Statement of Problem 1:

A group of ethnographers analyze some oral history data they've collected by interviewing members of a village to learn about the lives of the people who've lived there over the past few hundred years. Every bit of data provided by the villagers comes in one of two forms:

- For some i and j, person P_i died before person P_j was born
- For some i and j, the life spans of P_i and P_j overlapped

Efficiently decide if the data provided is consistent.

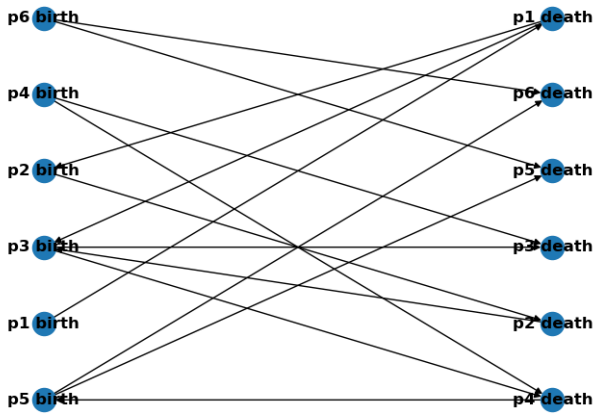
Solution
We can solve this using a topological sort on a directed graph. In this directed graph, all persons $P_1, P_2, ..., P_n$ will be the vertices. For each person, one vertex will represent their "birth" and another will represent their "death". In the case of the first type of data, we will draw a directed edge from the death vertex of P_i to the birth vertex of P_j to indicate that time period ordering goes from the death of P_i to the birth of P_j. In the case of the second type of data, we will draw two edges. One edge will be drawn from the birth of P_i tho the death of P_j and a second edge will be drawn from the birth of P_j tho the death of P_i to indicate that they lived in the same time period. Once we have finished drawing all the directed edges, we will conduct a topological sort on the data to order it from earliest to latest. If during our ordering we cannot find some vertex v such that the in-degree of v is zero, that must mean that our data is inconsistent and we can let those ethnographers know that they wasted a lot of time...

Our inability to find some vertex v such that its in-degree is zero suggests that their is a cycle in our graph. A cycle indicates that there is some conflict in the ordering i.e. P_a lived before P_b, P_b lived before P_c, and P_c lived before P_a.

Given some input data which looks like the following:

```
p1 b p2
p2 b p3
p1 b p3
p3 s p4
p4 b p5
p5 s p6
```

'b' indicates before, and 's' indicates overlapping. This will produce the following bipartite graph:



Topologically sorting the data proves it is consistent by providing the following possible sequence of events:
p6 born → p4 born → p1 born → p1 died → p2 born → p2 died → p3 born → p4 died → p5 born → p6 died → p5 died → p3 died

Let's formally prove that this method works for determining data consistency.

Proof
We can break this proof into cases.

Case 1: The data was consistent but the algorithm returned inconsistent
If the algorithm returns that the data is inconsistent, then the data created cannot be topologically sorted and there must exist some cycle in the graph. However, edges are only drawn between vertices if the vertex with the out-degree came before the vertex with the in-degree. For births p_{1_b}, p_{2_b}, p_{3_b}, and deaths p_{1_d}, p_{2_d}, p_{3_d}: A cycle can occurs if there are edges drawn from p_{1_d} to p_{2_b}, p_{2_d} to p_{3_b}, and p_{3_d} to p_{1_b}. This implies that p_1, p_2 and p_3 all came before each other which is impossible. Of course, there could be overlapping persons present within the cycle, but this still produces the inconsistency outlined. Therefore, the data is inconsistent.

Case 2: The data was inconsistent but the algorithm return a consistent ordering
The algorithm returning a consistent ordering could only occur if there was no cycle in the generated graph. This can only occur if among all the "facts" there was $n \geq 2$ people who claim to live before each other in a cyclical way. If two or more people stated to have lived before each other when such a statement is not possible, then because there are already edges between each persons birth and their death, the added edges which are drawn from death to birth would have caused a cycle in the graph. If there is a cycle in the graph, a topological sort is not possible. Therefore, the data must have been consistent. □

Statement of Problem 2:

Given a set of unsigned paintings created by two different artists whom we will call X and Y, researchers are trying to figure out which artist painted each one. They don't have any data on which artist painted each painting, but they do have data of the following form:

- For some i and j, the same artist created paintings p_i and p_j
- For some i and j, different artists created paintings p_i and p_j

We can't reason about which artist painted each painting from this data, but we can figure out if it is consistent.

Solution
Because we are reasoning about the connections between objects, it's helpful to think about this problem as a graph where vertices represent paintings and edges represent the connections between them. We cannot be sure if the data is complete, meaning any 3 vertices in the generated graph form a triangle. Let's think about it in terms of its components.

Claim 1: Any component of the graph which consists of a lone vertex cannot contain an inconsistency.

Proof
This is trivially proven by the fact that for there to be an inconsistency, either (a) or (b) must be true. Because there are no connections to this lone vertex, it can be either an X or Y without causing an inconsistency. □

Claim 1 is important because it allows us to effectively ignore any paintings which may exist but do not show up in the input to the program. Let's also make **Claim 2**: Any component which is a tree cannot contain an inconsistency.

