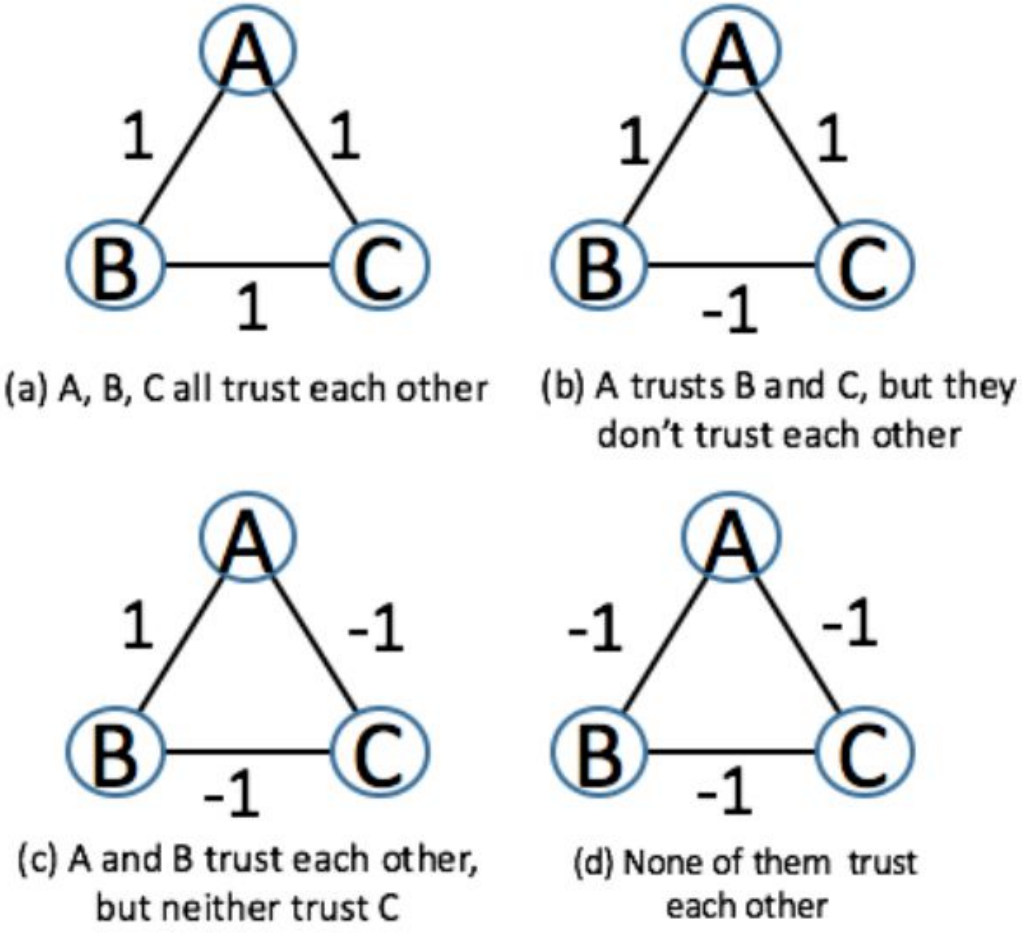
Approach:-
As initially started with the assignment I focused on the graph and assumed that it could be done by straight away parsing the csv file but it didn't look like that happening because i had to understand the fact that you can actually get the number of edges by going through csv file itself but capturing the triads would be difficult and literally more difficult if networkx is not used.

Like the Triads being shown in here:



So later I figured out that by using networkx you can simply convert the csv into the graph and then moved forward with creating the graph and then forming a list of nodes with weights given. Once the whole graph is traversed, I am calculating the expected and actual probabilities the way it's given in the assignment

File Structure:

I have three files in here,
\epinion.py \printing.py \formGraph.py \requirements.txt (if needed)

My main function and iteration is espion.py. Run this file and you would be able to see the magic.

I have tried to mention comments everywhere I could but may have missed few spots

References, which I have taken:
For displaying and calculating in DataFrame
https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.sort_values.html

I also took the broad picture from here, but you can see my code is completely different:
https://stackoverflow.com/questions/56537560/how-to-efficiently-calculate-triad-census-in-undirected-graph-in-python

You will notice that the actual distribution of triad types differs from the expected distribution based on random assignment. Discuss briefly how they differ and why that might be so.

Answer:- Well, I have measured the difference and it's because of the graph is not completely connected and there are few broken edges/ nodes are not connected, which basically causes the actual to be different from expected but I couldn't get into the depth of that because of the large file of csv, and debugging it becomes more difficult.