Final Exam

Instructions:

Relax. Smile. John has certified that the exam isn't too long or too hard. You'll do great.

As you go through these questions please keep in mind that I have a strong preference for correct answers! You'll end up with a better score doing half the problems completely correctly than all the problems "half way correct."

If you don't understand a question, please ask!

Remember that if I say "define" I'm not looking for an "idea" — I want a proper definition.

Algorithms and proofs should be clear and succinct. If it ain't clear, it ain't right!

Good luck and happy holidays,

— *Phil Rogaway*

Your name:

Name of your favorite algorithm:

On problem	you got	out of
1		70
2		30
3		20
Σ		120

1 Short answers [60 points]

A reminder — where it says "define" you should be mathematically precise.

1.1. *(5 points)* Define $O(n^2)$.

1.2. *(5 points)* Define a ***tree*** (as it arises in graph theory, not forestry). (Note that I said "tree" and not "binary tree").

1.3. *(5 points)* Define what it means for a problem Π to be ***NP-Hard***.

1.4. *(10 points)* Prove that the amortized cost of incrementing an n-bit counter, C, is $O(1)$. Use as your potential function $\Phi(C) = \#$ of 1's in C. Regard the actual cost of incrementing C to be the number of bits which you change.

1.5. *(5 points)* You are given 5 distinct numbers to sort. Describe an algorithm which sorts them using at most 6 comparisons, or prove that no such (comparison-based) algorithm exists.

1.6. *(5 points)* Solve the following recurrence relation using a recursion tree, finding a closed form expression $T(n) = \Theta(\cdots)$. Assume $T(n)$ is constant for sufficiently small n. Show your work.

$$T(n) = T(n/3) + T(n/5) + n$$

1.7. *(5 points)* Draw the "string matching automaton" for the pattern $P = 01001$. Assume an alphabet of $\Sigma = \{0, 1\}$.

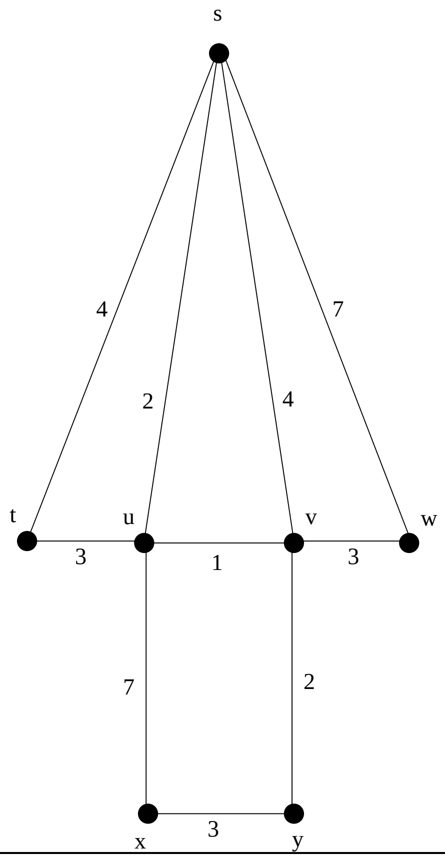
1.8 *(10 points)* Recall Huffman codes — they were those variable-length, prefix-free binary codes that minimize total encoding length, assuming that you know the relative frequency of each character. Construct a Huffman code for the characters `a`, …, `f`, assuming frequencies as specified in the following table:

char	frequency	code
`a`	12	
`b`	7	
`c`	10	
`d`	40	
`e`	13	
`f`	33	

Show your work. When you're finished, go back and fill in the column labeled "code" with the correct binary codeword for each character.

1.8 *(5 points)* Find the shortest path between s and every other node t by using Dijkstra's algorithm. Show your work by neatly changing the nodes' d-labels (distance upper bounds) as the algorithm progresses.

Warning: You won't get credit if you put down the right answers but it's not clear that you did this by running Dijkstra's algorithm.



1.9 *(5 points)* Professor Glizzenblock has invented a fancy new data structure, which he calls a G-heap. A G-heap supports the same operations as a Fibonacci heap (F-heap) but with different time bounds:

—INSERT takes $O(1)$ amortized time;

—EXTRACT-MIN on an n-node heap takes $O(\lg^* n)$ amortized time;

—DECREASE-KEY on an n-node heap takes $O(\lg^* n)$ amortized time.

