CS41 Homework 1

This homework is due at 11:59PM on Monday, September 6. This is a 10-point homework. 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, you are encouraged 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 post-homework survey) who you've worked with and what parts were solved during lab.

The main **learning goals** of this lab are to (i) familiarize you with writing in LaTeX, and (ii) to begin to formalize and analyze algorithms.

1. **EdStem.** Log onto the course forum on EdStem, and either ask a question, or respond to an existing post. Don't feel like your question/post has to be about computer science! The goal is just to make sure you're comfortable using the forum.

2. **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 = 6k.
```

   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-O notation. How does this algorithm compare to the algorithms we saw in class and lab?

3. **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.

4. **(extra challenge problem)** We discussed in lab 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 challenge 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-O notation. How does this algorithm compare to the algorithms we saw in class?