This is a closed book closed notes exam; Cell or iphones, tablets, laptops, calculators, apple watches, etc. must be turned off, and must be placed in your bag next to the wall.

Solve and turn in any 6 problems. Each question has 17 points (you receive 2 bonus points). If you turn in more than 6 problems, only the first 6 will be graded. You will be graded for accuracy, and clarity.

(1)- Solve $T(n) = 8T(n/2) + n^2, n = 2^k, T(1) = 1$. Show all your work.

(2)-Solve $T(n) = 2T(n/2) + n, n = 2^k, T(1) = 1$. Show all your work.

(3)- Assume that we have $n$ integers in the range of 1 to 10000. Outline an algorithm for sorting these numbers in $O(n)$ time. What is the name of this algorithm?

(4) Let $X$ be a sequences of $n$ integers. We want to compute the longest subsequence of $X$ which is monotonically increasing. Design an $O(n^2)$ time algorithm that solves this problem. You should define all proper terms first, and then derive a recurrence relation, and then solve it. (Please do not give an example.)

(5) Let $G = (V,E)$ be a directed graph with weight $w(i,j)$ assigned to any edge $ij \in E$. We want to solve all pairs shortest path problem. Design a dynamic programming algorithm (Floyd Warshall Algorithm) for solving this problem. You should define all proper terms first, and then derive a recurrence relation, and then derive the algorithm, and then write the pseudo-code.

(6) Assume that we can multiply two 2 by 2 matrices using 7 scalar multiplications. Write a recurrence relation for a divide and conquer algorithm that uses this fact, to multiply two $n$ by $n$ matrices, and solve this recurrence relation, to derive the time complexity of the algorithm

(7) Consider the vertex cover problem. Outline a 2 times the optimal approximation algorithm for solving this problem. What is the time complexity of this algorithm?

(8) Consider the problem for finding the maximum of $n$ numbers. Prove that any algorithm that solves this problem by only comparing, has to perform at least $n - 1$ comparisons in the worst case.

(9) Answer True or False.

1. Any comparison based algorithm for sorting requires $\Omega(n^3)$ comparisons.

2. Time complexity of Prim’s algorithm (using a priority queue) is $O(n \log n)$, where $n$ is the number of vertices and $m$ is number of edges.

3. We can solve traveling salesman problem, exactly, in $O(n^2)$ time, on an $n$ vertex graph.

4. The time complexity of dijkstra’s algorithm is $O(n \log n)$, where $n$ is the number of vertices the graph.

5. The time complexity of Kruskal’s algorithm Using union and Find algorithm is $O(n^2)$, where $n$ is the number of vertices.