the elements are in the max-heap and the other half are in the min-heap.

2. There is a stream of integers that comes continuously to a small server. The job of the server is to keep track of k largest numbers that it has seen so far. The server has the following restrictions:

   (a) It can process only one number from the stream at a time, which means it takes a number from the stream, processes it, finishes with that number and takes the next number from the stream. It cannot take more than one number from the stream at a time due to memory restriction.

   (b) It has enough memory to store up to k integers in a simple data structure (e.g. an array), and some extra memory for computation (like comparison, etc.).

   (c) The time complexity for processing one number must be better than $\Theta(k)$. Anything that is $\Theta(k)$ or worse is not acceptable.

   Design an algorithm on the server to perform its job with the requirements listed above.

3. When we have two sorted lists of numbers in non-descending order, and we need to merge them into one sorted list, we can simply compare the first two elements of the lists, extract the smaller one and attach it to the end of the new list, and repeat until one of the two original lists become empty, then we attach the remaining numbers to the end of the new list and it's done. This takes linear time. Now, try to give an algorithm using $\mathcal{O}(n \log k)$ time to merge k sorted lists (you can also assume that they contain numbers in non-descending order) into one sorted list, where n is the total number of elements in all the input lists. (Hint: Use a min-heap for k-way merging.)

4. Suppose you are given two sets A and B, each containing n positive integers. You can choose to reorder each set however you like. After reordering, let a_i be the i-th element of set A, and let b_i be the i-th element of set B. You then receive a payoff on $\prod_{i=1}^{n} {a_i}^{b_i}$. Give an algorithm that will maximize your payoff. Prove that your algorithm maximizes the payoff,and state its running time.

2 Practice Problems

1. The police department in a city has made all streets one-way. The mayor contends that there is still a way to drive legally from any intersection in the city to any other intersection, but the opposition is not convinced. A computer program is needed to determine whether the mayor is right. However, the city elections are coming up soon, and there is just enough a time to run a linear-time algorithm.

   a Formulate this as a graph problem and design a linear-time algorithm. Explain why it can be solved in linear time.

   b Suppose it now turns out that the mayors original claim is false. She next makes the following claim to supporters gathered in the Town Hall: ”If you start driving from the Town Hall (located at an intersection), navigating one-way streets, then no matter where you reach, there is always a way to drive legally back to the Town Hall.” Formulate this claim as a graph problem, and show how it can also be varied in linear time

2. You are given a weighted directed graph $G = (V, E, w)$ and the shortest path distances $\delta(s, u)$ from a source vertex s to every other vertex in G. However, you are not given $\pi(u)$ (the predecessor pointers). With this information, give an algorithm to find a shortest path from s to a given vertex t in $O(|V|+|E|)$