CIS 675 Spring 2021 Final Exam
Instructions:

1. Your exam grade will be based on your performance in the oral exam. The solution you present must match your written solution.

2. You are permitted to use your textbook, lecture materials, and existing online sources.

3. You are permitted to collaborate with other currently enrolled CIS 675 students. However, you are *strongly discouraged* from doing so. If you do not create your own solutions, you will do poorly on the oral exam.

4. You are *not* permitted to post exam questions online for others to answer (e.g., on StackOverflow or Chegg).

5. If you are confused about what any of the problems is asking, reach out to the instructor or TA immediately. It is better to clarify now than during the exam!

Problem 1: Suppose that you have M factories producing keychains, and N stores that can sell those keychains. Each factory i can produce up to F_i keychains, and each store j can sell up to S_j keychains. Keychains are shipped via directed trains- that is, each train runs in a particular direction, and there is not necessarily a train running in the opposite direction (though there might be). Assume that each train runs from: (a) a factory to a train station, (b) a factory directly to a store, (c) a train station to another train station, or (d) a train station to a store. You may assume that all values are integers.

Each train can carry an unlimited amount of keychains. However, at each train station, an inspector must check all of the goods being sent through that station, and because the number of inspectors is limited, each station can only accommodate a certain amount of keychains traveling through it. This amount is not necessarily the same for all train stations.

Your goal is to sell the maximum total number of keychains. Represent this problem as a network flow problem, and then use the Ford-Fulkerson algorithm to determine how many keychains each factory should produce, which stores those keychains should be sent to, and the route that should be used.

Problem 2: Suppose that you have a set of pipes carrying water from a lake to a city. The lake can supply an (effectively) unlimited amount of water, and the city wants as much water as possible. Each pipe is directed (i.e., there may or may not be a pipe in the opposite direction), and each pipe runs from: (a) the lake to a pipe junction, (b) one pipe junction to another pipe junction, or (c) a pipe junction to the city. Each pipe can carry a certain amount of water. This amount varies by pipe. You may assume that all values are integers.

Suppose that someone before you has already determined how much water to send along each pipe so that total flow of water from the lake to the city is maximized. Thus, you know how much water currently flows along each pipe, and you know that this is the maximum achievable. You do not know what process they used to determine this, but can assume that it is indeed optimal.

Now, one of the pipes is being replaced with a larger pipe. This larger pipe can accommodate up to one additional unit of water than the previous pipe. Create an efficient algorithm to determine how much water to send along each pipe so that the total amount of flow reaching the city is maximized.

Problem 3: Suppose you have a set of N farms that grow vegetables. Farm i can grow up to v_i units of vegetables. There are M cities that need a supply of vegetables. Each city j needs c_j units of vegetables. Vegetables are shipped by trains. Assume that each train's route segment runs from: (a) a farm to a train station, (b) a train station to another train station, (c) a train station to a city, or (d) a city to another city. Each train runs in a particular direction, and there may or may not be another train running along the opposite route segment. Each train can carry a certain amount of vegetables (this amount varies depending on the route segment).

In addition to carrying vegetables, each train carries passengers. Along each route segment, the passengers will need to eat a certain amount of vegetables (this amount varies depending on the route segment). Once eaten, these vegetables are gone: they can no longer be routed to a city. The passengers need this food regardless of whether the train ultimately carries vegetables to another train or city. In other words, each city demands a certain amount of vegetables, and in addition, each train route segment demands a certain amount of vegetables. You may assume that all values are integers.

Your goal is to determine whether it is possible to satisfy all demands (cities' and train passengers'). Represent this problem as a network flow problem, and then use the Ford-Fulkerson algorithm to determine how many vegetables should be sent along each route segment so that all demands are satisfied.

Problem 4: In the MaxClique problem, you are given an undirected, unweighted graph G and an integer k, and must return a clique of size at least k nodes in G, where a clique is a set of nodes that are all connected to one another. The largest clique in a graph is the clique with the greatest number of nodes. MaxClique is known to be NP-Complete. In the Double Connected Clique problem, you are given an undirected, unweighted graph G and an integer k, and must find the largest set of nodes S of size at least k with the following properties: (1) S consists of the union of two cliques S_1, S_2 where S_1 and S_2 do not share any nodes, and (2) every node in S_1 is connected to exactly one node in S_2, and vice versa. Prove that Double Connected Clique is NP-Complete.

Problem 5: In the HAMILTONIAN PATH problem, you are given an undirected, unweighted graph G and specified start and end nodes s and t, and must find a path in G from s to t that touches every node exactly once. HAMILTONIAN PATH is known to be NP-Complete. In the SINGLE NODE REPEAT HAMILTONIAN PATH problem, you are given an undirected, unweighted graph G and specified start and end nodes s and t, and must find a path in G from s to t that touches every node exactly once, except that at most one node can appear twice. Prove that SINGLE NODE REPEAT HAMILTONIAN PATH is NP-Complete.

Problem 6: We have discussed the SAT problem extensively in class. The 2T1F SAT problem is like SAT, except now instead of finding an assignment of truth values to literals so that each clause contains at least one true literal, we must find an assignment of truth values to literals so that each clause contains at least two true literals and one false literal. Assume that SAT is NP-Complete and prove that 2T1F SAT is NP-Complete.