Lab 7. Calculating centrality measures and creating an interest graph for a Github user

In this lab, you need to calculate the centrality measures of a graph and also create an interest graph of a Github user

1. Calculate the degree, betweenness, and closeness centrality measures of a krackhardt kite graph

In [1]:

```
from operator import itemgetter
from IPython.display import HTML
from IPython.core.display import display
import networkx as nx
```

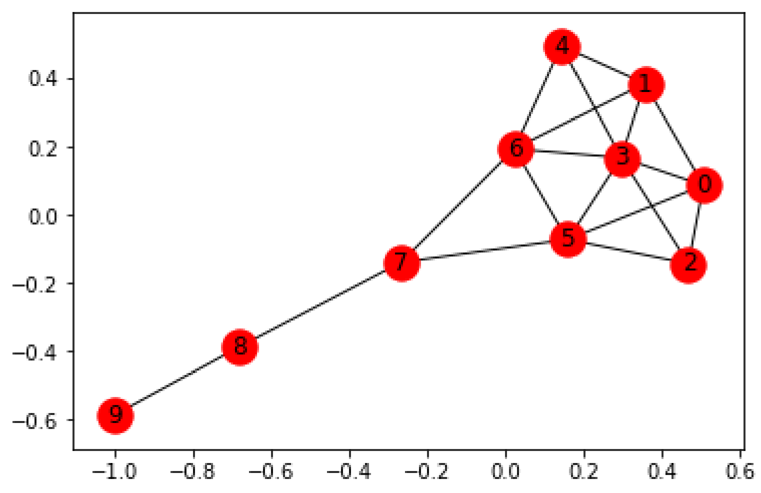
In [2]:

```
# The classic Krackhardt kite graph
kkg = nx.generators.small.krackhardt_kite_graph()

print ("Degree Centrality")
print (sorted(nx.degree_centrality(kkg).items(),
             key=itemgetter(1), reverse=True))
print

print ("Betweenness Centrality")
print (sorted(nx.betweenness_centrality(kkg).items(),
             key=itemgetter(1), reverse=True))
print

print ("Closeness Centrality")
print (sorted(nx.closeness_centrality(kkg).items(),
             key=itemgetter(1), reverse=True))
```

```
Degree Centrality
[(3, 0.6666666666666666), (5, 0.5555555555555556), (6, 0.5555555555555556),
(0, 0.4444444444444444), (1, 0.4444444444444444), (2, 0.3333333333333333),
(4, 0.3333333333333333), (7, 0.3333333333333333), (8, 0.2222222222222222),
(9, 0.1111111111111111)]
Betweenness Centrality
[(7, 0.38888888888888884), (5, 0.23148148148148148), (6, 0.2314814814814814
8), (8, 0.2222222222222222), (3, 0.10185185185185183), (0, 0.023148148148148
143), (1, 0.023148148148148143), (2, 0.0), (4, 0.0), (9, 0.0)]
Closeness Centrality
[(5, 0.6428571428571429), (6, 0.6428571428571429), (3, 0.6), (7, 0.6), (0,
0.5294117647058824), (1, 0.5294117647058824), (2, 0.5), (4, 0.5), (8, 0.4285
7142857142855), (9, 0.3103448275862069)]
```

