# COM S 311 Exam 2

| Name | Recitation Time |
|---|---|
| | |

| Problem | Max Points | Score |
|---|---|---|
| 1 (Solving Recurrences) | 20 | |
| 2 (Graph Algorithms) | 20 | |
| 3 (Graph: Shortest Paths) | 20 | |
| 4 (Divide and Conquer) | 20 | |
| 5 (Greedy) | 20 | |
| 6 (Extra Credit: MST) | 25 | |
| Total | 100 | |

- **When asked to design an algorithm, write pseudo code, not code of any specific programming language and not library operations of any specific programming language library.**
- Proof of correctness ⇒ justifiable and unambiguous arguments. No need for formalism.
- Level of points for solution

```
if designed algorithm is c
   if proof of correctne
      if run-time anal
           points = 100%
      else
           points = 75%
   else if run-time analysis shows derivation steps and is correct then
           points = 75%
         else
           points = 50%
else if designed algorithm is correct and brute force then
   points = 30%
else if designed algorithm is incorrect then
   points = 0--20% (at the discretion of grader)
else if answer is "DO NOT GRADE" (YOU NEED TO EXPLICITLY WRITE DO NOT GRADE)
   points = 15%
else
   points = 0%
```

**You may use following algorithms, runtimes and their correctness proofs as blackbox**

1. Basic Data Structures: BST, Hash Tables, Priority Queue
2. BFS, DFS, Reversing a Graph, SCC, Topological Sorting, MST, Dijkstra's Shortest Path Algorithm that returns Distance Array as well as Shortest Path Tree.
3. Sorting Algorithms.

Your application/use of these algorithms must match input/output behaviors.

1. Solve the recurrences (assume $T(2)$ or $T(3)$ is constant). Do not use Master Theorem.

   (a) $T(n) = 3T(n/2) + n$

   (b) $T(n) = T(n/3) + T(2n/3) + c$

2. Given a directed graph $G = (V, E)$, we say that a vertex $v \in V$ is a bottom vertex if there exists a path from all vertices to $v$. Write an algorithm to find all bottom vertices in a directed graph. Justify the correctness of your algorithm. Derive the runtime of your algorithm.

3. Let $G = (V, E)$ be a weighted undirected graph and let $s \in V$ be a vertex. Let $d[u]$ denote the length of the shortest path from $s$ to $u$. We say that an edge $\langle x, y \rangle$ is *purple edge* if the following condition holds:

   $$d[y] = d[x] + c(\langle x, y \rangle),$$

   where $c(\langle x, y \rangle)$ is the cost of the edge $\langle x, y \rangle$.

   Give an algorithm that gets $G$ and $s$ as input and outputs all *purple edges*. State the run-time of your algorithm.

4. Given a sorted array $A$ of integers and an integer $k$, write an algorithm that outputs the number of times $k$ appears in $A$. Justify the correctness of your algorithm. Derive the runtime of your algorithm.

5. You own a car repair business, and each morning you have a set of customers whose cars need to be fixed. You can fix only one car at a time. Customer $i$'s car needs $t_i$ time to be fixed. Given a schedule (i.e., an ordering of the jobs), let $C_i$ denote the finishing time of fixing customer $i$'s car. For example, if job $j$ is the first to be done, we would have $C_j = t_j$; and if job $i$ is done right after job $j$, we would have $C_i = C_j + t_i$. Each customer $i$ also has a given weight $w_i$ that represents his or her importance to the business. The hap hing time of $i$'s job. So the company de e completion times, $\Sigma$

   The company is scheduling the jobs in decreasing order of $w /t$ . Prove that this strategy gives an optimal solution, i.e., minimizes $\Sigma_{i=1}^{n} w_i C_i$. Use an exchan ss.

6. *Extra Credit.* There are $n$ towns in a mountainous country and $m$ roadways. The objective of the local government officials is to ensure that some subset of r through the winter and to maintain accessibility to all towns.

   You can view the network of roadways as an undirected graph $G = (V, E)$, $|V| = n$ and $|E| = m$. Each edge $e$ in this graph is annotated with a number $a_e$, that gives the altitude of the highest point on the road. We'll assume that no two edges have exactly the same altitude value $a_e$. The height of a path $P$ in the graph is then the maximum of $a_e$ over all edges $e$ on $P$. Finally, a path between towns $i$ and $j$ is declared to be winter-optimal if it achieves the *minimum possible height* over all paths from $i$ to $j$. The local government wants to find a subset $E' \subseteq E$ of roadways such that the subgraph $(V, E')$ is connected and for every pair of towns $i$ and $j$, the height of the winter-optimal path in $(V, E')$ should be no greater than it is in the full graph $G = (V, E)$. We'll say that $(V, E')$ is a *minimum-altitude connected subgraph* if it has this property.

   Prove or disprove that a MST is a minimum-altitude connected subgraph.