CSE 21 Spring 2016
Homework 5
Due: Friday, August 19 at 11:59pm

INSTRUCTIONS

Homework should be done in groups of **one to two** people. You are free to change group members at any time throughout the quarter. Problems should be solved together, not divided up between partners. A **single representative** of your group should submit your work through Gradescope. Submissions must be received by 11:59pm on the due date, and there are no exceptions to this rule.

Homework solutions should be neatly written or typed and turned in through **Gradescope** by 11:59pm on the due date. No late homeworks will be accepted for any reason. You will be able to look at your scanned work before submitting it. Please ensure that your submission is legible (neatly written and not too faint) or your homework may not be graded.

Students should consult their textbook, class notes, lecture slides, instructors, TAs, and tutors when they need help with homework. Students should not look for answers to homework problems in other texts or sources, including the internet. Only post about graded homework questions on Piazza if you suspect a typo in the assignment, or if you don't understand what the question is asking you to do. Other questions are best addressed in office hours.

Your assignments in this class will be evaluated not only on the correctness of your answers, but on your ability to present your ideas clearly and logically. You should always explain how you arrived at your conclusions, using mathematically sound reasoning. Whether you use formal proof techniques or write a more informal argument for why something is true, your answers should always be well-supported. Your goal should be to convince the reader that your results and methods are sound.

For questions that require pseudocode, you can follow the same format as the textbook, or you can write pseudocode in your own style, as long as you specify what your notation means. For example, are you using "=" to mean assignment or to check equality? You are welcome to use any algorithm from class as a subroutine in your pseudocode. For example, if you want to sort list A using InsertionSort, you can call InsertionSort(A) instead of writing out the pseudocode for InsertionSort.

REQUIRED READING Rosen 10.1, 10.2, 10.3, 10.4 through Theorem 1, 10.5 through Example 7.

KEY CONCEPTS Graphs (definitions, modeling problems using graphs), Hamiltonian tours, Eulerian tours, Fleury's algorithm, DAGs.

---

1. Let G be a DAG with vertices labeled $1, 2, \ldots, n$. Define the adjacency matrix and adjacency list representations of G as in Rosen, page 669.

   (a) (3 points) Let A be the adjacency matrix that represents G. Write pseudocode that takes A as an input and outputs the array `InDegree[]` for the graph G.

   (b) (2 points) Let A be the adjacency matrix that represents G. Write a high-level English description of an algorithm that takes as inputs A and a source vertex v and outputs the adjacency matrix that represents $G - v$.

   (c) (3 points) Let L be the adjacency list that represents G. Write pseudocode that takes L as an input and outputs the array `InDegree[]` for the graph G.

   (d) (2 points) Let L be the adjacency list that represents G. Write a high-level English description of an algorithm that takes as inputs L and a source vertex v and outputs the adjacency list that represents $G - v$.

2. Say we have a directed graph where each vertex v_i represents a function $f_i(n)$ and there is an edge from v_i to v_j if and only if $f_i \in O(f_j)$ (and $i \neq j$, i.e, we don't put in self-loops). Give a necessary and sufficient condition for this graph to be a DAG. Prove your answer correct. (5 points correct condition, 5 points proof).

3. A daily flight schedule is a list of all the flights taking place that day. In a daily flight schedule, each flight F_i has an origin airport OA_i, a destination airport DA_i, a departure time d_i and an arrival time $a_i > d_i$. This is an example of a daily flight schedule for May 10, 2016, listing flights as $F_i = (OA_i, DA_i, d_i, a_i)$:

$$
\begin{aligned}
F_1 &= (\text{SAN, LAX, 7:00am, 8:00am}) \\
F_2 &= (\text{LAX, LV, 8:30am, 9:30am}) \\
F_3 &= (\text{LV, LAX, 7pm, 8pm }) \\
F_4 &= (\text{SAN, LV, 9:00am, 10:30am}) \\
F_5 &= (\text{LV, SAN, 4:00pm, 6:00pm}) \\
F_6 &= (\text{LV, LAX, 8:00pm, 9:30pm}) \\
F_7 &= (\text{LAX, SAN, 9pm, 10pm})
\end{aligned}
$$

   (a) (4 points) Describe how to construct a DAG so that paths in the DAG represent possible sequences of connecting flights a person could take. What are the vertices, and when are two vertices connected with an edge?

   (b) (2 points) Why is your graph always a DAG?

   (c) (2 points) Draw the DAG you described for the given example of May 10, 2016.

   (d) (2 points) I start at 6AM in San Diego (where I live) on May 10, 2016, and have a yearning to spend time in LV. What is the maximum amount of time I could spend in LV and still get home tonight?

4. In this scheduling problem, you are given a set of tasks (shown in bold) that all need to be performed to complete a large-scale software project. However, certain tasks cannot be started until other tasks have been completed. These constraints are:

   - **Beta Testing** cannot begin until **Alpha Testing** and **Setting up Test Sites** are done.
   - **Project Release** cannot begin until **Beta Testing** is done.
   - **Alpha Testing** cannot begin until **Writing Documentation** and **Integrating Modules** are done.

- **Writing Documentation**, **Developing Module A**, **Developing Module B**, and **Developing Module C** cannot begin until **Developing System Requirements** is done.
- **Writing Functional Requirements** cannot begin until **Determining User Needs** is done.
- **Integrating Modules** cannot begin until **Developing Module A**, **Developing Module B**, and **Developing Module C** are done.
- **Developing System Requirements** and **Setting up Test Sites** cannot begin until **Writing Functional Requirements** is done.

(a) (5 points) Draw a graph that models this situation. Is your graph directed or undirected, and why? Does it contain cycles, and why?

(b) (5 points) Use the graph to help you list all the tasks in an order in which they can actually be performed, given the above set of constraints. Explain how you found your answer.

5. A binomial tree is a special kind of rooted tree used for various data structures in computer science. A degree d binomial tree can be defined recursively as follows. A degree 0 binomial tree is a single vertex with no edges. A degree d binomial tree has a root vertex with out-degree d. The first (that is, leftmost) subtree is a degree $d-1$ binomial tree. The second (that is, second to left) subtree is a degree $d-2$ binomial tree. Continue on in this way so that the last (rightmost) subtree is a degree 0 binomial tree.

   (a) (2 points) Draw binomial trees of degree 1, 2, 3, and 4.

   (b) (3 points) What is the height $h(d)$ of a degree d binomial tree? Prove your result by induction on d.

   (c) (2 points) Write a recurrence for the number of vertices $n(d)$ in a binomial tree of degree d.

   (d) (3 points) Find a closed form solution to the recurrence.