Comp 305 Algorithms and
February 2021 Complexity

Amnesty Exam

- You have 120 minutes.
- The exam is closed book, closed notes except your cheat sheet.
- You will get 10% of the credit for blank answers.

Name	

For staff use only:

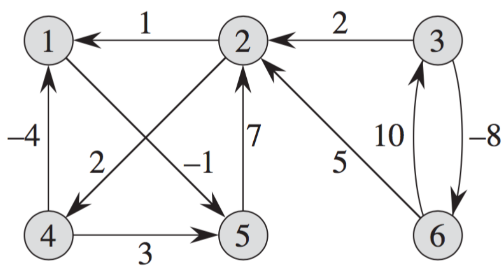
Q1.	All-pairs shortest paths	/15
Q2.	Flow networks	/10
Q3.	Linear programming	/15
Q4.	Minimum spanning tree	/30
Q5.	NP-Completeness	/30
	Total	/100

Q1. [15 pts] All-pairs shortest paths

Floyd-Warshall is a dynamic programming algorithm to solve the all-pairs shortest paths problem. It uses the following recurrence:

$$d_{ij}^{(k)} = \begin{cases} w_{ij} & \text{if } k = 0\,, \\ \min\left(d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{(k-1)}\right) & \text{if } k \geq 1\,. \end{cases}$$

where $d_{ij}^{(k)}$ is the shortest path between i, j where all the intermediate vertices belong to the set $\{1, \ldots, k\}$. Run the Floyd-Warshall algorithm on the weighted directed graph given below. Write down the matrix $D^{(k)} = (d_{ij}^{(k)})$ for each value of $k \in \{0, \ldots, |V|\}$.



Q2. [10 pts] Flow networks

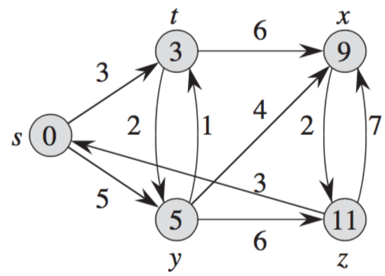
A (net) flow on G is a function $f : V \times V \to R$ satisfying the following:

- Capacity constraint: For all $u, v \in V$, $f(u,v) \le c(u,v)$.
- Flow conservation: For all $u \in V - \{s,t\}$, $\sum_{v \in V} f(u,v) = 0$.
- Skew symmetry: For all $u, v \in V$, $f(u,v) = -f(v,u)$.

Prove that flows in a network form a convex set. That is, show that if f_1 and f_2 are flows, then so is $\alpha f_1 + (1-\alpha) f_2$ for all α in the range $0 \le \alpha \le 1$.

Q3. [15 pts] Linear programming

Write out explicitly the linear program corresponding to finding the shortest path from node s to node y in the following figure.



Q4. [30 pts] Minimum spanning tree

In this problem, we give pseudocode for three different algorithms. Each one takes a connected graph and a weight function as input and returns a set of edges T . For each algorithm, either prove that T is a minimum spanning tree or prove that T may not be a minimum spanning tree by giving a counterexample.

a. MAYBE-MST-A(G, w)

```
sort the edges into nonincreasing order of edge weights w
T = E
for each edge e, taken in nonincreasing order by weight
    if T − {e} is a connected graph
        T = T − {e}
return T
```

b. MAYBE-MST-B(G, w)

```
T = ∅
for each edge e, taken in arbitrary order
    if T ∪ {e} has no cycles
        T = T ∪ {e}
return T
```

c. MAYBE-MST-C(G, w)

```
T = ∅
for each edge e, taken in arbitrary order
    T = T ∪ {e}
    if T has a cycle c
        let e′ be a maximum-weight edge on c
        T = T − {e′}
return T
```

Q5. [30 pts] NP-Completeness

Bonnie and Clyde have just robbed a bank. They have a bag of money and want to divide it up. For each of the following scenarios, either give a polynomial-time algorithm, or prove that the problem is NP-complete. The input in each case is a list of the n items in the bag, along with the value of each.

a. The bag contains n coins, but only 2 different denominations: some coins are worth x dollars, and some are worth y dollars. Bonnie and Clyde wish to divide the money exactly evenly.

b. The bag contains n coins, with an arbitrary number of different denominations, but each denomination is a nonnegative integer power of 2, i.e., the possible denominations are 1 dollar, 2 dollars, 4 dollars, etc. Bonnie and Clyde wish to divide the money exactly evenly.

c. The bag contains n checks, which are, in an amazing coincidence, made out to "Bonnie or Clyde." They wish to divide the checks so that they each get the exact same amount of money.