CSCI 2300 Final Exam
Fall 2021.

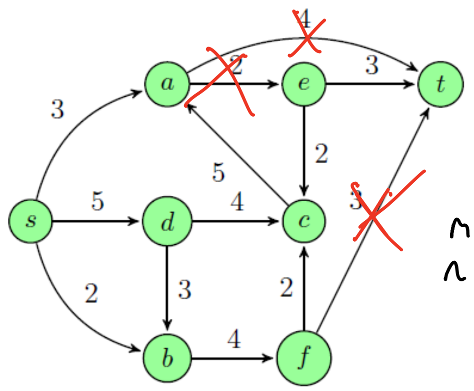
Open notes and textbook use are permitted. You may use electronic devices to compose your answers, but you may not use them to solve problems or check your work. There are 4 questions adding to 80 points + 1 OPTIONAL BONUS question which you do not have to do.

120mins exam time

LAST NAME	FIRST NAME	RIN NUMBER
-----------------	-------------------	----------------------

Zack answer

Question 1 [20 points]:



1.1. [3 points]:
Consider the graph in the figure with edge capacities as shown. Assume that the Ford–Fulkerson algorithm would stop with max flow 3+4+2=9 on this graph.

List the edges that form a minimum capacity s-t cut for this graph.

min-cut
max-flow

a - e → 2
a - t → 4
f - t → 3

1.2. [3 points] What is the time complexity of the Ford–Fulkerson algorithm? Why is it considered to be pseudo-polynomial time?

O(E·f) → f is unknown, this would no longer be polynomial

1.3. [2 points]: What are the two elements of dynamic programming?

overlapping subproblems
optimal substructure

1.4. [2 points]: Is dynamic programming a bottom-up or top-down approach? Why?

bottom-up

1.5. [2 points]: Is a greedy algorithm a bottom-up or top-down approach? Why?

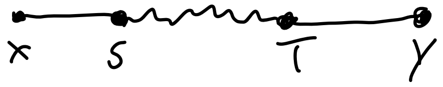
top-down

1.6. [1 point]: What is memoization?

Remembering/Storing data to avoid the cost of recalculation

1.7. [3 points]: Given an undirected graph G=(V,E), the Shortest Path Problem (SHP) asks you to find a path from vertex u to vertex v with the fewest edges. Prove or disprove that SHP has optimal substructure.

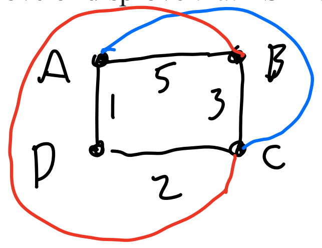
Proof by contra.



- If we know Sh. P x-y includes Sh. P x-S & T-y, we also know that Sh. P S-T must be taken too

1.8. [4 points]: Given an undirected graph G=(V,E), the Longest Simple Path problem (LSP) asks you to find a simple path (one with no cycles) from u to v with most edges. Prove or disprove that LSP has optimal substructure.

Proof by Contra./Counter



- Longest path A-C does <u>not</u> include the longest path B-C

Question 2 [20 points]: The classical Traveling Salesman Problem asks for the most efficient way to visit every city on a map exactly once, starting and ending at the same city.

Consider a variation where the salesman does not have to visit *every* city. Additionally, the salesman is paid a variable amount for each city he visits. He pays for his own travel, so his net profit is the sum of his payments and his expenditures. He still must return home at the end.

This can be formulated as a graph problem. Given an undirected graph G=(V,E,D,P), where D is a matrix of distances between vertices and P is a list of payouts per vertex, find a cycle that maximizes the profit of the salesman.

2.1. [5 points] What is the appropriate subproblem? Hint: It should include the *last* city being visited for that subproblem.

$$C(S,j) = \max_{i \neq j \in S} C(S-\{j\}, i) + (p_{ij} - d_{ij})$$

2.2. [10 points] Formulate this problem as a Dynamic Programming problem and give your algorithm.

```
high = 0
C({1}, 1) = 0
for s = 2...n:
    for all subsets of S length s containing 1:
        C(S,1) = ∞
```

2.3. [5 points] What is the total running time of your DP algorithm?

$$O(n^2 2^n)$$

$WD_p \quad WD_f \quad WD_m$

Question 3 [20 points]: Linear Programming.

3.1. [10 points]: Suppose you are making a weekly schedule. Your schedule has 20 weekday daytime (**WD**) hours, 10 weekday nighttime (**WN**) hours, and 10 weekend (**WE**) hours.

You can do *only one* of the following activities at a time:

1. Solve puzzles 2. Hang out with your friends 3. Do part-time programming work.

You have 3 goals to achieve:

1. You must solve exactly 50 puzzles. For each **WD** hour or **WE** hour you can solve 2 puzzles, but for each **WN** hour, you can only solve 1 puzzle (because you are tired :))
2. You must spend *at least* 10 hours with your friends, but they are not available during **WD** hours.
3. You must make money by working. $10/h for **WD** hours, $15/h for **WN** hours, $20/h for **WE** hours.

Model this problem as Linear Program to ensure that you make as much money as possible whilst solving all the puzzles and spending enough time with your friends.

