This project analyzes the graph data from the data set of butterfly similarity network using depth first search and then topological sorting. This data set is found on Standford datasets:http://snap.stanford.edu/biodata/datasets/10029/10029-SS-Butterfly.html. The dataset contains two files. One file includes nodes and edges, and the second file includes the nodes, edges, and weights of each edge. The nodes represent organisms of butterflies while the edges represent visual similarities between the organisms. I found this data set to be very interesting as I was researching the applications of graph algorithms in different fields. I wanted to check the graph data for if it was cyclic or acyclic so that I could do topological sorting or implement Dijkstra's algorithm. However, I found that there was a debugging issue and I could not figure it out in time. I left my code for Dijkstra's algorithm in the comments so I could debug it and test it out if I had time. I found that my output repeated the nodes in ascending chronological order after looking for strongly connected components. I first reversed the edges of the graph and checked for unvisited nodes and visited nodes using a stack. After going through all the nodes, I found that there were two components, but the majority of the nodes were under the second component compared to the first. I tried testing for the closeness centrality of the graph, however, I had difficulty debugging my code. My graph was directed and weighted, so I could implement shortest-path algorithms on my data. I wanted to try applying Dijkstra's algorithm and the closeness centrality algorithm. Since the edges in my data had positive weights, I could also implement Dijkstra's algorithm. I could then check for each node and calculate the distance from all the other nodes. This would allow me to calculate the closeness centrality score using the shortest paths between all nodes and the sum of its distance compared to all the other nodes. This was not a success due to how much time I had, but I split my code into modules. I had one for depth-first search and one for my graph structure. This project helped me explore the applications of depth-first search and how different algorithms can only be used on graphs with certain properties. I learned a lot about the types of centrality algorithms and how shortest paths can be applied.