Final

Name:

TA:

Section Time:

Course Login:

Person on Left:

Person on Right:

Answer all questions. Read them carefully first. Be precise and concise. Write your answers in the space provided. We will provide separate scratch paper. Do NOT turn in the scratch paper!

The exam has a total of 180 points.

Good luck!

1 True/False [2 points each]

Write either True or False to the *left* of the number.

1. If an undirected connected graph has the property that between any two nodes u and v, there is *exactly* one path between u and v, then that graph is a tree.

2. Modular exponentiation is known to take polynomial time. (Modular exponentiation takes numbers a, b, c—all in binary—as input and returns $a^b \pmod c$ in binary as output).

3. Finding all solutions to the equation $x^2 = a \pmod N$ is known to be solvable in polynomial time.

4. Dijkstra's algorithm will find the correct shortest distances from a given starting node on a directed graph with negative edges provided there are no negative cycles.

5. Let $T(n)$ be the time taken by some algorithm on inputs of size n. If $T(2n) \leq (T(n))^2$ for all positive n, then the algorithm runs in polynomial time.

6. Suppose that the running time $T(n)$ of an algorithm on inputs of size n satisfies the recurrence $T(n) = T(n/2) \log n + O(n)$. Then $T(n) = O(n^c)$ for some finite positive constant c.

7. Given a problem B, if there exists an NP-complete problem that can be reduced to B in polynomial time, then B is NP-complete.

8. If P $\neq$ NP, then any search problem in NP can be reduced to any other problem in NP in polynomial time.

9. Given any connected undirected graph G, there exists a maximum independent set of G that is also a vertex cover of G.

10. Given any NP-complete optimization problem, a b-approximate solution can be found in polynomial time for some positive finite constant b.

11. Any acyclic undirected graph is 2-colorable.

12. Given an undirected, unweighted, connected graph G, suppose we run a DFS on G starting on some node s. We find that the DFS tree has the property that for any vertex v, the path in the DFS tree from s to v is a shortest (fewest number of edges) path from s to v in G. G is a tree.

13. For integers $a, N > 2$, if $a^{N-1} \neq 1 \pmod{N}$ then N is not prime.

14. For positive distinct integers a and b, if for an $x \neq 0 \pmod{ab}$, $x^{(a-1)(b-1)} \neq 1 \pmod{ab}$ then either a or b is not prime.

15. 8 has a multiplicative inverse mod 99.

16. If the maximum flow problem is not NP-complete, then P $\neq$ NP.

17. Consider a DFS in a directed graph (which has an edge (u, v)) where pre$(u) = 5$, post$(u) = 8$, and pre$(v) = 3$ and post$(v) = 10$. This graph must have a cycle.

18. The node with the highest post order number in a depth first search of an undirected connected graph is always the starting node.

19. The node with the highest post order number in a depth first search of a directed graph is in a sink strongly connected component.

20. One can always make a directed graph with vertices V strongly connected by adding at most $|V|$ edges.

21. Given two points $\mathbf{v_1}$ and $\mathbf{v_2}$ that each satisfy a set S of linear constraints, the point $\frac{\mathbf{v_1}}{2} + \frac{\mathbf{v_2}}{2}$ also satisfies all the constraints in S.

22. Given a linear program on n non-negative variables, with more than n constraints, the linear program has a bounded optimal value.

23. The linear program for the maximum flow problem on a graph with integer capacities has an optimal solution where all the variables have integer values.

24. Consider the 8 node directed graph on the professor and the course staff with edges from older to younger (assuming no ties), where each edge has capacity 1. To clarify, note that there is a directed edge (a, b) from a to b if a is older than b. There is a flow of value 7 from the oldest (the professor) to the youngest.

25. Chapter 9 suggests that backtracking can be used to solve all problems in NP in polynomial time.

26. The cost of a TSP tour with a given set of distances is never less than the cost of a minimum spanning tree of the complete graph where the costs of the edges are the distances in the TSP instance.

27. There is a graph with 12 nodes which has a vertex cover of size 5 and a matching of size 6.

28. A valid superposition for two qubits is $\frac{1}{2}|00\rangle + \frac{1}{2}|01\rangle$.

2 Let's work some examples. [32 points]

No need to justify.

1. What is $5^{49} \mod 35$?

2. What is x, when $8x = 7 \pmod{99}$?

3. What is the FFT of $[0, 1, 0, 0]$?

5. Given a positive integer d, what is a set of frequencies for 2^d characters where the optimal prefix code has an associated tree of depth exactly d? (The depth of a tree is the length of the path from the root to the deepest leaf.)

6. Give an example of a set cover instance with at most 5 sets and 10 elements where the greedy algorithm always produces a sub optimal solution.

7. Express the problem of finding an optimal row strategy for the following game as a linear program. The row player wants to maximize the payoff here. $\begin{bmatrix} 1 & 2 \\ 3 & 1 \end{bmatrix}$

8. Describe (give the coordinates of) one vertex of the resulting linear program from the previous problem.

3 Where have I seen this before? [18 points]

These problems should have very short answers given knowledge of the material. You may provide a brief justification, but it is not necessary.

