CS 170 Final Exam 16 May 2007

NAME (1 pt): ______________________________

TA (1 pt): ______________________________

Name of Neighbor to your left (1 pt): ______________________________

Name of Neighbor to your right (1 pt): ______________________________

Instructions: This is a closed book, closed calculator, closed computer, closed network, open brain exam, but you are permited a 2 page, double-sided set of notes (4 pages total), large enough to read without a magnifying glass.

You get one point each for filling in the 4 lines at the top of this page. Each other question is worth 20 points.

Write all your answers on this exam. If you need scratch paper, ask for it, write your name on each sheet, and attach it when you turn it in (we have a stapler).

1	
2	
3	
4	
5	
6	
7	
Total	

1

Question 1 (20 points). Greedy Algorithms.

For each of the optimization problems described below, determine whether or not a greedy algorithm solves it exactly. If one does, briefly describe the algorithm and explicitly state what the greedy choice is. If a greedy algorithm does not solve it, give a candidate greedy choice (a reasonable greedy choice that sometimes fails to find the best solution) and briefly describe an example problem instance for which your greedy choice does fail to find the best solution.

Part 1 (7 points). A thief robs a local jewelry store that specializes in large valuable diamonds. Although he's greedy and would like to steal all the diamonds, his loot bag has a finite volume L. The ith diamond in the store has profit p_i and volume v_i. Which diamonds should he select to maximize his profit? Assume all selected diamonds must be placed in the loot bag without overflow and that the thief has no means to cut a diamond.

Part 2 (6 points). A cashier at a local coffee shop wishes to make change using 1 cent, 5 cent, 10 cent, and 25 cent denominations. That is, for any value C cents, the cashier wishes to select a set of coins (from pennies, nickels, dimes, and quarters) whose values sum to C cents. How should the cashier select coins to minimize the number of coins in the set? Assume there is an infinite supply of pennies, nickels, dimes, and quarters.

Part 3 (7 points). After such a big heist, the thief's hunger sends him to the coffee shop which is known for great pastries. He finds a large paper grocery bag with volume G and wishes to fill it with pastries. The ith pastry in the store has c_i calories and volume v_i. If he can break the pastries to fit into the bag (i.e. any fraction of a pastry may be chosen), how should the thief make his selection of pastries to maximize the number of calories taken (he needs energy for his next robbery)? Assume all selected pastries must be placed in the grocery bag without overflow.

Question 2 (20 points). Bodybuilding with Linear Programming.

Mr. R is a bodybuilder, who would like to increase his muscle mass by carefully choosing how he eats and trains. In his diet, we distinguish proteins (e.g. steak) and non-proteins (e.g. rice), both measured in calories. He can train with weightlifting and/or cardio (e.g. running or biking). Throughout, we refer to a single day of training.

Each minute of weightlifting builds 1 gram of muscle, burns 5 calories of protein and 2 calories of non-protein. Each minute of cardio burns 0 calories of protein and 10 calories of non-protein. His basal metabolism (activity of the heart, brain, etc.) already burns 800 calories of protein and 1200 of non-protein, daily, whether he trains or not, and these calories cannot be used for weightlifting or cardio.

For digestion to work, no more than 40% of the calories he eats can be protein. Since he does not want to lose weight, he must eat at least the number of calories of each type that he burns, but he can eat more. If he eats more than he burns, any excess calorie (protein or non-protein) becomes fat, at the rate of 1 gram of fat per 10 calories.

He wants to maximize his muscle gain, but does not want to gain more than 50 grams of fat, and cannot train for more than 2 hours (per day, weightlifting and cardio taken together). What should he do?

Part 1 (15 points). Formulate this problem as a linear program. It does not have to be in standard form (i.e. with all variables non-negative), and you do not need to solve it. For partial credit, and to help you clarify, you are encouraged to give a brief title to each variable describing its intended meaning (e.g. "p: number of protein calories eaten").

Part 2 (5 points). Show that this linear program is feasible.

Question 3 (20 points). Baby cubes by divide and conquer.

Baby Selma is trying to put back all her cubes into their box. They are all of different sizes, and for each cube there is a hole in the box that has exactly the right size, so to fit them all she must match each cube to its correct hole.

She can compare a cube and a hole by trying to put the cube in the hole. When she does, all she can tell is whether the cube is too big, too small, or just right. However she cannot tell whether a cube is bigger than an other by comparing them directly, nor can she compare two holes. Let C and H be the set of cubes and holes, and let $n = |C| = |H|$ denote the number of cubes and of holes.

