H9.

PROBLEM 1 *Long stable matches*

We showed in class that the stable matching algorithm ends in $O(n^2)$ time with a stable match. Find a set of preference inputs for 6 proposers and reviewers that requires at least 30 (in general, $n \cdot (n-1)$) proposals to be made before a stable matching is find no matter which order the proposals are made. Your answer can just list the preferences of the proposers and the reviewers.

Solution:

Consider a set of preferences:

P1: $1 > 2 > 3 > 4 > 5 > 6$	R1: $2 > 3 > 4 > 5 > 6 > 1$
P2: $2 > 3 > 4 > 5 > 1 > 6$	R2: $3 > 4 > 5 > 6 > 1 > 2$
P3: $3 > 4 > 5 > 1 > 2 > 6$	R3: $4 > 5 > 6 > 1 > 2 > 3$
P4: $4 > 5 > 1 > 2 > 3 > 6$	R4: $5 > 6 > 1 > 2 > 3 > 4$
P5: $5 > 1 > 2 > 3 > 4 > 6$	R5: $6 > 1 > 2 > 3 > 4 > 5$
P6: $1 > 2 > 3 > 4 > 5 > 6$	R6: $1 > 2 > 3 > 4 > 5 > 6$

After the first 6 proposals, P2–P6 will be matched, but P1 will be broken. P1 breaks P2, who breaks P3, P4, P5 who breaks P6, who breaks P1,...etc, until finally P1 asks P6 on the 31 proposal, and the algorithm finishes.

PROBLEM 2 *Fair Tug of War*

Recall the problem from homework 3 in which you devised an algorithm for Tug of War. Consider the closely related Fair Tug of War problem (FTW): given weights of n people, $W = (w_1, w_2, \ldots, w_n)$, divide the people into two teams, L, R, such that every person in W is either in L or in R, and such that $\sum_{\ell \in L} w = \sum_{r \in R} r$. In other words, the two teams include everyone, and both teams sum to exactly the same weight. Note: the team sizes are not necessarily evenly split as was required in the homework tug of war problem.

The subset sum problem is a famous NP-complete problem: given a list of non-negative numbers $L = (x_1, x_2, \ldots, x_n)$ and a target t, find a subset $S \subseteq L$ of numbers that sum to t, i.e., $\sum_{s \in S} s = t$.

Show that our Fair Tug of War problem is NP-complete.

Solution: First, note that FTW is in NP. Given a witness, namely a list of people on one team, it is easy to verify that both teams have equal sums of weights. To show that FTW is NP-complete, we will show

$$\textsc{subset-sum} \leq \textsc{ftw}$$

Given an instance of subset-sum (L, t), construct the set

$$W = L \cup \{z, 2t\}$$

where $z = \sum_{x \in L} x$ and run $FTW(W)$.

- If there exists a subset $M \subseteq L$ such that $t = \sum_{x \in M} x$, then notice that the set $M \cup \{z\}$ sums to $z + t$, and the remaining elements plus $2t$ sum to $(z - t) + 2t = z + t$, and thus there is an FTW solution.
- If there exists a fair-tug-of-war in the instance W, then note each team must have weight $(z + z + 2t)/2 = z + t$. This means that the elements $z, 2t$ must be on different teams. Consider the team with the element z. Remove z, and the remaining team M consists of elements of L which sum to t, and thus the subset-sum instance (L, t) has a solution.

The tricky parts of this reduction are to ensure that the instance of FTW contains only positive elements, and to ensure that if you add two elements to the set L, that those two elements end up on different sides of the FTW solution. Reductions that add one element $2z - t$ may not work because $2z - t$ may be negative.

PROBLEM 3 *Ministry of Museums*

The ministry of museums wants to ensure that every town either has a museum or is next to a town that has a museum. Given a map of a country consisting of towns and roads between them, the *Museum* problem is to decide whether k museums can be placed in towns so as to satisfy the Ministry's goal.

Let language $L = \left\{ (G,k) \;\middle|\; \begin{array}{l} G \text{ is a graph, and } k \text{ museums can be placed} \\ \text{at verticies such that every vertex either has} \\ \text{a museum or is adjacent to one with a mu-} \\ \text{seum.} \end{array} \right\}$.

1. Argue that $L \in NP$. (Recall, this means showing that it is easy to verify an instance in polynomial time given a witness.)

   Solution: First, observe that $L \in NP$ because the museum property is efficiently verifiable. In other words, given a set S of k verticies, one can efficiently check that every $v \in V$ is either in S or adjacent to a node in S.