1. Given a directed graph, explain how to find all edges that are not part of a cycle.

2. We wish to determine the most efficient way to deliver power to a network of cities. Initially, we have n cities that are all disconnected. It costs $c_i \geq 0$ to open up a power plant at city i. It costs $r_{ij} \geq 0$ to build a cable between cities i and j. A city is said to have power if either it has a power plant, or it is connected by a series of cables to some other city with a power plant (in other words, the cables act as undirected edges). Give an efficient method for finding the minimum cost to power all the cities.

3. We are given a set of n lists of sizes n_i. A merge takes as input two lists. If the lists are of size ℓ and r, the merge takes $\ell + r$ operations to complete. How should we merge the lists into a single list using pairwise merges while minimizing the total number of operations?

4 How low can you go? [24 points]

1. (Warm up.) Consider an undirected *weighted* connected graph. We wish to find the shortest path between nodes s and t that takes *exactly* K edges (the path can visit each node and edge any number of times). We will give an efficient algorithm for this problem. Let $L(v, k)$ be the length of the shortest path from s to v using k edges. What are the base cases and the recurrence to compute $L(v, k)$?

2. Given a row of n houses that can each be painted red, green, or blue (with a cost $P(i, c)$ for painting house i with color c), devise an algorithm to find a minimum cost coloring of the entire row of houses such that no two adjacent houses are the same color.

3. Given a string s, we wish to do a series of operations to change s into a palindrome. More specifically, we know that adding a character c at any position costs $\texttt{add}(c)$ and deleting a character c at any position costs $\texttt{rem}(c)$ (all these costs are non negative and can change depending on the character). Give an efficient algorithm to determine the minimum cost needed to change s into a palindrome.

5 Half-way there [20 points]

Recall that the knapsack problem takes in a set of n items with profit p_i and size s_i with size bound B. We want to find a subset I of the items which maximizes profit given that the size is less than the bound (we can only use each item once). You may also assume that all $p_i \geq 0$, and $0 < s_i \leq B$

We know a dynamic programming solution exists which unfortunately takes time $O(Bn)$ (exponential in terms of the input), so we create some greedy approximations:

1. **Greedy:** Assume items are sorted according to decreasing $\frac{p_i}{s_i}$ (breaking ties arbitrarily). While we can add items, greedily pick items in the given order.

   Show that this greedy algorithm has no nonzero constant approximation ratio.

2. We make one very small adjustment to our greedy algorithm from the previous part

   Almost Greedy: Assume items are sorted according to decreasing $\dfrac{p_i}{s_i}$ (breaking ties arbitrarily). Greedily add items until we hit an item a_i that is too big (can't fit in the knapsack). Pick the better set of $\{a_1, \ldots, a_{i-1}\}$ and $\{a_i\}$

   Show that this algorithm has an approximation ratio of $\frac{1}{2}$.

6 Reductions [30 points]

In this problem, you may use the NP-completeness results described in the book.

1. In the 4D matching problem, one has equal-sized sets of boys, girls, pets and houses, and a set of consistent 4-tuples, each consisting of a boy, a girl, a pet and a house. One then needs to find a subset S of the consistent tuples such that every boy, girl, pet, and house is a member of exactly one tuple in S. Show that this problem is NP-complete.

2. We will produce a reduction from 3SAT to node disjoint paths; given a directed graph $G = (V, E)$ and a set of terminal pairs $(s_1, t_1), \ldots, (s_k, t_k)$ find a path connecting each terminal pair where no two paths have a node in common.

   Given a formula ϕ, we make a graph consisting of a pair of terminals (s_x, t_x) for each variable x in ϕ and a pair of terminal (s_c, t_c) for each clause c in ϕ. We further make a node v_ℓ for each literal, ℓ, in the formula (recall a literal is an occurrence of a variable, negated or not, in a clause). We also make a pair of paths between the terminal pair (s_x, t_x) for variable x, as follows. One path goes through all literals of form x and one path goes through all the literals of from $\overline{x}$. Note that for a variable x, s_x and t_x can only be connected using one of the two paths; one of these corresponds to True, and the other corresponds to False.

   Given a clause c, $(\ell_i \vee \ell_j \vee \ell_k)$ where ℓ_i, ℓ_j and ℓ_k are literals, describe which nodes the clause terminals s_c and t_c should be connected to or from so that the resulting graph has a set of node-disjoint paths connecting all the terminal pairs if and only if the formula, ϕ, is satisfiable. (No need to prove correctness for this part.)

3. Show that given a graph G and a positive integer k, the problem of finding a connected spanning subgraph where every node has degree at most k is NP-complete. A spanning subgraph of $G = (V, E)$ is another graph $G' = (V, E')$ where $E' \subseteq E$ (notice that the set of vertices is the same).