Part 1 (1 point). Give a simple algorithm to put all the cubes in the box in $O(n^2)$. You do not need to prove the running time.

Part 2 (7 points). That's not good enough for Selma, and we're going to help her design a faster algorithm. Referring to quicksort or median-finding as inspiration, give a randomized method that, given an $\epsilon \in (0, 1)$, splits the holes into two groups H_L and H_U and a single hole h, such that all members of H_L are smaller than h and all members of H_U are bigger than h, and with the property that

$$\frac{1}{2} - \frac{\epsilon}{2} \leq \frac{|H_L|}{n} \leq \frac{1}{2} + \frac{\epsilon}{2}.$$

By computing the expected number of iterations of your algorithm, show that the average time needed to perform this split is linear in n.

(Hint: you may use the fact from algebra that $\sum_{i=1}^{\infty} i(1-x)^{i-1} = 1/x^2$. You may also assume that ϵn is an integer.)

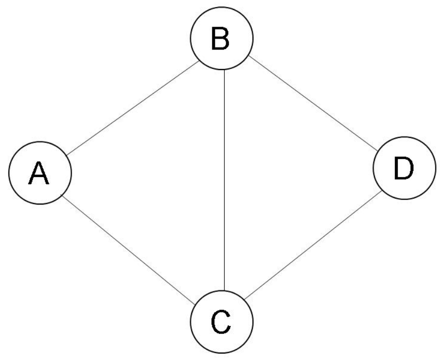
Part 3 (2 points). Show that your algorithm can be easily extended to also split the cubes into C_L, c, and C_U, where the members of C_L correspond to those of H_L, c fits in h, and the members of C_U correspond to those of H_U.

Part 4 (10 points). Use parts 2 and 3 to design a divide-and-conquer algorithm to put all the cubes in the box. Derive a recurrence equation for the average running time $T(n)$, and solve it. Your answer will depend on n and ϵ.

Question 4 (20 points). Reasoning about Graphs.

The first 2 parts of this question are about the following *undirected* graphs: Let $G = (V, E)$ be an undirected graph, with no self-loops (edges of the form (v, v)), and at most one edge between any two vertices. Let $G'(V', E')$ be constructed from G as follows: For every edge $e \in E$, there is a vertex $v_e \in V'$. There is an edge $(v_{e1}, v_{e2}) \in E'$ if and only if $e1$ and $e2$ share an end point in V.

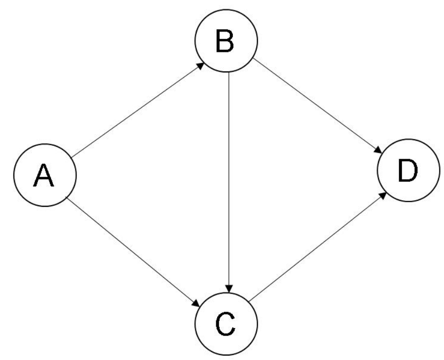
Part 1 (1 point). Given the graph G below, draw G'.



Part 2 (9 points). Is the statement "G' is acyclic if and only if G is acyclic" true or false? Either way, provide a proof (eg a counterexample, if it is false.)

The last 2 parts of this question are about the following *directed* graphs: Let $G = (V, E)$ be a directed graph, with no self-loops, and at most one edge pointing from one vertex to another. Let $G'(V', E')$ be constructed from G as follows: For every edge $e \in E$, there is a vertex $v_e \in V'$. There is a directed edge in E' from v_{e1} to v_{e2} if and only if $e1 = (u, v)$ and $e2 = (v, w)$, i.e. if there is a path from u to v to w in G.

Part 3 (1 point). Given the graph G below, draw G'.



Part 4 (9 points). Is the statement "G' is acyclic if and only if G is acyclic" true or false? Either way, provide a proof (eg a counterexample, if it is false.)

Question 5 (20 points). Scheduling to maximize profit.

You have one machine in your factory, and n jobs you could do, numbered from 1 to n, where job i takes time t_i on the machine. The machine can only do one job at a time, and must finish a job once it starts it. The machine starts processing jobs at time 0.

For example, suppose there are two jobs. Then either job 1 runs from time 0 to t_1 and job 2 runs from t_1 to $t_1 + t_2$, or else job 2 runs from time 0 to t_2 and job 1 runs from t_2 to $t_1 + t_2$.

Each job has a deadline $d_i > 0$. If job i finishes by its deadline d_i, then you make profit p_i, otherwise profit q_i, where $0 \leq q_i < p_i$.

