CS41 Homework 3

This homework is due 11:59PM on Wednesday, September 27. Write your solution using \LaTeX. Submit this homework using github. Try to make your answers as clear and concise as possible. This is an individual homework. It’s ok to discuss approaches and strategies at a high level. 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.

0. Before final submission, make sure to fill out the README file.

1. **Cycle detection.** (Kleinberg and Tardos, 3.2)
   
   Give an algorithm to detect whether a given undirected graph contains a cycle. If the graph contains a cycle, then your algorithm should output one. Your algorithm should run in $O(m + n)$ time for a graph with $n$ vertices and $m$ edges.

2. **Network fragility.** (Kleinberg and Tardos, 3.9)

   There’s a natural intuition that two nodes that are far apart in a communication network—separated by many hops—have a more tenuous connection than two nodes that are close together. There are a number of algorithmic results that are based to some extent on different ways of making this notion precise. One way involves the susceptibility of paths to the deletion of nodes. 

   Suppose that an $n$-node graph $G = (V, E)$ contains two nodes $s$ and $t$ such that the distance$^1$ between $s$ and $t$ is strictly greater than $n/2$. Show that there must exist some node $v$ not equal to either $s$ or $t$, such that deleting $v$ from $G$ destroys all $s \rightsquigarrow t$ paths. (In other words, the graph obtained from $G$ by deleting $v$ contains no $s \rightsquigarrow t$ paths).

   Give an algorithm with running time $O(n + m)$ to find such a node $v$.

3. **Rumor spreading.**

   Hipsters don’t like sharing coffee shops, but they do sometimes congregate online. After arriving at their assigned (distinct) coffee shops, the same group of $n$ hipsters (from week 1) go online. One of them, $h_1$, wants to start a rumor that their coffee shop is full of cats, fixed-gear bicycles, and the best coffee on Earth, and is thus the best coffee shop, but $h_1$ wants to be sure that every other hipster will hear the rumor. Hipsters always repeat rumors to their friends, but not all hipsters are friends with all other hipsters.

   If it takes one minute to repeat the rumor (copy-paste, plus time to pick an emoji), design and analyze an algorithm which hipster $h_1$ can use to figure out whether every other hipster $h_2, h_3, \ldots, h_n$ will hear the rumor (and if they do, then the algorithm should report how long it will take until everyone has heard the rumor).

4. *(Extra Credit.)* This week, we saw an algorithm for testing bipartiteness which used BFS to color the vertices one of two colors. For a positive integer $k$, call a graph $k$-colorable if the vertices can be properly colored using $k$ colors. In other words, a bipartite graph is

---

$^1$Formally, define the *distance* between $s$ and $t$ as the length of the shortest path between $s$ and $t$. 

1
two-colorable. In this problem, you will investigate algorithms dealing with three-colorable graphs.

- Design and analyze an algorithm which takes as input a graph $G = (V, E)$ and returns \textsc{yes} if $G$ is three-colorable, and \textsc{no} otherwise.
- Design and analyze an efficient algorithm which takes as input a \textit{three-colorable} graph $G = (V, E)$ and colors the vertices of the graph using $O(\sqrt{n})$ colors. (Note: while the input graph is three-colorable, it does not mean that we know what that coloring is!)