ASSIGNMENT 1

Social network of TV shows actors

Goal:

- ***develop a Python-application that builds a social network and computes a graph of relations between actors from TV shows.***
  - ✓ *the nodes of the graph will be the actors,*
  - ✓ *and the edges indicate that one work with another in the same TV show,*
  - ✓ *the edges will carry a weight representing the number of times the actors have worked together.*

Given a corpus of data, you will extract who has worked with whom, and whenever a connection is found, an edge is added to the graph of a social network, or an existing edge is strengthened.

1. Extract data about the first **8** TV shows from JSON-file "*tvshows.json*"
2. Create the class *Actor* to store data about an actor of TV show: *actorID, actorName, actorBirthday, actorCharacterName, actorShows*.
   actorShows - is a dictionary of show's IDs (where the actor participated). Initially – empty. It stores TV show data as follows: keys are show's IDs, values – TV shows dates (year of premiere).
   Actor class has a method to fill the *actorShows* dictionary.
3. Create the class *ActorGraph* to store and maintain a dictionary of actors (*Actor*), these are the nodes of our graph.
4. *Actor* will in turn maintain a list of shows. *Actor* will also maintain a dictionary in which the keys are other *Actors* (*actorIDs*) instances, and the values indicate the weight of the relationship. This thus makes up the edges of our graph *ActorGraph*.
5. The weight of the relationship between actors is calculated as follows:
   If actors participated in the same show the default weight of the relation is 1. If actors were together in more than one TV show the weight increases appropriately (+1).
6. Design appropriate methos to calculate the "influence" of each actor.
   "Influence" is the average of relationships weights.
7. Build the graph of actors and their relationships.
8. Visualize "Influences" of the top 5 the most "influential" actors by using Seaborn bar plot.
9. Visualize the graph of actors and their relationships by using NetworkX.
10. Customize the first version of the graph visualization by adding colors for nodes, colors and widths for edges according to "influence" of actors, and weights of relationships.