SamLy_HW02

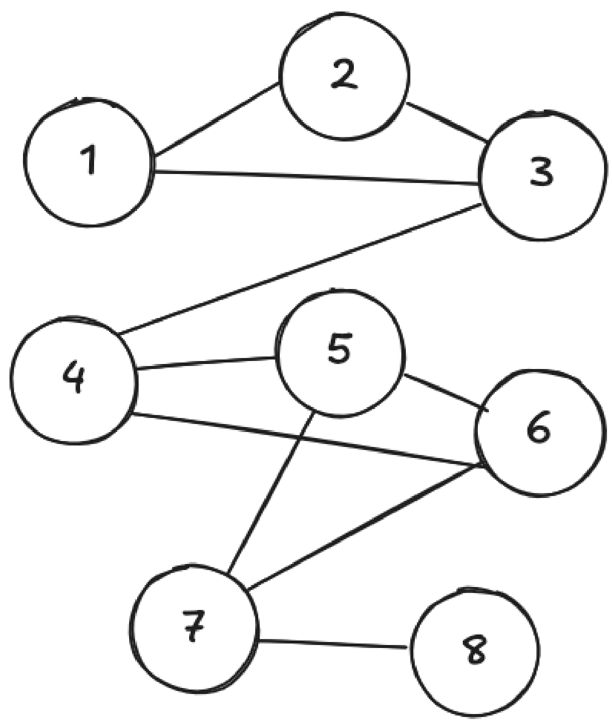
September 28, 2025

1 Part A: By Hand (15 points)

Edges: (1, 2), (1, 3), (2, 3), (3, 4), (4, 5), (4, 6), (5, 6), (5, 7), (6, 7), (7, 8)

Nodes: {1, 2, 3, 4, 5, 6, 7, 8}

No. Edges: 10



1.1 Density (3 points)

1.1.1 Write the formula for density.

$$D = \frac{2E}{N(N-1)}$$

1.1.2 Count nodes and edges.

8 nodes, 10 edges.

1.1.3 Compute the density of this graph.

$$D = \frac{2(10)}{8(7)} = 0.357$$

1.2 Local Clustering Coefficient for Node 3 (3 points)

1.2.1 Write the formula.

$$C_i = \frac{2E_i}{k_i(k_i - 1)}$$

1.2.2 Identify Node 3's neighbors.

Node 3's neighbors are node 1, node 2, and node 4.

1.2.3 Count edges among them.

There is one edge between node 3's neighbors: (1,2).

1.2.4 Compute C_3

$$C_3 = \frac{2(1)}{3(2)} = \frac{1}{3}$$

1.3 Global Clustering Coefficient (3 points)

1.3.1 Compute the local custering coefficient for each node with degree $\geq$ 2.

Node i	k_i	E_i	C_i
1	2	1	1
2	2	1	1
3	3	1	1/3
4	3	1	1/3
5	3	2	2/3
6	3	2	2/3
7	2	1	1
8	1	-	-

1.3.2 Average them.

$$C = \frac{1}{7}\sum_{i=1}^{7} C_i = \frac{5}{7}$$

1.4 Average Path Length (4 points)

1.4.1 List all unique pairs of nodes.

Starting at Node:

1. (1,2), (1,3), (1,4), (1,5), (1,6), (1,7), (1,8)

2. (2,3), (2,4), (2,5), (2,6), (2,7), (2,8)
3. (3,4), (3,5), (3,6), (3,7), (3,8)
4. (4,5), (4,6), (4,7), (4,8)
5. (5,6), (5,7), (5,8)
6. (6,7), (6,8)
7. (7,8)

1.4.2 Find the shortest distance $d(i, j)$ for each pair.

	1	2	3	4	5	6	7	8
1		1	1	2	3	3	4	5
2			1	2	3	3	4	5
3				1	2	2	3	4
4					1	1	2	3
5						1	1	2
6							1	2
7								1
8								

1.4.3 Compute the average

Average = 2.29

1.5 Communities (by eye) (2 points)

1.5.1 Identify groups of nodes you believe form "communities."

Groups {1, 2, 3} and {4, 5, 6, 7, 8} seem to be communities. This is because {1 2, 3} form a triangle, while {4, 5, 6} and {5, 6, 7} also form triangles. 8 is only connected to 7, so it is part of the community that 7 is in.

2 Part B: Coding (10 points)

[ ]:
```python
import networkx as nx

# 1. Graph representation (2 points)
edges = [
    (1, 2), (1, 3), (2, 3), (3, 4), (4, 5),
    (4, 6), (5, 6), (5, 7), (6, 7), (7, 8)
]
nodes = list(range(1, 9))

graph = nx.Graph()
graph.add_nodes_from(nodes)
graph.add_edges_from(edges)

# 2. Density function (2 points)
print("density", nx.density(graph))

# 3. Clustering coefficients (2 points)
print("clustering coef", nx.clustering(graph, 3))

# 4. Average path length (2 points)
print("avg. path length", nx.average_shortest_path_length(graph))

# 5. Modularity (2 points)
print("modularity", nx.algorithms.community.modularity(
    graph, [{1,2,3}, {4,5,6,7,8}]))
```

```
density 0.35714285714285715
clustering coef 0.3333333333333333
avg. path length 2.2857142857142856
modularity 0.355
```

3 Part C: Reflection (5 points)

The results from the code seems to match up with our hand-calculated results. The communities we identified by eye seems to have a relatively strong modularity score when tested. A value of 0.355 suggests that our initial selection of communities was solid. If this were an online community, the modularity score tells us that participants 1, 2, 3 and 4, 5, 6, 7, 8 form tight sub-communities. They are connected only via the edge between 3 and 4.