CS41 Homework 1

This homework is due 10AM Thursday January 29. Write your solution using LaTeX. Submit this homework using handin41. This is an individual homework. It’s ok to discuss approaches at a high level. In fact, I encourage you to discuss general strategies. 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 partner while in lab. In this case, note who you’ve worked with and what parts were solved during lab.

1. **Algorithm Analysis.** Consider the following algorithm for the Hiking Problem.

   ```plaintext
   Algorithm 4()
   1     \( k = 2 \).
   2     while you haven’t met your friend
   3         walk \( k \) miles north.
   4     return to start.
   5     walk \( k \) miles south.
   6     return to start.
   7     \( k = k^2 \).
   ```

   Describe the distance travelled in Algorithm 4 as a function of the initial distance of your friend in the worst case. Express your answer in big-Oh notation. How does this algorithm compare to the algorithms we saw in class?

2. **Stable Matching Runtime.** We showed in class (at least after Friday’s class) that the Gale-Shapely Algorithm for stable matching terminates after at most \( n^2 \) iterations of the while loop.

   (a) For two sets \( A \) and \( B \) of size \( n \), can a particular list of rankings actually result in a quadratic number of iterations? If so, describe what the rankings would look like. If not, argue why no set of rankings would ever result in a quadratic number of iterations. **Note:** the algorithm need not take exactly \( n^2 \) iterations, but asymptotically \( n^2 \) iterations, meaning \( 0.1n^2 \) would be sufficient to show your claim.

   (b) Can a particular set of rankings result in strictly less than a quadratic number of iterations? Can you design an input that requires \( O(n) \) iterations? If so, describe the structure of this input. If not, argue why this is not possible.

   (c) Finally, can you design an input that takes fewer than \( n \) iterations? Why or why not?

   Aim for clarity and conciseness in your write up of this problem. You should have all the necessary tools to express your solutions. You do not need formal proofs or psuedocode, but you should be able to clearly articulate your ideas in plain English.

3. **Minor League Baseball.** In the past ten years, there has been a renaissance in minor league baseball. Teams have become very successful via clever marketing and creating unique mascots, such as IronPigs, Isotopes, Sand Gnats, etc.
A group of $n$ baseball fans $F = \{f_1, \ldots, f_n\}$ are touring $n$ minor league baseball stadiums $S = \{s_1, \ldots, s_n\}$, each in search of a new favorite team over the course of a season (of $m \geq n$ days). Each fan $f_i$ chooses an itinerary where he/she decides to visit one baseball stadium per day. When a fan decides on a team to root for, he/she will stay at that stadium for the rest of the season. However, these fans are pretentious and fiercely independent, preferring not to be fans of the same team. In fact, no two of these fans want to be at the same stadium on the same day.

Show that it is possible to assign each fan $f$ a unique stadium $s_f$, such that when $f$ arrives at $s_f$ according to the itinerary for $f$, all other fans $f'$ have either stopped touring baseball stadiums themselves, or $f'$ will not visit $s_f$ after $f$ arrives at $s_f$. Describe an algorithm to find this matching.

**Hint:** The input is somewhat like the input to stable matching, but at least one piece is missing. Find a clever way to construct the missing piece(s), run stable matching, and show that the final result solves the baseball fan problem. You can assume that the itineraries of the fans are such that no two fans will plan on touring the same stadium on the same day. (Of course, if one fan decides to stay at a stadium, there might be a problem.)

4. **Same-Sex Stable Marriage (Extra Credit).** In class, we discussed a version of the Stable Matching problem where we want to match $n$ men to $n$ women. In this problem, we discuss the single-sex version. The input is a set of people $A = \{p_1, \ldots, p_{2n}\}$ of size $2n$. Each person ranks the others in order of preference. A matching $M = \{(i, j)\}$ is instable if there exists $(i, j), (i', j') \in M$ such that $i$ prefers $j'$ to $j$ and that $j'$ prefers $i$ to $i'$. A matching is stable if it is perfect and there are no instabilities.

(a) Does a same-sex stable matching always exist? Prove that such a matching must always exist, or give an example where no stable matching occurs.

(b) Design and analyze an efficient algorithm that either returns a same-sex stable matching or outputs that no such matching exists.