General Input: graph, which is the given graph instance
start_person, which in the graph is a node that represents the start person
end_person, which in the graph is a node that represents the end person
parents, which is a map of the form output by bfs(graph, start_person) that maps nodes in given graph as keys to their own parent nodes as values

General Output: path, which consists of a sequence of tuples containing two elements, one element is the actor while another element is a set of attribute between two nodes

Algorithum: find_path, which is used to find the path from start_person to end_person in the graph
bfs, which is formely defined function that is used to performs a breadth-first search on given graph starting at the given node as the starting node

1. create a new sequence referred by the variable actor_movie
2. create a new sequence referred by the variable path
3. check whether given start_person equlas to end_person
   - If the output is true(which means start person and end person are the same person):
     - Translate path from a sequence to a set. Then translate the set into sequence and translate start_person to a sequence. Connect two sequences into one sequence by adding them directly. Use the variable actor_movie to refer to this sequence

       Translate the sequence referred by actor_movie into a tuple. Use the variable actor_movie to refer to this tuple.

       Add the tuple referred by actor_movie into the sequence path

       Return path
   - If the output is not true, check whether the value of the key end_person in the map parents equals to None(which means the end person doesn't have parent nodes):
     - If the output is true:
       - Return the empty sequence path
     - If the output is false:
       - Translate path from a sequence to a set. Then translate the set into sequence

and translate end_person to a sequence. Connect two sequences into one sequence by adding them directly. Use the variable actor_movie to refer to this sequence

- Translate the variable actor_movie to a tuple. Add the tuple referred by actor_movie into the sequence path
- While end_person doesn't equal to start_person:

    - Get the attribute between the end_person and its parent node whose value is the value of the key end_person in the map named parents. Assign the value of attribute to the variable attribute.
    - Translate attribute from a set to a sequence. Translate the parent node of the variable end_person to a sequence. Connect two sequences into one sequence by adding them directly. Use the variable actor_movie to refer to this sequence
    - Translate the variable actor_movie to a tuple. Add the tuple referred by actor_movie into the sequence path
    - Assign the value of the parent node of the variable end_person to the variable end_person.

4. Reverse the order of the elements inside the sequence path

5. Return the sequence path.

Discussion

1. By calling the function bfs (graph, start_node), we can get one map which maps the distance from the start node to each node in the graph. Therefore we can choose any node we want as the starting node. Then we can compare distances between that node and the other nodes. The maximum distance in the map is the value of the diameter.
2. We can call the function distance_histogram(graph, node) to show the frequency of the distance between the node and the other nodes. According to the returned map, we can count the time of the situation that distance equals to one. This shows how many other nodes are directly connected to that node. We then do the same thing for the other nodes. Then we compare all the frequency and choose the one with maximum value. The node which has the most time of situation that the distance equals to one is the hub.
3. The criteria can be the total number of time that this movie is used as the path to connect two actors. The larger number means the movie is used for more times to connect two actors. In another word, this movie is more important.
4. The difference is the type of the distance and the kinds of distance and the frequency of each corresponding distance. In Kevin Bacon plot, the distance's type is decimal and he has six kinds of distance while in Stephanie Fratus plot the distance's type is an integer and has four kinds of different distance.
   Since the frequency of the situation that distance=1 in Kevin Bacon plot is larger than that in

Stephanie Fratus plot, therefore Kevin Bacon is more directly connected with other actors. In another word, he is more like a hub. However, since Stephanie Fratus has 6 kinds of distance while Kevin Bacon has 4, this means Kevin Bacon can have relationship with any actors within four degrees while Stephanie Fratus can have relationship with any actors within six degrees. In another word, if we suppose there is a center point directly connecting to any other points, Kevin Bacon is more close to this center point while Stephanie Fratus is more far away.