Problem 1. Answer the following questions briefly:

1. Consider an undirected graph with $n \geq 2$ vertices. What are the minimum and maximum number of different levels that the graph could have, respectively?

   Solution: 2 and n respectively.

2. Consider an undirected graph with n vertices and m edges. What are the minimum and maximum number of connected components that the graph could have?

   Solution: 1 and n respectively.

3. What is the runtime of depth first search algorithm as a function of the number of vertices, n and edges, m, if the input graph is represented by an adjacency matrix?

   Solution: $O(n^2)$

4. How many edges are there in an undirected graph with 20 vertices each of degree six? Justify.

   Solution: 60
   Since there are 20 vertices each with a degree of 6, therefore $\sum_{v \in V} deg(v) = 120$
   For an undirected graph $\sum_{v \in V} deg(v) = 2|E|$
   Therefore, the number of edges is $2|E| = 120 \Rightarrow |E| = 60$

Problem 2. A cynosure among a group of n people is a person who knows nobody but is known to everybody else. The goal is to identify a cynosure by only asking questions to people of the form: "Do you know him/her?"

1. Model this relationship with a directed graph that is represented with an adjacency matrix. Answer the following questions:

   (a) Who are the vertices of this graph?

   Solution: Vertices are the people in the group.

   (b) What is the meaning of a directed edge in this graph?

   Solution: A directed edge means the person represented by the tail of the arrow knows the person represented by the head of the arrow.

(c) What is a cynosure in terms of this graph?

> **Solution:** A cynosure is an edge with in-degree $n - 1$ and an out-degree zero.

(d) What is the equivalent in graph terms to the defined question?

> **Solution:** Does an edge, $v_i \rightarrow v_j$, between vertices i and j exist.

(e) What are we trying to minimize? (hint: What is the equivalent of runtime?)

> **Solution:** the number of "questions asked".

(f) Can a group have more than one cynosure?

> **Solution:** No.

2. Write pseudo-code for an efficient algorithm to identify a cynosure or determine that the group has no such person. How many exact number of questions does this algorithm need in the worst-case?

Solution:

```
findCynosure(W):
    # Anyone or no one could be a cynosure
    for i = 1 to n:
        A[i] = 0
    # cindex is a cynosure
    cindex = 1
    A[cindex] = 1
    for i = 2 to n:
        # If cindex knows i, then i is the new cynosure.
        if W[cindex,i] == 1:
            A[i] = 1;
            A[cindex] = 0;
            cindex = i
    # Check that everyone knows the cynosure.
    for i = 1 to n:
        if A[i] != 1 and W[i,cindex] == 1:
            continue;
        else:
            cindex = -1
    # Check that the cynosure does not know anyone.
    for i = 1 to n:
        if i != cindex and W[cindex,i] == 1:
```

```
                cindex = -1
        return cindex
```