CS41 Homework 12

This homework is due at 11:59PM on Friday, December 15. Note the unusual due date. This is a 4-point homework. Write your solution using LaTeX. Submit this homework using **github** as **.tex** file. This is a **partnered homework**. You should primarily be discussing problems with your homework partner.

It's ok to discuss approaches at a high level with others. However, you should not reveal specific details of a solution, nor should you show your written solution to anyone else. The only exception to this rule is work you've done with a lab teammate *while in lab*. In this case, note (in your **homework submission poll**) who you've worked with and what parts were solved during lab.

1. **Hospitals coping with natural disaster.** (K&T 7.9)

   The same hospitals from earlier in the semester have now hired all the doctors they need. There is a widespread natural disaster, and a lot of people across an entire region need to be rushed to emergency medical care. Each person should be brought to a hospital no more than 50 miles away from their current location. Additionally, we want to make sure that no single hospital is overloaded, so we want to spread the patients across the available hospitals. There are n people who need medical care and h hospitals; we want to find a way to coordinate emergency medical evacuations so that each hospital ends up with at most $\lceil n/h \rceil$ patients in emergency care. (Also, obviously: every patient should end up at a hospital!)

   Give a polynomial-time algorithm that takes the given information about patients' locations and hospitals and determines whether this is possible. If it is possible, your algorithm should also output an assignment of patients to hospitals ensuring that every patient gets to a nearby hospital and that no hospital is overloaded. Your algorithm should be a reduction to network flow.

   Prove that your algorithm is correct. You may assume that you have a polynomial-time black box implementation which solves the network flow problem.[1]

[1]Note that our original Ford-Fulkerson algorithm wasn't quite polynomial time. However, it is possible to improve the algorithm to be polynomial time by specifying how it selects an augmenting path at the beginning of each iteration. We didn't analyze this in lecture, but you can assume it exists and use it as a black box.