You are NOT required to solve the linear program. You only need to formulate it.

$WD_p + WD_m \leq 20$ $\quad \max 10 \cdot WD_m + 15 \cdot WN_m + 20 \cdot WE_m$

$WN_p + WN_f + WN_m \leq 10$

$WE_p + WE_f + WE_m \leq 10$ $\quad 0 \leq WD_p, WD_f, WD_m \ldots$

$WE_f + WN_f \geq 10$

$2 \cdot WE_p + 2 \cdot WD_p + WN_p = 50$

3.2. [10 points]: What is the dual of the following LP formulation?

$\min 30y_1 + 24y_2 + 36y_3$

Max 3x1 +x2 +2x3
Subject to
y_1 x1+x2+3x3 <=30 $\quad y_1x_1 + y_1x_2 + y_1 3x_3 <= 30y_1$
y_2 2x1+2x2+5x3 <= 24 $\quad y_2 \cdot 2x_1 + y_2 2x_2 + y_2 5x_3 <= 24y_2$
y_3 4x1+x2+2x3 <=36 $\quad y_3 \cdot 4x_1 + y_3 x_2 + y_3 2x_3 <= 36y_3$
x1,x2,x3 >=0

$y_1 + 2y_2 + 4y_3 \geq 3$

$y_1 + 2y_2 + y_3 \geq 1$

$3y_1 + 5y_2 + 2y_3 \geq 2$

$$(y_1 + 2y_2 + 4y_3)x_1 + (y_1 + 2y_2 + y_3)x_2$$

$$+ (3y_1 + 5y_2 + 2y_3)x_3 <= \underline{30y_1 + 24y_2 + 36y_3}$$

Question 4 [20 points]: Reductions and NP completeness.

$(x \vee \bar{y}) \wedge (\bar{x} \vee y)$

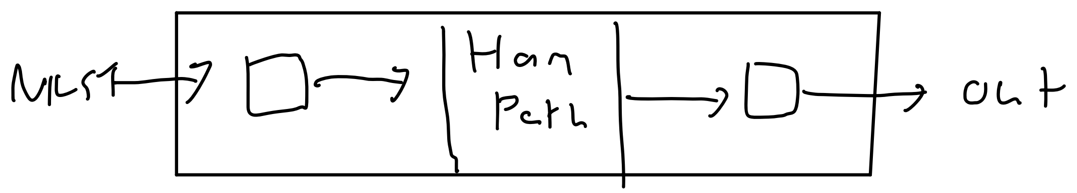
4.1. [10 points]: Show how we can solve any instance of a 2-SAT problem (i.e., each clause has at most 2 literals), in O(N) time, using a reduction to a problem solved via DFS. Hint: what should there be *N* of? Also, do you remember about strongly connected components?

1. Change bool formula to implication form
   $x \vee \bar{y} \rightarrow \bar{x} \Rightarrow \bar{y} \wedge y \Rightarrow x$
2. Make nodes for literals. Make C source node connect to one node. Make sink node ¬C that has an edge from the opp. of what C connected to. Add edges for each implication
3. Run SCC alg on G. if x and $\bar{x}$ are in the same SCC, NOT a solution to the problem

4.2. [10 points]: Consider the Minimum Leaf Spanning Tree problem: Given a graph G = (V, E) and an integer k, is there a spanning tree T in G that contains at most k leaves?

Show that MLST problem is NP-Complete by describing a reduction (hint: use the Hamiltonian Path problem, which asks if we can find a path that visits every vertex once).

By def, a Ham. path is a spanning tree with two leaves. If one is the source, then the existance of a Ham. path implies the existance of MLST with 1 leaves since k can't be 0 or negative, then an MLST for any k exists if a Ham. path does.

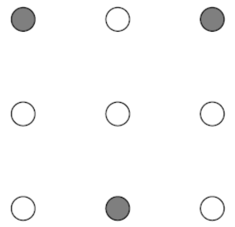


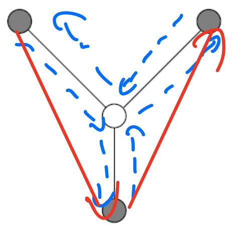
$k >= 1$

MLST must be NP-Complete since otherwise, we'd contradict known Hamiltonian

Bonus Question [16 points]: You do not have to solve this problem unless you like to get extra credit. Partial credit may be awarded, but it will be less generous than on other problems.

In the MINIMUM STEINER TREE problem, the input consists of: a complete graph $G = (V, E)$ with distances d_{uv} between all pairs of nodes; and a distinguished set of *terminal nodes* $V' \subseteq V$. The goal is to find a minimum-cost tree that includes the vertices V'. This tree may or may not include nodes in $V - V'$.





Suppose the distances in the input are a metric (recall the definition on page 277). Show that an efficient ratio-2 approximation algorithm for MINIMUM STEINER TREE can be obtained by ignoring the nonterminal nodes and simply returning the minimum spanning tree on V'. (*Hint:* Recall our approximation algorithm for the TSP.)

metric: $d(x,y) > 0$, $d(x,y) \leq d(x,z) + d(z,y)$

Red lines at most 2 · one blue