Com S 311, Final Exam

Problem	Max Points	Score
1 (Basics)	20	
2 (Hash and Heaps)	20	
3 (Recurrence)	10	
4 (Graph)	15	
5 (Greedy)	15	
6 (Dynamic Programming)	20	
(Extra Credit-1: Graphs)	20	
(Extra Credit-2: NP)	20	
Total	100 + 40 (EC)	

- If you write "*Do not grade*" and nothing else, you will receive 15% credit.
- For all problems that involve writing an algorithm, please write clear pseudocode, not Java or C.
- In general you may use the following algorithms and their time bounds as a "black box" on the exam (that is, unless modifications are needed, you do not need to write the code nor derive runtime for them): sorting, breadth-first search, depth-first search, Prim's algorithm, Kruskal's algorithm, Dijkstra's algorithm, topological sort. You can assume that hashtable operations for integers are O(1)

  However, if you modify any of these methods, then you must write a complete description of the modified method. Merely stating the modification does not suffice.
- Level of points for solutions related to design and analysis algorithm

```
if designed algorithm is correct and efficient as expected then
   if proof of correctness is unambiguous and justifiable then
      if run-time analysis shows derivation steps and is correct then
           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%
```

1. Short Answer questions. Do not write explanations. There is *Do not grade* option for this question.

   (a) True or False: $2^{3n} \in O(2^{2n})$.

   (b) True or False: $2^{3^n} \in O(2^{2^n})$.

   (c) True or False: Every divide and conquer algorithm runs in $O(n \log n)$ time.

   (d) True or False: If G has a topological ordering, then G must be a DAG.

   (e) True or False: Every undirected graph with more than n edges has a cycle.

   (f) True of False: The runtime for solving any decision problem in NP is not polynomial with respect to the size of the problem..

   (g) What is the solution to the recurrence $T(n) = T(n/2)$, $T(1) \in O(1)$.

   (h) What is the runtime for remove the largest element from a min-heap?

   (i) Assuming that the computing hash function takes $O(1)$ time, what is the time taken to output the smallest element of a hash table consisting of n elements?

   (j) What is the runtime of Prim's Algorithm?

2. *Hashing and Heap*

   (a) Let A_1, A_2 be two arrays each consisting of n integers and T be an integer. Give an algorithm using hashing that checks whether there exist integers $x \in A_1$ and $y \in A_2$ such that $x + y$ equals T. State the run-time of your algorithm.

   (b) Given an array A of n integers, and an integer k, give an algorithm that outputs the kth smallest element. Express runtime as a function of k and n.

3. Solve the recurrence relation $T(n) = 4T(n/3) + n$ [assume $T(1) \in O(1)$].

4. You are given a directed graph G with n vertices. You need to verify the claim that the graph has a vertex v with the following properties.

   - Number of vertices from which there is a path to v (other than v itself) is k_1.
   - Number of vertices that have a path from v (other than v itself) is $n - k_1 - 1$.
   - There is no vertex $u \neq v$ such that there is a path from u to v and there is a path from v to u.

   Write the verification algorithm.

5. You are given n jobs numbered $1, 2, \cdots, n$ to complete and each job i comes with a difficulty d_i. Each job takes exactly one week to complete irrespective of its difficulty. If you complete job i during week $j (\leq n)$, then you earn a profit of $d_i(n - j)$. You plan to complete all the jobs in n weeks, since the goal is to maximize the profit. Consider greedy algorithm that completes that jobs in the based on the *decreasing order* of difficulty. Use exchange argument to prove that the greedy algorithm produces optimal solution. You may assume that all d_i's are distinct.

6. Let M be a matrix of integers with n rows (rows numbered 1 to n) and m columns (columns numbered 1 to m). Let $M[i, j]$ denote the entry in ith row and jth column. A *horizontal cut* in M is a sequence $[c_1, c_2, \cdots, c_m]$ such that

   - For every i, $1 \leq c_i \leq n$
   - For $1 \leq i \leq m - 1$, $c_{i+1} \in \{c_i - 1, c_i, c_i + 1\}$

   Given a horizontal cut $[c_1, c_2, \cdots, c_m]$, its cost is $M[c_1, 1] + M[c_2, 2] + \cdots + M[c_m, m]$. A horizontal cut is a *max-cost cut* if its cost is at least the cost of any other horizontal cut.

   Give a dynamic programming algorithm, that takes a matrix M as input, and outputs the *cost of the* max-cost cut. The following needs to be presented as part of your answer.

(a) Recurrence relation (recursive characterization or recursive definition) describing the value of max-cost cut.

(b) Iterative algorithm for computing the value of max-cost cut.

(c) Runtime of the iterative algorithm.

* **Extra Credit-1.** Let G be an undirected graph where every edges has exactly the same weight and let s be a vertex of the graph. Write an algorithm that takes s as input and outputs for every vertex v, the number of shortest path from s to v.

* **Extra Credit-2.** Let $L = \{S_1, S_2, \cdots, S_n\}$ be a collection of sets. A set H is a *blanket* for L if $\forall i \in [1, n] : H \cap S_i \neq \emptyset$. Show that the following decision problem is in NP:

  Input: $L = \{S_1, S_2, \cdots S_n\}$
  Decision: Does L have a blanket of size $\leq \log n$.