CS41 Homework 1

This homework is due at 11:59PM on Wednesday, September 7. Write your solution using LaTeX. Submit this homework using **github**. This is an individual homework. It's ok to discuss approaches at a high level. In fact, we 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 (in your **README file**) who you've worked with and what parts were solved during lab.

When you submit homework assignments this semester, please keep the following in mind:

1. Don't forget to fill out the README.md file.
2. Don't include your name in hw1.tex. (I'd like the graders to *not* know who you are, to minimize grader bias)
3. Don't submit a .pdf – just the .tex will do.
4. Graders will compile your code from the .tex file using pdflatex. It is your responsiblity to make sure the LaTeX compiles.

The main **learning goals** of this homework assignment are to familiarize you with the tools/process we use for assignments and to begin to formalize and analyze algorithms.

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

   ```
   Hiking()
   k = 1.
   while you haven't arrived at your friend:
       hike k miles north
       return to start
       hike k miles south
       return to start
       k = 5k.
   ```

   Describe the distance traveled in Hiking as a function of the initial distance from 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. **Algorithm Design.** Choose a problem you encounter in everyday life (e.g. how to get from your dorm room to Sharples by 8:30AM, or how to get into college) and describe an algorithm for solving that problem.

   Be as specific and descriptive as you can.

3. A horde of seven barbarians has recently returned from pillaging the countryside, looting a hoard of 49 gold coins in the process. Now, they would like to divide their treasure. Being the enlightened barbarians they are, they have decided to vote on how to best divide the treasure.

The barbarians' voting process is as follows. First, the strongest barbarian proposes a scheme for dividing the treasure. For example, she might propose to keep 31 gold coins for herself and give the six remaining barbarians 3 coins each. However, being the barbarians they are, if the majority vote against this scheme, then the rest of the barbarians kill the strongest. In this case, the strongest of the remaining six barbarians proposing a way to divide the treasure, risking death if more than half of the barbarians vote against him.

The process repeats in a similar fashion (strongest remaining barbarian proposes a way to divide the hoard, barbarians vote, and the strongest remaining barbarian dies if they reject his suggestion) until a division is accepted.

How should the strongest barbarian divide the hoard? You can assume that all barbarians are greedy and care only about how much treasure they receive.

Note: If there are an even number of barbarians and a vote that is evenly split, the tie goes to the proposing barbarian (the other half of the barbarians are weaker and don't have the strength to kill the strongest).

4. **(extra credit problem)** We discussed in lecture a reason why m is a lower bound for the Hiking Problem. Show that $3m$ is a lower bound for the Hiking Problem.

5. **(extra credit problem)** In lab we argued that updating $k \leftarrow 2k$ is more efficient than $k \leftarrow k + 1$. However, why stop there? Would it be more efficient to increase k even more rapidly? Consider the following algorithm for the Hiking Problem.

ExtremeHiking()

$k = 2$.
while you haven't arrived at your friend:
    hike k miles north
    return to start
    hike k miles south
    return to start
    $k = k^2$.

Again, describe the distance traveled in Hiking as a function of the initial distance from 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?