CMPSCI 311: Introduction to Algorithms **Spring 2017**

Homework 2

Your Name: ____________________

Collaborators and sources: ____

You make work in groups, but you must write solutions yourself. List collaborators on your submission.

If you are asked to design an algorithm, please provide: (a) the pseudocode for the algorithm, (b) an explanation of the intuition for the algorithm, (c) a proof of correctness, (d) the running time of your algorithm and (e) justification for your running time analysis.

Submission instructions. This assignment is due by 8:00pm on Friday, Feb 24 in Gradescope (as a pdf file). Please review the course policy about Gradescope submissions on the course website.

1. **(10 points) Graph Properties**

   (a) **(5 points) K&T Ch. 3 Ex. 7** Claim: Let G be an undirected graph on n nodes, where n is an even number. If every node of G has degree at least $n/2$, then G is connected. Decide whether you think the claim is true or false, and either give a proof of the claim or give a counterexample.

   (b) **(5 points) DAGs.** True or false: if G is a directed graph that has a node with no incoming edges, then G is a DAG. Either prove this statement is true, or give a counterexample to show it is false.

2. **(10 points) Directed Graphs.** Given a directed acyclic graph G, give an $O(m+n)$ time algorithm to determine if the graph has a directed path that visits every vertex.

3. **(15 points) K&T Ch 3, Ex 3**. The algorihtm described in Section 3.6 for computing a toplogical ordering of a DAG repeatedly finds a node with no incoming edges and deletes it. This will eventually produce a topological ordering, provided that the input graph really is a DAG.

   But suppose that were given an arbitrary graph that may or may not be a DAG. Extend the topological ordering algorithm so that, given an input directed graph G, it outputs one of two things: (a) a topological ordering, thus establishing that G is a DAG; or (b) a cycle in G, thus establishing that G is not a DAG. The running time of your algorithm should be $O(m+n)$ for a directed graph with n nodes and m edges.

4. **(20 points) K&T Ch 3 Ex 12** You're helping a group of ethnographers analyze some oral history data they've collected by interviewing members of a village to learn about the lives of people who've lived there over the past two hundred years. From these interviews, they've learned about a set of n people (all of them now deceased), whom we'll denote $P_1, P_2, \ldots, P_n$. They've also collected facts about when these people lived relative to one another. Each fact has one of the following two forms:

   - For some i and j, person P_i died before person P_j was born; or
   - For some i and j, the life spans of P_i and P_j overlapped at least partially.

   Naturally, they're not sure that all these facts are correct; memories are not so good, and a lot of this was passed down by word of mouth. So what they'd like you to determine is whether the data they've collected is at least internally consistent, in the sense that there could have existed a set of people for which all the facts they've learned simultaneously hold.

   Give an efficient algorithm to do this: either it should produce proposed dates of birth and death for each of the n people so that all the facts hold true, or it should report (correctly) that no such dates can exist — that is, the facts collected by the ethnographers are not internally consistent.

5. **(25 points) K&T Ch4.Ex3.** You are consulting for a trucking company that does a large amount of business shipping packages between New York and Boston. The volume is high enough that they have to send a number of trucks each day between the two locations. Trucks have a fixed limit W on the maximum amount of weight they are allowed to carry. Boxes arrive at the New York station one by one, and each package i has a weight w_i. The trucking station is quite small, so at most one truck can be at the station at any time. Company policy requires that boxes are shipped in the order they arrive; otherwise, a customer might get upset upon seeing a box that arrived after his make it to Boston faster. At the moment, the company is using a simple greedy algorithm for packing: they pack boxes in the order they arrive, and whenever the next box does not fit, they send the truck on its way.

   But they wonder if they might be using too many trucks, and they want your opinion on whether the situation can be improved. Here is how they are thinking. Maybe one could decrease the number of trucks needed by sometimes sending off a truck that was less full, and in this way allow the next few trucks to be better packed.

   Prove that, for a given set of boxes with specified weights, the greedy algorithm currently in use actually minimizes the number of trucks that are needed. Your proof should follow the type of analysis used in the book for the Interval Scheduling Problem: it should establish the optimality of this greedy packing algorithm by identifying a measure under which it stays ahead of all other solutions.

6. **(20 points) Greedy Algorithms**

   For the upcoming bake sale, you are planning to bake n different recipes, labeled $r_1, r_2, \ldots r_n$. Recipe r_i requires p_i minutes of *preparation* time and b_i minutes of *baking* time. Fortunately, you have access to a test kitchen with n ovens, so all of recipes can bake simultaneously once they are prepared. However, there is only one of you, so you need to decide in which order to complete the preparation of each recipe.

   For example: as soon as you complete preparing the first recipe, you can put it in the oven to bake and immediately begin preparing the second recipe. When you complete the second recipe, you can put it in the oven to bake whether or not the first recipe is done baking; and so on.

   Lets say that a schedule is an ordering for preparation of the recipes, and the completion time of the schedule is the earliest time at which all recipes are done baking. This is an important quantity to minimize, because the bake sale is coming up soon!

   Give a polynomial-time algorithm that finds a schedule with as small a completion time as possible.

7. **(0 points).** How long did it take you to complete this assignment?