1. Pebbling is a solitaire game played on an undirected graph G, where each vertex has zero or more pebbles. A single pebbling move consists of removing two pebbles from a vertex v and adding one pebble to a neighbor of v (the vertex v must have ta least two pebbles before the move.)

   Given a graph $G = (V, E)$ and a pebble count $p(v)$ for each vertex v, the Pebble Destruction asks whether there is a sequence of pebbling moves that removes all but one pebble. Prove that this problem is NP-complete.

2. In the Multiple Interval Scheduling problem, each job requires a set of intervals of time during which it needs to use the processor. For example, a single job could require the processor from 10 AM to 11 AM, and again from 2 PM to 3 PM. If you accept this job, it ties up your processor during those two hours, but you could still accept jobs that need any other time periods, including the hours from 11 AM to 2 PM.

   For a given number k, we want to know if it is possible to accept at least k of the jobs so that no two of the accepted jobs have any overlap in time. Prove that this problem is NP-complete.

3. The ranking officer of the ROTC decides to postpone a picnic and must notify everyone else.

   Each ROTC person except the ranking officer reports to a single superior officer. If u is the superior officer of v, then v is the direct subordinate of u.

   To notify everyone of the postponement, the ranking officer first calls each of her direct subordinates, one at a time. As soon as each subordinate gets the phone call, he or she immediately notifies each of his or her direct subordinates, one at a time. The process continues this way until everyone has been notified. Note that in this process each person can only call direct subordinates.

   We can picture this process as being divided into rounds. At each round, each person who has already learned of the postponement can call exactly one of his or her direct subordinates on the phone.

   The number of rounds it takes for everyone to be notified depends on the sequence in which each person calls their direct subordinates. Give an efficient algorithm that determines the minimum number of rounds needed for everyone to be notified.

4. You are organizing a three-day event where several 30-minute sets of music will be played. You need to hire DJs according to the following constraints:

   - Exactly k sets of music must be played each day, and thus $3k$ sets altogether.
   - Each set must be played by a single DJ in a single music genre.
   - Each genre can only be played at most once per day.
   - Each candidate DJ has given you a list of genres they are willing to play.
   - Each DJ can play at most three sets during the entire event.

Suppose there are n DJs and g genres. Describe and analyze an efficient algorithm that either assigns a DJ and a genre to each of the $3k$ sets, or correctly reports that no such assignment is possible.

5. Suppose $G = (V, E)$ is a directed graph with edge weights that may be negative, but contains no negative cycles. Let $l(e)$ denote the length of edge e. A *price function* p is a function that assigns a real number to each vertex. Given any price function p, the *reduced cost function* l_p is:

$$l_p(u, v) = p(u) + l(u, v) - p(v).$$

   (a) Prove that every reduced cost function preserves negative cycles and shortest paths.

   (b) We say that a price function p is *feasible* if $l_p(u, v) \geq 0$ for all edges (u, v). Suppose there exists a vertex s that can reach every other vertex by a path, and define a price function p by letting $p(u)$ be the length of the shortest path from s to u. Prove that p is feasible.

   (c) Combine your two previous answers to show that computing shortest paths from k sources in a graph with negative weight edges can be accomplished with only one application of Bellman-Ford and $k - 1$ applications of Dijkstra.

6. Solve the following recurrences:

   (a) $T(n) = T(\lceil n/15 \rceil) + T(\lceil n/10 \rceil) + 2T(\lceil n/6 \rceil) + n$.

   (b) $T(n) = T(n-1) + 2n - 1, T(0) = 0$.

   (c) $T(n) = 5T(n/2) + n^2, T(1) = 1$.

   (d) $T(n) = 4T(n/2) + n^2, T(1) = 1$.

   (e) $T(n) = 3T(n/2) + n^2, T(1) = 1$.