Assignment2 (Score: 9.0 / 11.0)

1. Test cell (Score: 0.0 / 0.0)
2. Test cell (Score: 0.0 / 1.0)
3. Test cell (Score: 1.0 / 1.0)
4. Test cell (Score: 1.0 / 1.0)
5. Test cell (Score: 1.0 / 1.0)
6. Test cell (Score: 0.0 / 0.0)
7. Test cell (Score: 1.0 / 1.0)
8. Test cell (Score: 1.0 / 1.0)
9. Test cell (Score: 1.0 / 1.0)
10. Test cell (Score: 1.0 / 1.0)
11. Test cell (Score: 1.0 / 1.0)
12. Test cell (Score: 0.0 / 1.0)
13. Test cell (Score: 0.0 / 0.0)
14. Test cell (Score: 1.0 / 1.0)

Assignment 2 - Network Connectivity

In this assignment you will go through the process of importing and analyzing an internal email communication network between employees of a mid-sized manufacturing company. Each node represents an employee and each directed edge between two nodes represents an individual email. The left node represents the sender and the right node represents the recipient. We will also store the timestamp of each email.

In [1]:
```
import networkx as nx

#!head assets/email_network.txt
```

In [2]:
```
def plot_graph(G, weight_name=None):
    '''
    G: a networkx G
    weight_name: name of the attribute for plotting edge weights (if
G is weighted)
    '''
    %matplotlib notebook
    import matplotlib.pyplot as plt

    plt.figure()
    pos = nx.spring_layout(G)
    edges = G.edges()
    weights = None
#     print(edges)
    if weight_name:
        weights = [int(G[u][v][weight_name]) for u,v in edges]
        labels = nx.get_edge_attributes(G,weight_name)
        nx.draw_networkx_edge_labels(G,pos,edge_labels=labels)
        nx.draw_networkx(G, pos, edges=edges, width=weights);
    else:
        nx.draw_networkx(G, pos, edgelist=edges);
```

Question 1

Using networkx, load up the directed multigraph from `assets/email_network.txt` . Make sure the node names are strings.

This function should return a directed multigraph networkx graph.

In [3]:
Student's answer (Top)

```
def answer_one():
    # Your Code Here
    return nx.read_edgelist(path='assets/email_network.txt', data=
[('time', int)], create_using=nx.MultiDiGraph()) # Your Answer Here
answer_one()
```

Out[3]:
```
<networkx.classes.multidigraph.MultiDiGraph at 0x7fedacd6ea30>
```

In [4]:
Grade cell: `cell-483a56c89f9db231` Score: 0.0 / 0.0 (Top)

```
ans_one = answer_one()
```

Question 2

How many employees are represented in the network?

How many `sender` -> `recepient` pairs of employees are there in the network such that `sender` sent at least one email to `recepient` ? Note that even if a `sender` sent multiple messages to a `recepient` , they should only be counted once. You should exclude cases where an employee sent emails to themselves from this count.

This function should return a tuple with two integers (#employees, # `sender` -> `recepient` pairs).

In [5]:

Student's answer (Top)

```
def answer_two():

    # Your Code Here
    numEmails = len(answer_one().edges())
    numEmployees = len(answer_one().nodes())
    return numEmployees, numEmails # Your Answer Here
answer_two()
```

Out[5]: (167, 82927)

In [6]:

Grade cell: `cell-5b6391549b076d2b` Score: 0.0 / 1.0 (Top)

```
ans_two = answer_two()
```

```
You have failed this test due to an error. The traceback has been remo
ved because it may contain hidden tests. This is the exception that wa
s thrown:

AssertionError: Incorrect number of sender/recepient counts
```

Question 3

- Part 1. Assume that information in this company can only be exchanged through email.

  When an employee sends an email to another employee, a communication channel has been created, allowing the sender to provide information to the reciever, but not viceversa.

  Based on the emails sent in the data, is it possible for information to go from every employee to every other employee?

- Part 2. Now assume that a communication channel established by an email allows information to be exchanged both ways.

  Based on the emails sent in the data, is it possible for information to go from every employee to every other employee?

This function should return a tuple of bools (part1, part2).

