Problem 1[40 pts]
Notations:
Red: won't broadcast to neighbors of red node since they have been reached and
their neighbors will receive the message.
underscore: source node
highlight: The path
These notation is only done for first and second hop, just for you to get an idea, its
too many in the third and forth hop
In the following part, the style is “start, neighbor, count(excluding the source)”

1)[10 pts] Supernode graph: **111 messages**
   **First hop:** total message sent 1
   5 [2] 1
   **Second hop:** total message sent 33
   2 [0, 1, 3, 4, 5, 9, 12, 24, 25, 27, 31, 34, 38, 44, 49, 52, 57, 58, 59, 60, 62, 63, 64, 67, 68, 69, 74, 75, 76, 79, 88, 89, 93, 98] 33
   **Third hop:** total message sent 77
   0 [1, 2, 3, 4, 70, 7, 40, 73, 66, 13, 78, 77, 16, 17, 83, 97, 22, 87, 36, 90] 19
   1 [0, 2, 3, 4, 8, 10, 11, 14, 15, 18, 19, 26, 41, 42, 46, 47, 48, 50, 53, 54, 55, 61, 65, 72, 91, 94, 95, 96] 27
   3 [0, 1, 2, 99, 4, 33, 43, 45, 80, 82, 21, 56, 39, 29] 13
   4 [0, 1, 2, 3, 37, 6, 32, 28, 71, 81, 35, 20, 85, 86, 23, 84, 92, 30, 51] 18

2)[10pts] Random graph: **291 messages**
   **First hop:** total message sent 6
   5 [0, 1, 8, 74, 15, 19] 6
   **Second hop:** total message sent 56
   0 [96, 2, 3, 4, 5, 70, 73, 75, 45, 11, 81, 82, 30, 18, 26, 27, 62, 63] 17
   1 [2, 3, 5, 6, 9, 13, 16, 22, 24, 25, 26, 28, 31, 36, 40, 42, 44, 48, 53, 56, 57, 58, 78, 85] 23
   8 [97, 4, 5, 47, 49, 20, 25, 57, 37] 8
   74[32, 5] 1
   15[20, 5, 7] 2
   19[64, 99, 4, 5, 21, 93] 5
   **Third hop:** total message sent 160
   Notation “[nodes to receive information], count(excluding the source)”
Forth hop: total message sent 69

[64, 2, 4, 11, 34, 15, 29] 6
[83, 19, 7] 2
[9, 19] 1
[2, 19] 1
[32, 19] 1
Part b) [10 pts]

For average number of hops, you may take a bunch of two random nodes get the shortest path between them.

Then get the average of that.

Avg = 2.6826 for superpeer network
Sd = 0.5845

Avg = 2.9717 for random network
Sd = 0.8478

Code: (get standard deviation in the same manner)
for line in open(infile):
    G = json_graph.loads(line)
count = 0
lent = 0
for node in range(0, 100):
    for node1 in range(0, 100):
        if node != node1:
            lent += len(nx.shortest_path(G, node, node1)) - 1
    count += 1
print lent / count

Answer is flexible for these two questions [10 pts]

• What conclusions can you draw about the two network topologies?
• Finally, can you suggest some benefits for choosing one topology over the other, even if it has a larger average hop count?

Problem 2: [30 pts]
a) I guess everyone can do it.
b) Successor 92, Predecessor 33
Chord requires each node to keep a "finger table" containing up to $m$ entries. The $i^{th}$ entry of node $n$ will contain the address of successor $(n + 2^{i-1}) \mod 2^m)$.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>92</td>
</tr>
<tr>
<td>1</td>
<td>92</td>
</tr>
<tr>
<td>2</td>
<td>92</td>
</tr>
<tr>
<td>3</td>
<td>101</td>
</tr>
<tr>
<td>4</td>
<td>135</td>
</tr>
<tr>
<td>5</td>
<td>135</td>
</tr>
<tr>
<td>6</td>
<td>160</td>
</tr>
<tr>
<td>7</td>
<td>10</td>
</tr>
</tbody>
</table>

c) 10
   33

d) forth entry in the table changes to 135

Problem 3 Rubrics [30pts]

Report (pdf) with screenshots: 5 pts!

registry server starts up: 3 pts!

two peers join: 8 pts!

• After connection has been built, the server should have recorded each connected peer's address information, no matter how server collected: 3 pts!

a 3rd peer joins and requests the list of online peers: 3 pts!

the 3rd peer requests a connection to the 2nd peer; they begin to chat: 5 pts!
If A sends a message to B, B should be able to receive it and reply to A immediately: 3 pts!

We don’t require A should record B’s address, though it is preferred!
the 1st peer quits: 3 pts!

finally, the registry server shuts down while two peers are chatting; chat should not be impacted: 3 pts