1 Shortest Paths and Cycles

a) Load the graph `small_graph.glickle`

b) How many nodes and edges does the graph have?

c) How many cycles does the cycle basis of the graph contain? How Many edges does the longest cycle in the cycle basis have?

d) Create a small graph using `G.add_nodes_from()` and `G.add_edges_from()` containing about 5-10 nodes and edges. Check if the graph has a planar embedding using the following check for planarity:

```python
#function checks if graph G has K(5) or K(3,3) as minors,
#returns True /False on planarity
def is_planar(G):
    result=True
    n=len(G.nodes())

    if n > 5:
        for subnodes in it.combinations(G.nodes(),6):
            subG=G.subgraph(subnodes)
            # check if the graph G has a subgraph K(3,3)
            if bipartite.is_bipartite(G):
                X, Y = bipartite.sets(G)
                if len(X)==3:
                    result=False

    if n > 4 and result:
        for subnodes in it.combinations(G.nodes(),5):
            subG=G.subgraph(subnodes)
            # check if the graph G has a subgraph K(5)
            if len(subG.edges())==10:
                result=False

    return result
```

e) Select two (random) nodes in the graph and calculate the length of the shortest path between them.

f) What is the greatest distance between any pair of vertices? (Longest shortest path/diameter)

g) Select one node in the graph. Create and plot a histogram of the shortest paths from this node to every other node.

2 Edge and Node Attributes

a) Which node/edge attributes does the graph have?

b) Using the node attributes calculate the total length of the graph.

c) Using the lengths calculated in c) create a new edge attribute called "length" for each edge. Calculate the length of the graph again using the new edge attribute.

d) Create and plot a histogram of edge lengths.