In [7]:

Student's answer (Top)

```
def answer_three():

    # Your Code Here
    # Part 1
    strong = nx.is_strongly_connected(answer_one())
    # Part 2
    weak = nx.is_weakly_connected(answer_one())
    return strong, weak # Your Answer Here
answer_three()
```

Out[7]: (False, True)

In [8]:

Grade cell: cell-82b3f0bc45e2895f Score: 1.0 / 1.0 (Top)

```
ans_three = answer_three()
```

Question 4

How many nodes are in the largest weakly connected component of the graph?

This function should return an int.

In [9]:
Student's answer (Top)

```
def answer_four():

    # Your Code Here
    maxNumWeak = max(nx.weakly_connected_components(answer_one()), key=len)
    return len(maxNumWeak) # Your Answer Here
answer_four()
```

Out[9]: 167

In [10]:
Grade cell: cell-2b1b7b06ecfa751d Score: 1.0 / 1.0 (Top)

```
ans_four = answer_four()
```

Question 5

How many nodes are in the largest strongly connected component?

This function should return an int

In [11]:
Student's answer (Top)

```
def answer_five():

    # Your Code Here
    maxNumStrong = max(nx.strongly_connected_components(answer_one()), key=len)
    return len(maxNumStrong) # Your Answer Here
answer_five()
```

Out[11]: 126

In [12]:
Grade cell: cell-b0524f7dc1fbdec4 Score: 1.0 / 1.0 (Top)

```
ans_five = answer_five()
```

Question 6

Using the NetworkX functions `strongly_connected_components` and `subgraph` , find the subgraph of nodes in the largest strongly connected component. Call this graph G_sc.

This function should return a networkx MultiDiGraph named G_sc.

In [13]:

Student's answer (Top)

```
def answer_six():

    # Your Code Here
    g = max(nx.strongly_connected_components(answer_one()), key=len)
    G_sc = answer_one().subgraph(g)
    return G_sc # Your Answer Here
answer_six()
```

Out[13]: <networkx.classes.multidigraph.MultiDiGraph at 0x7fedbc38ff10>

In [14]:

Grade cell: `cell-cf148ef273b3b19c` Score: 0.0 / 0.0 (Top)

```
ans_six = answer_six()
assert type(ans_six) == nx.MultiDiGraph , "Your return type should b
e a MultiDiGraph object"
```

Question 7

What is the average distance between nodes in G_sc?

This function should return a float.

In [15]:

Student's answer (Top)

```
G_sc = answer_six()
def answer_seven():

    # Your Code Here

    return nx.average_shortest_path_length(answer_six()) # Your Answ
er Here
answer_seven()
```

Out[15]: 1.6461587301587302

In [16]:

Grade cell: `cell-5b374fdd48f37e02` Score: 1.0 / 1.0 (Top)

```
ans_seven = answer_seven()
```

Question 8

What is the largest possible distance between two employees in G_sc?

This function should return an int.

In [17]:

Student's answer (Top)

```
def answer_eight():

    # Your Code Here
#     ecc = nx.eccentricity(G_sc)
#     int(max(ecc))
    return nx.diameter(G_sc) # Your Answer Here
answer_eight()
```

Out[17]: 3

In [18]:

Grade cell: `cell-c5714787854ef644` Score: 1.0 / 1.0 (Top)

```
ans_eight = answer_eight()
```

Question 9

What is the set of nodes in G_sc with eccentricity equal to the diameter?

This function should return a set of the node(s).

In [19]:
Student's answer (Top)

```
def answer_nine():

    # Your Code Here
#     d = nx.diameter(G_sc)
#     e = nx.eccentricity(G_sc)
#     n = [node for node in e.items() if node[1] == d]
#     set([node[0] for node in n])
    return set(nx.periphery(G_sc)) # Your Answer Here
answer_nine()
```

Out[19]: {'129', '134', '97'}

In [20]:
Grade cell: cell-77c9ca0b94df3d6f Score: 1.0 / 1.0 (Top)

```
ans_nine = answer_nine()
assert type(ans_nine) == set, "Student answer must return a set"
```

Question 10

What is the set of node(s) in G_sc with eccentricity equal to the radius?

This function should return a set of the node(s).

