Please hand in each problem on separate sheets with your name and netID on each. If a problem requires multiple sheets, please staple the sheets for that problem together.

Question 1

You are one of \(2n\) unlikely survivors from a plane that crashed onto a large mysterious island. After a few weeks of living on the beach, eating nothing but coconuts and raw fish, the group decides to begin a search of the island’s interior. Everyone agrees that this exploration should be done in pairs, to cover as much ground as possible without leaving anyone all alone in the jungle. What people can’t decide upon is who should get paired with whom.

Every survivor creates a preference list, ranking the relative desirability of being grouped with each of the other \(2n-1\) survivors. As the token computer scientist on the island, your assistance is requested in determining whether or not a “stable” matching can be found. As you might expect, a stable matching is a matching of all \(2n\) survivors such that no pair of survivors who aren’t matched to each other would both prefer to be so.

Your task is to either

a) provide an efficient algorithm that always finds such a stable matching, or

b) provide an example of preference lists for which no stable matching exists.

Question 2

On a less fanciful note, consider the following dilemma. Aliens from the planet Draxrok-7 have decided to destroy our planet. They have sent an interstellar battleship to do so, which appears to be impervious to every weapon we’ve ever created. Our top alien experts say there is only one possible way to prevent our total annihilation.

They have discovered (and don’t ask how) that this alien vessel is powered by \(n\) generators which are spread throughout the massive ship. They also know that there are \(n\) service robots, each of which follows its own fixed (and known) maintenance schedule. A schedule specifies when the robot is servicing each of the \(n\) generators. Note that a robot may sometimes be in transit, and thus at times will not be working on any generator. By the end of each day, each robot has serviced every generator exactly once. Furthermore, no two robots ever service the same generator at the same time.

The plan is as follows. Each of the robots will be infected with a custom-made virus (again, don’t ask how). While a virus can’t alter a robot’s schedule, a virus can cause a robot to sabotage a specific generator while performing its scheduled maintenance. The goal is simply to cause the sabotage of all the generators. But there is a problem. When a generator is sabotaged, it melts down, destroying not only the robot who did the sabotage, but also any other robot that later attempts to service it.

Unfortunately, this makes the task a bit trickier: For example, suppose we make robot \(r_1\) sabotage generator \(g_1\) and robot \(r_2\) sabotage generator \(g_2\). If robot \(r_2\)’s schedule has it servicing \(g_1\) before \(g_2\) but after \(r_1\) has serviced \(g_1\), then \(r_2\) will be destroyed when it visits generator \(g_1\) and thus will not have the opportunity to sabotage \(g_2\).

Give an efficient algorithm to specify which robot should sabotage which generator so that every generator is successfully destroyed.
**Hint:** Use the stable matching algorithm (the Gale-Shapley algorithm we did in class) to solve this problem.

**Question 3**

A few weeks ago, a master jewel thief daringly snatched the world’s largest diamond from the city museum in broad daylight. Since then, there have been a number of similar daytime heists in other cities in the area. The police suspect that some of these thefts are copy-cat crimes, done by lesser criminals, but they also suspect that others may actually have been done by the master thief herself. Since their primary goal is to catch the master thief, the police want to focus on thefts that might have been the master’s work. They’ve asked for your help in ruling out thefts that were definitely done by other thieves.

The police have the following information at their disposal: The master thief is known to only travel during the night, she always drives, and she always stays in a city during the day. Certain cities are sufficiently far apart so there is no way to drive from one to the other in a single night. Furthermore, on certain days, the police carefully searched various cities, and on certain nights, roadblocks were set up between pairs of cities. While the master thief was never found by these means, the police are confident that their attempts were thorough. That is, if they searched a city on a particular day, she couldn’t have been there on that day. Similarly, if they set up roadblocks between two cities on a particular night, she didn’t travel between those cities on that night.

The police give you all the information they have. In particular, they provide a list of the \( n \) relevant cities, and a list of which pairs of cities are within driving distance of each other. For each of the \( m \) days since the original crime, they list which cities had thefts, and which cities were searched. And for each of the \( m - 1 \) nights between these \( m \) days, they list where roadblocks were set up (assume that if a roadblock is set up between two cities, there is no other way of traveling between them).

Provide an efficient algorithm that determines from the given information which of the thefts could *not* have been carried out by the master thief.

**Hint:** Construct a directed graph that captures the known information and use a basic graph algorithm.