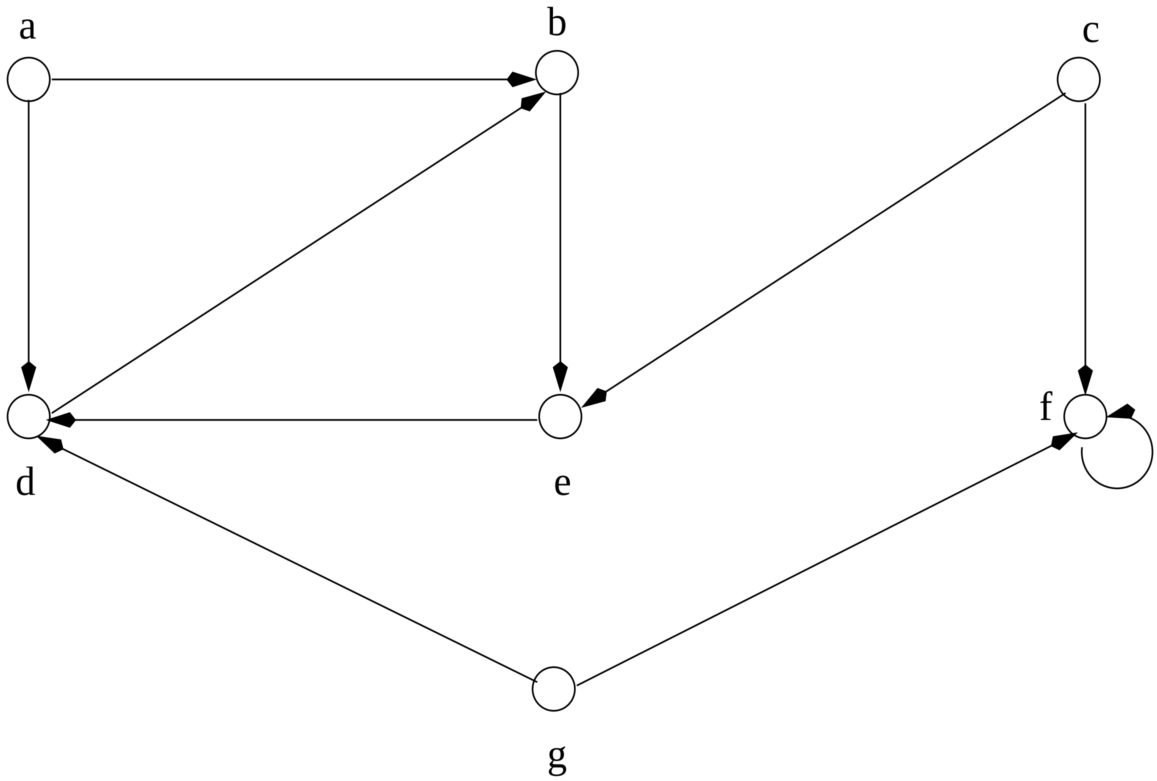
(If you've forgotten the definition of $\lg^*$ don't worry; it's not relevant.) If you implement Dijkstra's algorithm using a G-heap, what will be its running time? Answer with as tight an $O(\cdots)$ as you can. No justification required.

1.10 *(10 points)* Do a Depth-First Search (DFS) on the following digraph of $n = 7$ nodes. Assume the adjacency list gives the vertices in alphabetical order, and, in the outer loop of DFS, you explore vertices in alphabetical order.

Write the ***discovery time*** and ***finish time*** on each vertex. Do this in the form d/f (eg., 3/10 if a vertex was discovered at time 3 and finished at time 10). The first vertex discovered should have discovery time of 1 and the last vertex finished should have finish time $2 \cdot n = 14$.

Above every edge write one of the words ***tree***, ***back***, ***forward*** or ***cross*** to indicate if this edge is a tree edge, back edge, forward edge, or cross edge.

Do your work carefully; I'm unlikely to give partial credit.



2 Dynamic Programming [30 points]

Santa Claus is packing his sleigh with gifts. His sleigh can hold no more than c pounds. Santa wants to choose among n different gifts. (He has one of each gift.) Gift i has utility u_i (the amount of happiness gift i induces in some child) and weight w_i. The weight of a set of gifts is the sum of their weights, and the utility of a set of gifts is the sum of their utilities. All numbers mentioned are positive integers. Your job is to devise an algorithm that lets Santa maximize the utility of what he loads on his sleigh without exceeding its capacity c.

2.1. *(5 points)* A *greedy algorithm* for this problem takes the gifts in order of increasing weight until the sleigh can hold no more gifts. Give a small example to demonstrate that the greedy algorithm does not generate an optimal choice of gifts.

2.2 *(10 points)* Let $H_{k,x}$ be the maximal achievable utility if the gifts are drawn from $1, \ldots, k$ (where $k \leq n$) and these gifts may weigh at most x pounds. Write an expression for $H_{k,x}$ which is useful for developing a dynamic programming solution to this problem.

2.3. *(10 points)* Write clear and correct pseudo-code for a (reasonably efficient) function which computes $H_{k,x}$.

2.4. *(5 points)* How long will it take your program to calculate $H_{n,c}$? State your answer (as $\Theta(\cdots)$) as a function of c and n.

3 NP-Hardness **[20 points]**

A graph $G = (V, E)$ is ***k-colorable*** if there is a way to paint the vertices V using "colors" $1, 2, \ldots, k$ such that no adjacent vertices are painted the same color. Consider the following two problems:

GRAPH 3-COLORABILITY
Instance: A graph $G = (V, E)$
Question: Is G 3-colorable?

GRAPH 4-COLORABILITY
Instance: A graph $G = (V, E)$
Question: Is G 4-colorable?

In lecture we showed that GRAPH 3-COLORABILITY is NP-hard. Now show that GRAPH 4-COLORABILITY is NP-hard. To do this, use the NP-hardness of GRAPH 3-COLORABILITY — don't go all the way back to 3SAT.