CIS 675 Midterm 1

Instructions:

1. Your exam grade will be based on your performance in the oral exam. The solution you present must match your written solution.

2. You are permitted to use your textbook, lecture materials, and existing online sources.

3. You are permitted to collaborate with other currently enrolled CIS 675 students. However, you are *strongly discouraged* from doing so. If you do not create your own solutions, you will do poorly on the oral exam.

4. You are *not* permitted to post exam questions online for others to answer (e.g., on StackOverflow or Chegg).

5. If you are confused about what any of the problems is asking, reach out to the instructor or TA immediately. It is better to clarify now than during the oral exam!

Problem 1: Suppose you have a set S of boxes, where each box i has a height h_i and base surface area b_i (this surface area is simply the width times the depth). Assume that all of these values are distinct: i.e., no two boxes have the same height or the same base surface area. Your goal is to select the largest subset of boxes such that for each selected box j, there is no other box k in S for which $h_k > h_j$ **and** $b_k > b_j$ (in other words, there is no other box that is BOTH taller and has a larger surface area than j: it is ok if there is another box that is larger with respect to just one of these values). Design an efficient divide and conquer algorithm for finding this subset.

Problem 2: Suppose you have N computers arranged in a row. You are trying to connect them in a network by adding cables between machines. For security, you want to ensure that when any computer i sends data to any other computer j, the data does not pass through too many other computers. Assume that computers are organized in increasing order of security privileges. This means that data can be sent from computer i to j *if and only if* $i < j$. Ideally, you would connect every computer i to every other computer j with $i < j$, but that would require $O(N^2)$ connections, which is too costly. Which pairs of computers should you connect so that the following two conditions are met: (a) the shortest route from any computer i to any other computer j (with $i < j$ passes through at most one other computer, and (b) only $O(N \log N)$ direct connections are made.

Problem 3: Suppose you have an array A of size n. You want to sort the array in increasing order, but you don't have access to the normal 'write' operation (in other words, the process by which you sort this array is going to be very different from the sorting algorithms you're used to). You can only use a constant amount of space other than the memory already used by the array. The only operation that you can use to modify the array is $Reverse(k)$. This operation takes the first k elements of the array ($A[1]...A[k]$) and reverses them in the array. For example, if $A = [1, 4, 3, 2, 5]$ and you call $Reverse(3)$, you get $A = [3, 4, 1, 2, 5]$. $Reverse$ always flips the *first* k elements of the array. Design an algorithm to sort the array with $O(n)$ calls to $Reverse$ (you don't need to find the algorithm that makes the fewest calls- which would be a very difficult problem- as long as it make $O(n)$ calls, it's fine). Exactly how many calls to $Reverse$ does your algorithm require? Hint: try a few examples by hand.

Problem 4: Suppose you have a flight network N, showing flights between pairs of airports. Each flight has a given distance. You can assume that for each flight, its reverse (going in the opposite direction) also exists, and has the same distance. Recently, a large number of people have been flying from airport s to airport t. For this problem, you can assume that s and t are fixed. An airline is considering adding a new flight to the network, and wants to know how it will affect the length of the shortest path from s to t. Your job is to design an efficient algorithm to pre-process network N to store information about the shortest paths, and provide a data structure D with this stored information to the airline. For any given new flight (say, from x to y), the airline should be able use D to compute in *constant time* the new shortest path length between s and t once the flight from x to y is added. You do not know in advance which new flights the airline is considering.

For this problem, you should describe (a) what D is, and how it can be used to compute the shortest path length from s to t after a new flight is added, and (b) how to efficiently compute D.

Hint: In one valid solution, D consists of two arrays.

Problem 5: Suppose you are traveling along a directed network of trains, where the nodes represent train stations and the edges represent route segments between stations. The length of the route segment from station u to station v is denoted as $d_{u,v}$. You want to plan a trip from station A to station B. Your first priority is that the number of route segments should be minimized (because you hate transferring trains); but if there are multiple routes with the minimum number of segments, you want the one that is overall shortest in terms of total distance. Design an efficient algorithm to determine which route to take.

Problem 6: Suppose there is an undirected road network connecting many cities. Each road connects two cities. (Not all pairs of cities are directly connected by a single road.) You are given a list of all the roads. Some roads have bridges on them, and each bridge has a weight limit. You are given the location of each bridge, as well as its weight limit. You are driving a large truck from City A to City B. If the truck exceeds the weight limit of a bridge, you cannot drive it over that bridge. Design an efficient algorithm to identify the maximum weight truck that you can drive from City A to City B.