Proof
We can prove this by induction. Let n be the number of vertices.

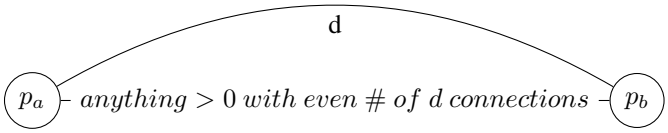
Base case $n = 2$. In this case if the connection between the two vertices is s, then make them both X or both Y. If the connection between them is d, then make one X and one Y.

Inductive step: Let's assume that for all $2 < n < k$ that our data is consistent and try to prove the same for k. Well, because the component must be a tree, adding one more vertex and edge to our $k - 1$ size graph cannot create a cycle. Therefore, we must be adding a vertex v onto a leaf in the tree which we will call u. Well, if the connection uv is s, then simply make v whatever the value of u is. This will not cause an inconsistency because there are no other connections to v than the edge uv. Similarly, if the connection uv is d, then simply make v the opposite value of u which will also be consistent by our previous logic.

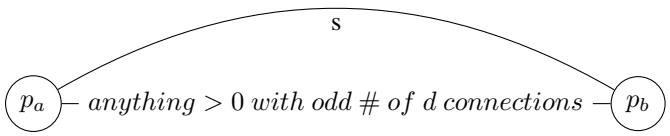
Because our claim holds for some arbitrary k, our claim holds in general. □

By Claims 1 and 2, we can conclude that only elements which lie on a cycle can have an inconsistency. We can succinctly identify which cycles will force our algorithm to return $false$:

(c)



(d)



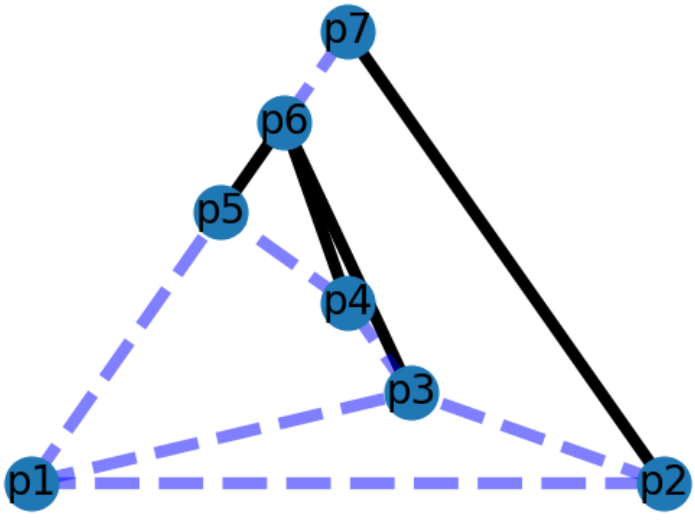
Proof
Observe that case (c) will cause an inconsistency because as long as the amount of d connections between p_a and p_b are even (which includes zero), p_a and p_b will be the same but there is a d connection between them. Similarly, case (d) will cause an inconsistency because an odd number of d connections between p_a and p_b will cause them to be different, but there is an s connection between them. □

Using Claims 1 and 2, as well as our observations about which cycles will cause an inconsistency, we can create an algorithm which will successfully perform the required task.

Step 1: Create a graph from all input data to the program. If two different connections for the same two vertices appear, then return false.
Step 2: Pick some arbitrary vertex v and label it X. Then, perform a DFS labeling vertices along the way based on the connection between them.
Step 2a: If we reach a vertex which we've already seen before, then we've identified a cycle. If the connection between the two vertices agrees with their respective labeling, then we can continue the algorithm. If not, return false.
Step 2b: Once we finish the search. We must make sure we have visited all vertices. If not, then pick some arbitrary vertex which is unvisted and go to Step 2.
Step 3: If all vertices have been visited, then our data must be consistent and we can return true.

Depth First Search will work wonderfully to identify inconsistencies. Because we are only checking that there are no logical conflicts in the data (and not which artist painted each painting), picking a random vertex and marking it X will not impede our ability to find conflicts. In other words, every vertex is either X or Y and we only need to be careful when we come across some vertex that has already been labeled. Given the following in put data where 's' indicates the same artist painted the two paintings and 'd' indicates the opposite, we can generate a graph. Dashed lines in the graph represent that the two adjacent paintings are the same while solid lines indicate they are different.

```
p1 s p2
p2 s p3
p1 s p3
p3 s p4
p4 s p5
p5 d p6
p6 d p4
p5 s p1
p3 d p6
p6 s p7
p7 d p2
```



Our algorithm determines the data is consistent and provides the two sets of paintings where each set is painted by one painter.

Painter 1 created = $\{p3, p2, p1, p5, p4\}$
Painter 2 created = $\{p6, p7\}$

Conclusion
I've shown two interesting uses of graph theory in data consistency problems and proved their correctness. All code for the project can be found here. Running requires the networkx and matplotlib libraries.