CS41 Homework 4

This homework is due at 11:59PM on Sunday, September 30. Write your solution using $\LaTeX$. Submit this homework using **github** as a file called **hw4.tex**. This is a **partnered homework**. You should primarily be discussing problems with your homework partner.

It's ok to discuss approaches at a high level with others. 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. **Asymptotic Analysis.** Let $f(n) = n^2 \log(n) + 17n - 4$ and $g(n) = \frac{n^{2.5}}{2} - n$. Prove that $f(n) = O(g(n))$. You may use techniques and facts from class and the textbook. Your proof should be complete and formal.

2. **Cycle Detection (K&T 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 runtime should be $O(m + n)$ for a graph with m edges and n vertices.

   Hint: Don't forget edge cases. Don't forget to return the cycle if one is detected.

3. **(K&T 3.9).** There's a natural intuition that two nodes are far apart in a communication network—separated by many hops—have a more tenuous connection than two nodes that are closer together. There are a number of algorithmic results that are based to some extent on different ways of making this notion precise. Here's one that involves the susceptiblility of paths to the deletion of nodes.

   Suppose that an n-node undirected graph $G = (V, E)$ contains two nodes s and t such that the distance 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* $s - t$ paths. Give an algorithm with running time $O(m+n)$ to find such a node v. You do not need a full proof of correctness, but should provide a convincing argument on why your algorithm works.

4. **Social Distance.** In analyzing social networks, sociologists are often interested in measuring the social distance between individuals or groups. The social distance between two people $d(v_1, v_2)$ can be defined as the number of edges along the shortest path between them. However, there are several ways to measure the social distance between groups.

   Design and analyze a polynomial-time algorithm that takes an undirected social network graph $G = (V, E)$, along with two (possibly overlapping) groups $V_1 \subset V$ and $V_2 \subset V$ and returns the *minimum distance* $D(V_1, V_2) := \min_{v_1 \in V_1, v_2 \in V_2} d(v_1, v_2)$.

   Give high-level pseudocode for your algorithm. You do not need a formal runtime analysis, but it should be clear that your algorithm runs in polynomial time, either directly from your pseudocode, or from a followup explanation.

5. **(Extra Challenge).** 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 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 YES if G is three-colorable, and NO otherwise.
- Design and analyze an efficient algorithm which takes as input a *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!)