CS570
Analysis of Algorithms
Spring 2014
Exam III

Name: ______________________
Student ID: __________________
USC Email: __________________

____On Campus ____DEN

	Maximum	Received
Problem 1	20	
Problem 2	15	
Problem 3	10	
Problem 4	20	
Problem 5	15	
Problem 6	20	
Total	100	

2 hr exam
Closed book and notes

If a description to an algorithm is required please limit your description to within 150 words, anything beyond 150 words will not be considered.

1) 20 pts
Mark the following statements as **TRUE** or **FALSE**. No need to provide any justification.

[TRUE/FALSE]
Not every decision problem in P has a polynomial time certifier.

[TRUE/FALSE]
If P!=NP and if two decision problems A and B both in NP are polynomial time reducible to each other, then A and B are both in NP-Complete.

[TRUE/FALSE]
If any two problems in NP are polynomial time reducible to each other then P=NP-Complete.

[TRUE/FALSE]
The maximum weight edge cannot be part of any minimum spanning tree for a graph that has cycles.

[TRUE/FALSE]
If a problem is not solvable in polynomial time, it is in the NP-Complete class.

[TRUE/FALSE]
In a flow network whose edges have capacity 1, the maximum flow always corresponds to the maximum degree of a vertex in the network.

[TRUE/FALSE]
Linear programming is at least as hard as the Max Flow problem in a flow network.

[TRUE/FALSE]
Bellman-Ford algorithm is more suitable for distributed processing than Dijkstra's algorithm is.

[TRUE/FALSE]
Suppose $f(n) = 8f\left(\frac{n}{2}\right) + 56$, then $f(n) = \Theta(n^3)$.

.

[TRUE/FALSE]
Suppose $f(n) = f\left(\frac{n}{2}\right)$+56, then $f(n) = \Theta(n)$.

2) 15 pts
Prove that the problem of deciding if a given graph G contains a simple cycle that visits at least half of the vertices in G is NP-complete.

3) 10 pts
Prove or disprove the following statement: For every 3-SAT instance there exists an assignment that satisfies at least half the clauses.

4) 20 pts

A network of n servers under your supervision is modeled as an undirected graph G=(V,E) where a vertex in the graph corresponds to a server in the network and an edge models a link between the two servers corresponding to its incident vertices. Assume G is connected. Each edge is labeled with a positive integer that represents the cost of maintaining the link it models. Further, there is one server (call its corresponding vertex as *S*) that is not reliable and likely to fail. Due to a budget cut, you decide to remove a subset of the links while still ensuring connectivity. That is, you decide to remove a subset of E so that the remaining graph is a spanning tree. Further, to ensure that the failure of *S* does not affect the rest of the network, you also require that *S* is connected to exactly one other vertex in the remaining graph. Design an algorithm that given G and the edge costs efficiently decides if it is possible to remove a subset of E, such that the remaining graph is a spanning tree where *S* is connected to exactly one other vertex and (if possible) finds a solution that minimizes the sum of maintenance costs of the remaining edges.

5) 15 pts

You are given a flow network with source s, sink t and edge capacities. Further, for every vertex v in the network, you are given a non-negative number d_v which is the vertex capacity. Design a polynomial time algorithm that finds a flow of maximum value that (in addition to satisfying the usual edge capacity constraints and flow conservation constraints) satisfies the constraint that for every vertex v, the flow into v is at most d_v.

6) 20 pts
Alice and Bob worked in a restaurant and received n currency notes in total as tips. Every note has a value (either \$1, \$5 or \$10) written on it. The currency notes are arranged from left to right on a table in a fixed but arbitrary sequence. In particular, they are not necessarily sorted according to value. Alice and Bob play the following game to split the tip money. Alice and Bob take turns to play and at each turn, the player chooses either the leftmost currency note or the rightmost currency note and takes it. Bob is greedy and always plays using the following strategy; “If the rightmost note has value larger than the leftmost, then take the rightmost. Otherwise take the leftmost". Design an efficient algorithm that determines the plays for Alice such that the tip money Alice gets is maximized. Assume n is even and Alice plays the first turn.

Additional Space