What jobs should be scheduled in what order to maximize profit? Use dynamic programming to solve this problem faster than simply trying all possible job orderings. What is the running time of your algorithm?

Question 6 (20 points). NP-Completeness.

Given an undirected graph $G(V, E)$ with no self-loops (edges of the form (u, u)), we call G *k-colorable* if we can assign an integer (or color) $c_v \in \{1, 2, ..., k\}$ to each vertex $v \in V$, so that no two connected vertices have the same color.

For example, suppose G has 3 vertices connected in a triangle. Then G is not 2-colorable, but is k-colorable for $k \geq 3$.

We call the problem of deciding whether a graph G is k-colorable or not "k-COLORABILITY".

Part 1 (1 points). Show that k-COLORABILITY is in NP.

Part 2 (3 points). Give a polynomial-time algorithm for deciding if G is 2-colorable, and coloring it.

Part 3 (15 points). Show that if k-COLORABILITY is NP-complete for some value of k, then so is $(k+1)$-COLORABILITY. Be explicit about which problem is reduced to which other problem, and why the reduction is polynomial-time. Hint: Given a graph G, construct another graph G' with the property that G is k-colorable if and only if G' is $(k+1)$-colorable.

Part 4 (1 point). It is known that 3-COLORABILITY is NP-complete. (You do not have to prove this.) Prove that k-COLORABILITY is NP-complete for all $k \geq 3$.

Question 7 (20 points). True/False questions.

Circle the correct answer. No explanation required. Each correct answer is worth 1 point, but 1 point will be *subtracted* for each wrong answer, so answer only if you are reasonably certain.

T or F: There is a solution to the equations $z \equiv 3 \pmod 5$, $z \equiv 2 \pmod 7$, $z \equiv 4 \pmod{11}$, and $z \equiv 22 \pmod{35}$.

T or F: If $a|d$, $b|d$ and $d \geq lcm(a,b)$, then $lcm(a,b)|d$.

T or F: If G has all strongly connected components of size 1, one source, and one sink, then it has a unique topologically sorted order.

T or F: $\log^*\left((n!)^{(n!)}\right) = O(\log^*(\log\log n))$.

T or F: If $T(n) = 9 \cdot T(n/2) + n^3$ and $T(1) = 19$, then $T(n) = O(n^3)$.

T or F: If $T(n) = 7 \cdot T(n/2) + n^3$ and $T(1) = 17$, then $T(n) = O(n^3)$.

T or F: It is possible to implement the FFT in $O(n \cdot \log n)$ time if the prime factors of n are from $\{2, 3, 7\}$.

T or F: It is possible to take a linear program in the form $\max c^T \cdot x$, $A \cdot x \leq b$, $x \geq 0$, and rewrite it (perhaps by introducing new variables) to eliminate all inequalities.

T or F: If Dijkstra's algorithm is run on a graph with negative edge weights, then it will still compute an upper bound on the shortest path from the source vertex to all other vertices.

T or F: If a directed graph has 2 strongly connected components that are sources, and 2 strongly connected components that are sinks, then it is possible to make the graph have just 1 connected component by adding at most another 3 edges.

T or F: After k iterations, Fermat's test for prime numbers determines that Carmichael numbers are composite with probability at least $1 - 2^{-k}$.

T or F: Suppose that a particular RSA encryption scheme uses a modulus N and a public exponent e that are not relatively prime. Then a problem with this scheme is that two different messages less than N may be encrypted into the same ciphertext.

T or F: Universal hash function families are useful because they guarantee something no single hash function can: for *any* data set, the probability that two items x and y collide is only $1/n$, where n is the number of possible hash values.

T or F: The extended Euclid's algorithm can be used to determine the inverse of 5 modulo 2007.

T or F: $\Omega(n^2)$ is a lower bound on the time complexity for any sequential algorithm multiplying two general n-bit numbers because all n^2 pairs of bits must be considered at least once.

The last 5 questions have four possible answers: T, F, T if P=NP (indicated by =), or T if P $\neq$ NP (indicated by $\neq$). As before, circle the correct answer.

T, F, = or $\neq$: Every NP complete problem is in NP.

T, F, = or $\neq$: Every NP complete problem is in P.

T, F, = or $\neq$: There is a reduction of the Halting Problem to SAT.

T, F, = or $\neq$: Suppose A and B are in NP. If A reduces to B and A is in P, then B is in P.

T, F, = or $\neq$: If A reduces to B and A is NP-complete, then B is not in P.