Experimenting with Community Structure

1. Community strucutures with Gephi
   (a) Load your favorite graph in Gephi.
   (b) In the *statistics* panel on the right, use the `Modularity` button to compute a partition. (Note than you have an option to ignore weights, and an option to change the resolution parameter).
   (c) Change the colors of nodes according to communities. Observe how nodes that are closed in the visualization tends to have similar colors, although those two properties(color, position) are yielded by unrelated processes.
   (d) Change the resolution parameter and observe the differences.

2. Community Structure with networkx

   To detect communities, you can use the `cdlib` package. It also contains functions for evaluation and comparison of partitions. For details, check the documentation at `https://cdlib.readthedocs.io/en/latest/` .

   (a) Using `networkx`, load the airport dataset on the class webpage.
   (b) Using `cdlib`, detect communities on this network using the louvain method. You have to use the `algorithms.louvain` method. (You can use a different algorithm just by calling a different function)
   (c) Add the communities found on your graph as a node property, using `NodeCluster.to_node_community_map()` and `nx.set_node_attributes` .
   (d) Visualize the communities found. In order to interpret them, you should draw each node at its geographical location, with a color per community. You can check properties existing for a node with g.nodes()['node_name'], and get attributes of a node with `nx.get_node_attributes("attribute_name")`

   There are several ways to draw a spatial network with colors corresponding to communities, the first one, ignoring edges, is to use a simple `scatterplot` . The second is to use the `plot_network_clusters` function of `cdlib` , but you will need to set a custom color palette to see a large number of clusters. Finally, another way is to export the graph in graphml format and to use Gephi to visualize it. You'll need the Gephi plug-in called *Geo Layout*, and choose an **Equirectangular** Projection with the proper **scale**. You can generate your own palette with as many colors as required, from the interface. Although indirect, it is probably the most flexible solution, since it allows to see node labels and to easily change node visualization parameters. You could also use `folium` library to obtain an interactive visualization on a map.

3. Comparing Partitions
   (a) The provided airport data also contains information about the country of each airport, which can be interpreted as a *ground truth* partition of the network. Transform this information into a `NodeClustering` object of `cdlib` (`NodeClustering(partition,graph,"GroundTruth")`).
   (b) Compute the AMI or NMI between the community structure and the partition in countries
   (c) By exploring systematically the values of the resolution parameter for modularity, find the partition with the highest similarity to the partition in countries.

(d) Compare visually the results, and try to interpret it. Is the partition in country a meaningful topological partition, i.e., is studying this network by considering that nodes in a same country form a coherent/homogeneous group a good approach? (yes and no, probably...)