Module 2 - Graphs: Advanced (Written Problems)

The purpose of this homework is to practice solving new problems related to Graphs on your own. Each of the problems below will ask you to develop / describe an algorithm, reason about / prove something, or reason about runtimes. For algorithms, your descriptions should be detailed enough that someone could produce code from your description. Proofs should be logically sound and pithy, but do not need to be fully formalized. Runtimes need to be provided and argued succinctly. Good luck!

1. Describe an algorithm that, given a directed graph G represented as an *adjacency matrix*, returns whether or not the graph contains vertex with in-degree $|V| - 1$ and out-degree 0. In other words, does the graph have a node such that every other node points to it, but it does not point to any other node. Your algorithm must be $O(V)$. Note that there are $\Theta(V^2)$ cells in your adjacency matrix so you'll need to be clever here.

2. Write clear pseudo-code to solve the following:

   given a graph G, a start vertex s, and a vertex node t, use *DFS* to find any path from s to t and return the list of vertices in that path. Your algorithm should stop the search as soon as it finds any path. If t is not reachable from s, return an empty path (i.e., an empty list). The vertices in the list that is returned should be in order from s to t. G could be directed or undirected. For this problem, please use an implementation of the search algorithm taught in class and modify it. *Note: You are going to need this exact algorithm on a future programming assignment, so take the time to think about it carefully here.*

3. Let G be an undirected graph with n nodes (let's assume n is even for simplicity). Prove or provide a counterexample for the following claim: If every node of G has a degree of at least $\frac{n}{2}$, then G must be connected. *Note: G cannot have any loops (edges from a node to itself).*

4. This problem is about robots that need to reach a particular destination. Suppose that you have an area represented by a graph $G = (V, E)$ and two robots with starting nodes $s_1, s_2 \in V$. Each robot also has a destination node $d_1, d_2 \in V$. Your task is to design a schedule of movements along edges in G that move both robots to their respective destination nodes. You have the following constraints:

   - You must design a schedule for the robots. A schedule is a list of steps, where each step is an instruction for a single robot to move along a single edge.
   - If the two robots ever get close, then they will interfere with one another (perhaps start an epic robot fight?). Thus, you must design a schedule so that the robots, at no point in time, exist on the same or adjacent nodes.

- You can assume that s_1 and s_2 are not the same or adjacent, and that the same is true for d_1 and d_2.

Your solution must contain the following items:

1. Describe an algorithm that produces an optimal schedule for the two robots.
2. What is the runtime of your algorithm?
3. How would the runtime change as the number of robots grows?