CIS 502 - Algorithms

Fall 2011 Homework 0

Due Mon September 12th in class or at Levine 502. The grades of this homework will not count. A regular homework will be 4 such problems. The goal of this homework is calibration on your part in terms of effort and direction needed to solve problems in algorithms. Late submissions are not relevant.

Remember, the answer(s) have be to typed up, you can use any software, and you need not copy the question(s).

Problem 1 *Question 6, pg 25–26.*

(We are reproducing the problem in case the students do not yet have textbooks.) We will not be reproducing the problems for any future homework.

Peripatetic Shipping Lines, Inc., is a shipping company that owns n ships and provides services to n ports. Each of its ship has a schedule that says, for each day of the month, which of the ports it's currently visiting, or whether it's out at sea. (You can assume that the month here has m days, for some $m > n$.) Each ship visits each port for exactly one day during the month. For safety reasons, PSL Inc., has the following strict requirement: *Not two ships can be in the same port on the same day.*

The company wants to perform maintenance on all the ships this month, via the following scheme. They want to *truncate* each ship's schedule: for each ship S_i, there will be some day when it arrives in its scheduled port and simply remain there for the rest of the month. (for maintenance). This means that S_i will not visit the remaining ports on it's schedule. (if any) that month, bu this is okay. So the *truncation* of S_i's schedule will simply consist of its original schedule up to a certain specified day on which it is in port P; and the remainder of the truncated schedule simply has it remain in port P.

Now the company's question to you is the following: Given the schedule for each ship, find a truncation of each such that the condition (safety requirement mentioned above) continues to hold: no two ships are ever in the same port on the same day.

Show that a set of truncations can always be found, and give an algorithm to find them.