Time Spent: 2 hours

Collaborators and Resources:

Problem 1

To summarize the problem given in K-T 1.6 we can say that we are given n ships and n ports. Each ship has a schedule that says (for each day of the month) which port it is visiting or whether it is out at sea. The month must have m(>n) days.
Now, the truncation of S_i's schedule consists of its original schedule up to a certain specified day, on and after which it remains in a port P.
(1) Every ship can visit all ports on their schedule(†) until docking at port P. That is, one-day visits to all ports for the rest of the truncated schedule.
(2) Each port does maintenance on (exactly one unique) ship. That is, the company wants to make sure they perform maintenance on all ships.
(3) Each day, no port has two ships at once(†) continues to hold.
An example of a shipping schedule as well as my modified Gale-Shapley algorithm (K-T p.6) are given below ...

A. Given an untruncated shipping schedule of Ships and Ports
B. While a ship is unassigned and hasn't contacted every port

1. Choose such a ship s Let p be the last port in s's schedule which s has not contacted
2. If p is not taken, match (s,p)
3. If p is already matched to s'
   (a) If s' visits p before (in the untruncated schedule) s does, then s remains unassigned
   (b) Else #they don't visit simultaneously per rules
      i. match (s,p). s' becomes unassigned.

C. Return the set of pairings.

Claim 1. The algorithm returns valid truncated schedules for any given valid shipping schedule.

Proof. The G-S algorithm makes sure that there aren't any instabilities(KT 1.6).
An instability in this case would involve two pairs (s, p) and (s', p') in a perfect matching S wherein s prefers p' to p, and p' prefers s to s'.
Ships are going to prefer to dock for maintenance as late as possible, first so they can visit as many ports as possible and more importantly so that they do not obstruct other ships' visits.
Now, we're going to define preference for ports as follows: ports prefer ships which will visit them as soon as possible. We define it this way because we know that the G-S algorithm returns a stable matching in which each port is paired with its "worst" valid ship(KT 1.8), which means the port is visited as late as possible.
So in an instability, s is docked sooner than necessary at p, and in the case that s' needs to visit p before it reaches p' we may run into conflicts.
Since a given shipping schedule can be translated into a stable matching problem (where ships prefer the latest ports, and ports prefer the earliest ships), we can use this algorithm (which is a translation of Gale-Shapley) in order to ensure that all ships can have the least amount of truncation. For example,
Given the following valid shipping schedule:

Ship 1	at sea	port P_1	port P_3	at sea	port P_2	port P_4
Ship 2	port P_3	port P_2	at sea	port P_1	port P_4	at sea
Ship 3	port P_4	at sea	port P_2	port P_3	port P_1	at sea
Ship 4	port P_1	port P_3	at sea	port P_4	at sea	port P_2

we can generate the following preference lists,
Ship 1: (P_4, P_2, P_3, P_1) Port 1: (S_4, S_1, S_2, S_3)
Ship 2: (P_4, P_1, P_2, P_3) Port 2: (S_2, S_3, S_1, S_4)
Ship 3: (P_1, P_3, P_2, P_4) Port 3: (S_2, S_4, S_1, S_3)
Ship 4: (P_2, P_4, P_3, P_1) Port 4: $(S_3, S_4, S_2, S_1$
and my algorithm (which is analogous to Gale-Shapley given these "preferences") will return the matchings
(Ship 1, Port 2), (Ship 2, Port 1), (Ship 3, Port 3) and (Ship 4, Port 4).

□