University of New Mexico
Department of Computer Science

Final Examination

CS 362 Data Structures and Algorithms
Spring, 2005

Name:
Email:

- Print your name and email, *neatly* in the space provided above; print your name at the upper right corner of *every* page. Please print legibly.
- This is an *closed book* exam. You are permitted to use *only* two pages of "cheat sheets" that you have brought to the exam and a calculator. *Nothing else is permitted.*
- Do all the problems in this booklet. *Show your work!* You will not get partial credit if we cannot figure out how you arrived at your answer.
- Write your answers in the space provided for the corresponding problem. Let us know if you need more paper.
- Don't spend too much time on any single problem. The questions are weighted equally. If you get stuck, move on to something else and come back later.
- If any question is unclear, ask us for clarification.

Question	Points	Score	Grader
1	20		
2	20		
3	20		
4	20		
5	20		
Total	100		

1. **True or False**

True or False: (circle one, 2 points each)

(a) **True or False**: There is a greedy algorithm for $0-1$ knapsack which runs in $O(n \log n)$ time.

(b) **True or False**: If there is an activity with start time earlier than all other activities, then the greedy algorithm for activity selection will always choose this activity.

(c) **True or False**: Let T be the MST for some graph G. Then for any pair of vertices x and y, the shortest path from x to y in G is the same as the path from x to y in T.

(d) **True or False**: Consider a graph $G = (V, E)$, with negative weight edges. We can solve the single source shortest paths problem on G in $O(|V||E|)$ time.

(e) **True or False**: Consider a graph $G = (V, E)$, with negative weight edges. The fastest time to determine if there is a negative cycle in G is $O(|V||E|)$.

(f) **True or False**: For a connected graph G, the BFS, DFS, MST and shortest path trees will all have the same number of edges.

(g) **True or False**: Consider a graph $G = (V, E)$, with negative weight edges. We can solve all pairs shortest paths on G in $O(|V|^3)$ time.

(h) **True or False**: If there is a polynomial time algorithm for some problem in NP, then all problems in NP can be solved in polynomial time

(i) **True or False**: We know of a problem in the class NP that is not in the class P.

(j) **True or False**: All problems are either in the class P or in the class NP.

2. **Short Answer (5 points each) Where appropriate, circle your final answer.**

(a) Solve the following recurrence using the master method: $T(n) = 2T(n/2) + 1$

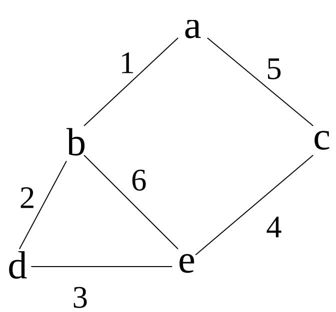
(b) Solve the following recurrence using annihilators: $T(n) = 4T(n-1) - 4T(n-2)$. Give the solution in general form i.e. do not solve for the constants

(c) Assume a data structure supports an operation *foo* such that a sequence of n calls to *foo* takes $O(n^2)$ time in the worst case. Answer the following: 1) What is the amortized cost of a *foo* operation and 2) What is the highest possible cost of a single call to *foo* and 3) What is the lowest possible cost of a single call to *foo*? Give your answers in Θ notation.

(d) Prove that $10n + 10 = O(n^2)$.

3. **Spanning Trees**

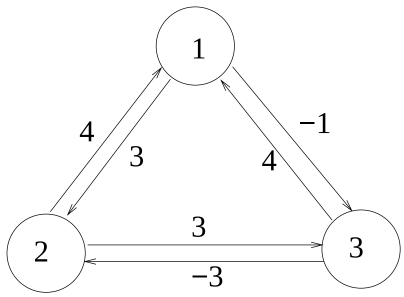
(a) Give the minimum spanning tree and the shortest path tree rooted at the vertex a for the following graph. Make sure that you label which tree is the MST and which is the shortest path tree rooted at a.



(b) Assume you are given an undirected, connected graph G with n nodes and at least n edges. Give an algorithm to return a cycle in G. (Your algorithm should return a minimal set of edges in G that form a cycle.)

4. **Graph Theory**

Recall that in the Floyd-Warshall algorithm $dist(u, v, r)$ is defined to be the shortest path from u to v where all intermediate vertices (if any) are numbered r or less. For the following graph, fill in the distance arrays computed by Floyd-Warshall for all values of r. *In the distance arrays, let the row be the vertex the path starts at and let the column be the vertex the path ends at.*



$r = 0$

	1	2	3
1			
2			
3			

$r = 1$

	1	2	3
1			
2			
3			

$r = 2$

	1	2	3
1			
2			
3			

$r = 3$

	1	2	3
1			
2			
3			

5. **NP-Hardness and Approximation Algorithms**

Your first job out of college is with the rapidly growing consulting company Downsize.com which specializes in "downsizing solutions". When Downsize.com consults for an organization, it is given a list of all employees of that organization and a list of all pairs of these employees that have personality conflicts. Downsize.com then seeks to find the *smallest* set of employees that must be fired in order to remove all personality conflicts in the organization. This problem is called the *Downsize* problem.

On your first day at work, your boss gives you the assignment of designing an efficient algorithm to solve the Downsize problem. In particular, he wants your algorithm to be given as input a list of employees and a list of personality conflicts and to output the minimum number of employees that must be fired to remove all personality conflicts.

(a) Show that you can not be expected to design an efficient algorithm to solve the Downsize problem. In particular, show that the problem is NP-Hard by doing a reduction from one of the following NP-Hard problems: 3-SAT, Clique, Vertex Cover, or TSP (the especially brave student can also try a reduction from Tetris :)

(b) Your boss accepts the fact that the Downsize problem is NP-Hard but is still unwilling to give you the day off. He now wants you to design an approximation algorithm for the Downsize problem. Can you do this? Explain.