CS 373U: Combinatorial Algorithms, Spring 2004
Homework 5

Due Wednesday, April 28, 2004 at noon

Name:	
Net ID:	Alias:
Name:	
Net ID:	Alias:
Name:	
Net ID:	Alias:

- For each numbered problem, if you use more than one page, staple all those pages together. **Please do *not* staple your entire homework together.** This will allow us to more easily distribute the problems to the graders. Remember to print the name and NetID of every member of your group, as well as the assignment and problem numbers, on every page you submit. You do not need to turn in this cover page.
- As with previous homeworks, we strongly encourage you to begin early.
- This will be the last graded homework.

#	1	2	3	4	5	Total
Score						
Grader						

CS 373U **Homework 5 (due April 28, 2004)** **Spring 2004**

1. (a) Prove that every graph with the same number of vertices and edges has a cycle.

   (b) Prove that every graph with exactly two fewer edges than vertices is disconnected.

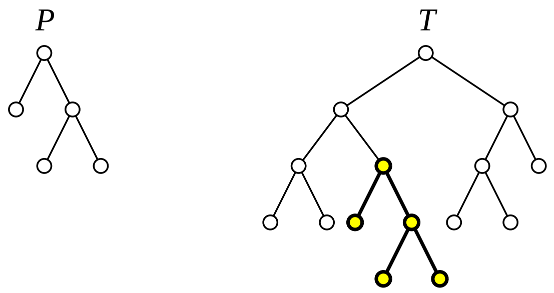
   Both proofs should be entirely self-contained. In particular, they should not use the word "tree" or any properties of trees that you saw in CS 225 or CS 273.

2. A *palindrome* is a string of characters that is exactly the same as its reversal, like `X`, `FOOF`, `RADAR`, or `AMANAPLANACATACANALPANAMA`.

   (a) Describe and analyze an algorithm to compute the longest *prefix* of a given string that is a palindrome. For example, the longest palindrome prefix of `RADARDETECTAR` is `RADAR`, and the longest palindrome prefix of `ALGORITHMSHOMEWORK` is the single letter `A`.

   (b) Describe and analyze an algorithm to compute a longest *subsequence* of a given string that is a palindrome. For example, the longest palindrome subsequnce of `RADARDETECTAR` is `RAETEAR` (or `RADADAR` or `RADRDAR` or `RATETAR` or `RATCTAR`), and the longest palindrome subsequence of `ALGORITHMSHOMEWORK` is `OMOMO` (or `RMHMR` or `OHSHO` or...).

3. Describe and analyze an algorithm that decides, given two binary trees P and T, whether T is a subtree of P. There is no actual *data* stored in the nodes—these are not binary *search* trees or binary *heaps*. You are only trying to match the *shape* of the trees.



P appears exactly once as a subtree of T.

4. Describe and analyze an algorithm that computes the *second* smallest spanning tree of a given connected, undirected, edge-weighted graph.

5. Show that if the input graph is allowed to have negative edges (but no negative cycles), Dijkstra's algorithm[1] runs in exponential time in the worst case. Specifically, describe how to construct, for every integer n, a weighted directed graph G_n without negative cycles that forces Dijkstra's algorithm to perform $\Omega(2^n)$ relaxation steps. Give your description in the form of an algorithm! *[Hint: Towers of Hanoi.]*

[1]This refers to the version of Dijkstra's algorithm described in Jeff's lecture notes. The version in CLRS is always fast, but sometimes gives incorrect results for graphs with negative edges.