In [21]:
Student's answer (Top)

```
def answer_ten():

    # Your Code Here
#     r = nx.radius(G_sc)
#     e = nx.eccentricity(G_sc)
#     n = [node for node in e.items() if node[1] == r]
#     set([node[0] for node in n])
    return set(nx.center(G_sc)) # Your Answer Here
answer_ten()
```

Out[21]: {'38'}

In [22]:
Grade cell: cell-bfd2ee304bc25264 Score: 1.0 / 1.0 (Top)

```
ans_ten = answer_ten()
assert type(ans_ten) == set, "Student answer must return a set"
```

Question 11

Which node in G_sc has the most shortest paths to other nodes whose distance equal the diameter of G_sc?

For the node with the most such shortest paths, how many of these paths are there?

This function should return a tuple (name of node, number of paths).

In [23]:

Student's answer (Top)

```
def answer_eleven():

    # Your Code Here
    # We need to use only the nodes in the periphery
    # The eccentricity is the maximum distance from one node to all
other nodes in G (returns an dict)
    # ecc = nx.eccentricity(G_sc)
    # The periphery is the set of nodes with eccentricity equal to t
he diameter.
    peri = nx.periphery(G_sc)
    # The diameter is the maximum eccentricity
    diam = nx.diameter(G_sc)

    numPathsDiam = {}

    # AQUI FICOU FODA!
    for node in peri:
        sp = nx.shortest_path(G=G_sc, source=node)
        pathsLenghtDiam = [path for path in sp.values() if (len(pat
h) -1 == diam)]
        numPathsDiam[node] = len(pathsLenghtDiam)
    # FIM AQUI FICOU FODA!

    keys = list(numPathsDiam.keys())
    values = list(numPathsDiam.values())
    resultKey = keys[values.index(max(values))]

    return resultKey, numPathsDiam[resultKey] # == ('97', 63) # Your
Answer Here
answer_eleven()
```

Out[23]: ('97', 63)

In [24]:

Grade cell: cell-f79b06650f61cf37 Score: 1.0 / 1.0 (Top)

```
ans_eleven = answer_eleven()
assert type(ans_eleven) == tuple, "Student answer must be a tuple"
```

Question 12

Suppose you want to prevent communication flow from the node that you found in question 11 to node 10. What is the smallest number of nodes you would need to remove from the graph (you're not allowed to remove the node from the previous question or 10)?

This function should return an integer.

In [25]:

Student's answer (Top)

```
def answer_twelve():

    # Your Code Here
    n = answer_eleven()[0]
    c = nx.center(G_sc)[0]
#     cut = nx.minimum_node_cut(G=G_sc, s=c, t=n)
    conn = nx.node_connectivity(G_sc, s=c, t=n) - 1
    return conn # Your Answer Here
answer_twelve()
```

Out[25]: 5

In [26]:

Grade cell: `cell-509cfa9f4136124d` Score: 0.0 / 1.0 (Top)

```
ans_twelve = answer_twelve()
```

```
You have failed this test due to an error. The traceback has been remo
ved because it may contain hidden tests. This is the exception that wa
s thrown:

AssertionError: Incorrect number of nodes
```

Question 13

Convert the graph G_sc into an undirected graph by removing the direction of the edges of G_sc. Call the new graph G_un.

This function should return a networkx Graph.

In [27]:
Student's answer (Top)

```
def answer_thirteen():
    un = G_sc.to_undirected()
    return nx.Graph(un) # Your Answer Here
answer_thirteen()
```

Out[27]:
```
<networkx.classes.graph.Graph at 0x7fed9086e6a0>
```

In [28]:
Grade cell: cell-d1c0627a327cd774 Score: 0.0 / 0.0 (Top)

```
ans_thirteen = answer_thirteen()
assert type(ans_thirteen) == nx.Graph , "Your return type should be
a Graph object"
```

Question 14

What is the transitivity and average clustering coefficient of graph G_un?

This function should return a tuple (transitivity, avg clutering).
Note: DO NOT round up your answer.

In [29]:
Student's answer (Top)

```
def answer_fourteen():
    g = answer_thirteen()
    return nx.transitivity(g), nx.average_clustering(g) # Your Answe
r Here
answer_fourteen()
```

Out[29]:
```
(0.570111160700385, 0.6975272437231418)
```

In [30]:
Grade cell: cell-41dda91202f58e7e Score: 1.0 / 1.0 (Top)

```
ans_fourteen = answer_fourteen()
assert type(ans_fourteen) == tuple, "Student answer must be a tuple"
```

This assignment was graded by mooc_adswpy:63f4b23a9e38, v1.25.120622