COMP1201 Assignment 3

**Q1:**

Prove that in any Facebook community, there exists two people who have the same number of friends.

Assumptions:

1) There are *n* people in the community.

2) A person *cannot* be friends with themselves.

3) Assuming that there *cannot* be a person with *0* friends. That means that the maximum number of friends a person can have is *n-1.* In this case, because friendship is symmetric, everybody else must at least know this person, and the minimum number of people a person can know is *1.* This gives us the set *{1, 2, ..., n-1}* which represents the possible number of people each person can know.

4) Assuming that there *can* be a person with *0* friends. This means that the maximum number of friends a person can have is *n-2*. Following the logic from *assumption 3*, this gives us the set *{0, 1, …., n-2}*.

Both of the sets, described in *assumption 3* & *assumption 4,* have *n-1* elements. These elements describe all the *possibilities* for the number of people each person can have as a friend.

Every single person must be assigned one of these *n-1* possible numbers. But since there are *n* people (*assumption 1*), one of these numbers must be used twice due to the pigeonhole principle.

This proves that there are at least two people that have the same number of friends.

Example:

There are 20 people (*n = 20*). This means that there are 19 (*n-1*) possibilities of how many friends each person can have.

⌈20 / 19⌉ = 2

We can see that there are at least two people with the same amount of friends.

Prove that any simple graph *G* with at least two vertices must contain two vertices of the same degree.

Assumptions:

1) A *simple* graph is unweighted and undirected. It cannot contain graph loops or multiple edges.

2) Since we have a *simple* graph, there can be *no edges*.

3) There are *n* vertices in a graph.

4) Since we can have a vertex with a degree of *0* (*assumption 2*), this gives us the set *{0, 1, …., n-2}* which represents the possible degrees a vertex can have.

Because there are *n-1* elements in the set (*assumption 4*) of possible degrees, and every vertex (of which there are *n*) must have a degree, using the pigeonhole principle, we can see that there are at least two vertices must have the same degree.

Example:

There are 4 vertices (*n = 4*). This means that there are 3 (*n-1*) possibilities of what degree each vertex can be.

⌈4 / 3⌉ = 2

We can see that there are at least two vertices with the same degree.

**Q2:**

**(A)** Consider a minimum spanning tree (MST) of a connected, weighted graph. If we remove an edge (u, v) of the MST, then we get two separate trees.

Example MST:

1

2

3

8

4

6

4

5

6

7

3

4

Are these two trees the MSTs on their respective sets of nodes?

Is the edge (u, v) a least-weight edge crossing between those two sets of nodes?

1) We cannot remove edges (6, 7), (4, 5) or (2, 3) because if we do remove them, we will not be left with trees (as for a set of vertices to be considered a tree, there must be at least 2 vertices).

Note: If we do remove the edges mentioned above, we can still get the MST of the graph by rerunning the MST finding algorithm.

2) If we remove an edge, which is not listed in 1) and is not the least-weight edge crossing between the two sets of nodes (edge (1, 2)), we get two trees, which *are* the MSTs on their respective sets of nodes.

Example: After removing edge (4, 7)

1

2

3

8

4

6

4

5

6

7

3

We have two trees, which are the MSTs on their respective sets of nodes.

3) If we delete the least-weight edge crossing between the two sets of nodes (edge (1, 2)) we still get two trees, which are the MSTs on their respective sets of nodes.

2

3

8

4

6

4

5

6

7

3

4

**(B)** Consider the following algorithm for finding a MST on a graph:

- Split the nodes of the graph arbitrarily into two nearly equal-sized sets.

- Find a MST on each of those sets.

- Connect the two MSTs with the least-cost edge between them.

Would this algorithm always return an MST of the original graph?