2. Next show that L is NP-complete by showing a reduction from one of the NP-complete problems we discussed.

Solution:

To show that L is NP-complete, given that $L \in NP$, we show $L_{vc} \leq_p L$. Note that vertex cover is a *different* problem. In VC, one hopes to cover every edge. So a graph with k nodes and zero edges has a VC of size 0. The property in L is different; in the same instance, one requires k museums.

Claim 1 $L_{vc} \leq_p L$.

The transformation Given an instance of the vertex cover problem (G, k), we produce an instance (G', k') of the L problem using the transformation F below.

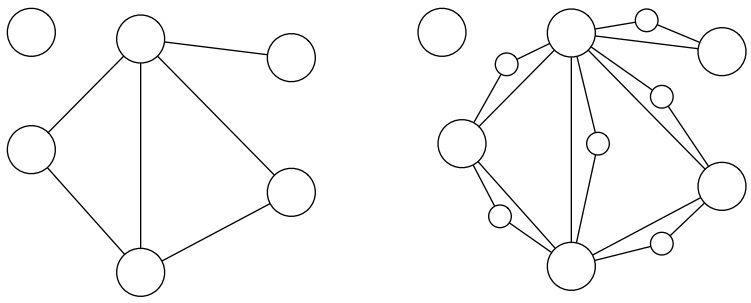

Figure 1: A vertex cover instance (G, k) on the left, and how it has been transformed into an instance of Museum, $\textsc{f}(G) = (G', k+1)$ on the right. The smaller nodes represent the "new nodes" $a_{u,v}$ that are inserted by $\textsc{f}$.

.

$\textsc{f}(G, k)$

1 Create graph G' by copying graph $G = (V, E)$.
2 Let ℓ be the number of *lonely* nodes in G, i.e. nodes with no edges.
3 **for** each edge $(u, v) \in E$
4 Add node $a_{u,v}$ to G'
5 Add two edges $(u, a_{u,v})$ and $(a_{u,v}, v)$ to G'.
6 Output $(G', k + \ell)$

An example of the transformation applied to a graph is given in Fig. 1.

Claim 2 *If* $(G, k) \in L_{vc}$, *then* $\textsc{f}(G, k) = (G', k + \ell) \in L_a$.

Proof. Let set S be a vertex cover of size k for graph G. Let I be the set of ℓ lonely nodes with no edges in G. By placing libraries at all nodes in $S \cup I$, then every vertex v in G' either has a library or is adjacent to one: If $v = a_{i,j}$ was one of the added nodes, then either node i or j must be in the vertex cover S. Therefore, v will be adjacent to a node with a library. If v is one of the original nodes in G (and has an edge e), then either $v \in S$, or its neighbor by edge e must be in S (because edge e must be covered by set S). Finally, if v is an original node in G but has no edge, then $v \in I$, and so v also has a library. Observe that the size of $|S \cup I| = k + \ell$. Therefore, $(G', k + \ell) \in L_a$. □

Claim 3 *If* $\textsc{f}(G, k) = (G', k + \ell) \in L_a$, *then* $(G, k) \in L_{vc}$.

Proof. Let L be the set of $k + \ell$ libraries that establishes that $(G', k + \ell) \in L_a$. We need to show that $(G, k) \in L_{vc}$ by constructing a vertex cover of size k for G.

If all of the libraries in L for G' are placed at "original nodes" of G and each town only has one library, then our transformation becomes simple. Let I be set of lonely nodes with no edges. Notice that $I \subseteq L$. Set $S = L - I$. We argue that $|S| = k$ and S is a vertex cover for graph G. Consider any edge $(u, v) \in G$. Since

node $a_{u,v} \in G'$ must be adjacent to a library, either $u \in S$ or $v \in S$. Thus, edge (u, v) must be covered.

However, the library cover L for G' may contain libraries at new nodes such as $a_{u,v}$ which were not in the original graph. In this case, we can move the library to either node u or to node v and still maintain the property that towns $a_{u,v}$, u, and v either have a library or are adjacent to a town with a library. By performing sufficiently many moves, we can construct a new library cover L' in which all libraries are placed at "original nodes" of G.

There is one special case to handle when nodes $a_{u,v}$, u and v all have libraries. In this case, when we move library from $a_{u,v}$ to either u or v, then u may, for example, have two libraries. In this case, we simply remove the redundant library. Then, when we apply the transformation above to construct a vertex cover for G, we then end up with a vertex cover of size $k - 1$ instead of k. Thankfully, the vertex cover problem can be defined as follows:

$$L_{vc} = \{(G, k) \text{ such that } G \text{ has a vertex cover of size} \leq k\}$$

which handles the problem. Alternatively, we can modify our library moving procedure to move the library at $a_{u,v}$ to any node in the graph that does not have a library (if such a node exists) and maintain the exact size k. □