CMPS 102 — Spring 2018 – Homework 2

Each problem is worth 25 points. For each problem you must specify your algorithm clearly and prove it correct.

Reminder: For each problem you will receive 6 of the 25 points if you write **nothing**. If you feel that you have a partial answer worth more than 6 points, e.g., you have an algorithm but can't prove it correct, or your algorithm can be proven correct but is too slow, you can still write this information down, as long as you begin your answer by assessing the state of your work. If we find your work as assessed, you will receive at least 6 points.

1 Star Wars

In a galaxy far, far away, the Empire has built n Death Satellites to attack n planets controlled by the rebels (each satellite will attack exactly one planet). The Rebels have installed a defensive shield in each planet. The i-th satellite has power f_i, while the shield on planet j can tolerate up to s_j power. A successful attack on the i-th planet brings benefit b_i to the Empire. That is, if planet i is attacked by the j-th satellite and $f_j > s_i$, then the Empire will receive benefit b_i; otherwise, it will receive benefit 0. Match satellites to planets to maximize the Empire's benefit in $O(n \log n)$ time, using a greedy algorithm. **Hint:** Make one or more heaps.

2 Super Slam

You are a wrestling promoter putting together a SuperSlam and you need to pair up your fighters by weight, in order to have the most balanced fights. Specifically, there are two pools of fighters, $A = \{a_1, a_2, \ldots, a_n\}$ and $B = \{b_1, b_2, \ldots, b_n\}$ and each fighter from A must be paired with exactly one fighter from B. Your job is to minimize the **sum** of $|w(a_i) - w(b_j)|$ over all matched pairs, where $w(f)$ denotes the weight of fighter f. Being an entrepreneurial person, you are naturally inclined to design a greedy algorithm for this task. Do it.

Bonus (10 points): What if the notion of imbalance changed from $|w(a_i) - w(b_j)|$ to $[w(a_i) - w(b_j)]^2$? Would your algorithm still work? If not, could you modify it so that it does?

3 You Can't Pause Toast

Leon needs to make breakfast toast for the n members of his family. His toaster can accommodate one person's toast at a time, and each family member, j, wants their toast to spend t_j seconds in the toaster. In Leon's household, certain family members must have their toast before others (the precedence relationships form a directed acyclic graph). Violating precedence constraints is **not** an option, as it will make everything topsy turvy. Also, the amount of complaining that will be voiced by family member, j, if they receive their toast at time t is $c_j(t)$, where c_j is a member-specific, non-negative, strictly increasing function.

One thing Leon has learned is that you cannot pause toast. Pausing toast causes it to lose its essence. So, once Leon starts toasting someone's toast, he cannot stop until it is done. Now, truth be told, Leon does *not* want to bring the ruckus to this breakfast. He asks you to develop a greedy algorithm that finds the ordering (schedule) that minimizes the **maximum** complaining among all valid toast schedules. **Hint: start at the end**

4 Dock of the Bay

You are in charge of the port of Oakland, where n ships are present. Ship i holds c_i containers and must depart the port by time t_i. Unloading a ship takes one hour, independently of how many containers are on it. Only a single ship may be unloaded at any given moment. The unloading of any ship can start as early as time 0. Develop a greedy algorithm to select which ships to unload so as to maximize the *total number of containers* unloaded.

Hint: make one safe decision after another