In [35]:

```
nx.draw_networkx(kkg)
```



In [7]:

```
!pip install pygithub
```

```
Collecting pygithub
  Downloading https://files.pythonhosted.org/packages/c1/1f/9dc4ba315eeea222
473cf4c15d3e665f32d52f859d9d6e73219d0a408969/PyGithub-1.55-py3-none-any.whl
 (https://files.pythonhosted.org/packages/c1/1f/9dc4ba315eeea222473cf4c15d3e
665f32d52f859d9d6e73219d0a408969/PyGithub-1.55-py3-none-any.whl) (291kB)
Collecting pyjwt>=2.0 (from pygithub)
  Downloading https://files.pythonhosted.org/packages/3f/32/d5d3cab27fee7f6b
22d7cd7507547ae45d52e26030fa77d1f83d0526c6e5/PyJWT-2.1.0-py3-none-any.whl (h
ttps://files.pythonhosted.org/packages/3f/32/d5d3cab27fee7f6b22d7cd7507547ae
45d52e26030fa77d1f83d0526c6e5/PyJWT-2.1.0-py3-none-any.whl)
Collecting deprecated (from pygithub)
  Downloading https://files.pythonhosted.org/packages/51/6a/c3a0408646408f72
83b7bc550c30a32cc791181ec4618592eec13e066ce3/Deprecated-1.2.13-py2.py3-none-
any.whl (https://files.pythonhosted.org/packages/51/6a/c3a0408646408f7283b7b
c550c30a32cc791181ec4618592eec13e066ce3/Deprecated-1.2.13-py2.py3-none-any.w
hl)
Collecting pynacl>=1.4.0 (from pygithub)
  Downloading https://files.pythonhosted.org/packages/72/0a/c489e5fd7ed00993
f0d7e96faa1b1ecbec6cb64e6fdeba1f200d3f7be410/PyNaCl-1.4.0-cp36-cp36m-win_amd
64.whl (https://files.pythonhosted.org/packages/72/0a/c489e5fd7ed00993f0d7e9
6faa1b1ecbec6cb64e6fdeba1f200d3f7be410/PyNaCl-1.4.0-cp36-cp36m-win_amd64.wh
l) (206kB)
Requirement already satisfied: requests>=2.14.0 in c:\programdata\anaconda3
\lib\site-packages (from pygithub)
Requirement already satisfied: wrapt<2,>=1.10 in c:\programdata\anaconda3\li
b\site-packages (from deprecated->pygithub)
Requirement already satisfied: cffi>=1.4.1 in c:\programdata\anaconda3\lib\s
ite-packages (from pynacl>=1.4.0->pygithub)
Requirement already satisfied: six in c:\programdata\anaconda3\lib\site-pack
ages (from pynacl>=1.4.0->pygithub)
Requirement already satisfied: chardet<3.1.0,>=3.0.2 in c:\programdata\anaco
nda3\lib\site-packages (from requests>=2.14.0->pygithub)
Requirement already satisfied: idna<2.7,>=2.5 in c:\programdata\anaconda3\li
b\site-packages (from requests>=2.14.0->pygithub)
Requirement already satisfied: urllib3<1.23,>=1.21.1 in c:\programdata\anaco
nda3\lib\site-packages (from requests>=2.14.0->pygithub)
Requirement already satisfied: certifi>=2017.4.17 in c:\programdata\anaconda
3\lib\site-packages (from requests>=2.14.0->pygithub)
Requirement already satisfied: pycparser in c:\programdata\anaconda3\lib\sit
e-packages (from cffi>=1.4.1->pynacl>=1.4.0->pygithub)
Installing collected packages: pyjwt, deprecated, pynacl, pygithub
Successfully installed deprecated-1.2.13 pygithub-1.55 pyjwt-2.1.0 pynacl-1.
4.0


You are using pip version 9.0.1, however version 21.2.4 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip'
command.
```

2. Create an interest graph of a github user by adding ‘follows’ as edges

a. Find the stargazers of a github user

In [3]:

```
from github import Github
ACCESS_TOKEN = "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"
USER = 'dphi-official'
REPO = 'Machine_Learning_Bootcamp'
#REPO = 'Mining-the-Social-Web-2nd-Edition'
client = Github(ACCESS_TOKEN, per_page=100)
user = client.get_user(USER)
repo = user.get_repo(REPO)
```

b. Create a graph of the star gazers using the networkx package and get the information about the graph

In [4]:

```
g = nx.DiGraph()
g.add_node(repo.name + '(repo)', type='repo', lang=repo.language, owner=user.login)
stargazers = [ s for s in repo.get_stargazers() ]
print("Number of stargazers", len(stargazers))
for sg in stargazers:
    g.add_node(sg.login + '(user)', type='user')
    g.add_edge(sg.login + '(user)', repo.name + '(repo)', type='gazes')
```

```
Number of stargazers 67
```

c. Add "follows" edges between stargazers in the graph if any relationships exist

In [5]:

```
import sys

for i, sg in enumerate(stargazers):

    # Add "follows" edges between stargazers in the graph if any relationships exist
    try:
        for follower in sg.get_followers():
            if follower.login + '(user)' in g:
                g.add_edge(follower.login + '(user)', sg.login + '(user)',
                           type='follows')
    except Exception as e: #ssl.SSLError
        print("Encountered an error fetching followers for", sg.login, "Skipping.", file=sy
        print(e, file=sys.stderr)

    print ("Processed", i+1, " stargazers.")
    print ("Num nodes",g.number_of_nodes())
    print ("edges",g.number_of_edges())
    print ("Rate limit remaining", client.rate_limiting)
nx.write_gpickle(g, "github.gpickle.1")
```

```
Processed 1  stargazers.
Num nodes 68
edges 67
Rate limit remaining (4910, 5000)
Processed 2  stargazers.
Num nodes 68
edges 67
Rate limit remaining (4909, 5000)
Processed 3  stargazers.
Num nodes 68
edges 67
Rate limit remaining (4908, 5000)
Processed 4  stargazers.
Num nodes 68
edges 67
Rate limit remaining (4907, 5000)
Processed 5  stargazers.
Num nodes 68
edges 67
```

3. Explore the graph with the updates ‘follows’ edges

a. Get the information about the updated graph

In [6]:

```
print(nx.info(g))
```

```
Name: 
Type: DiGraph
Number of nodes: 68
Number of edges: 88
Average in degree:   1.2941
Average out degree:   1.2941
```

b. Find the number of ‘follow’ edges

In [7]:

```
print(len([e for e in g.edges(data=True) if e[2]["type"] == "follows"]))
```

```
21
```

c. Find the number of popular users and the top 10 users

In [9]:

```
from collections import Counter
```

In [10]:

```
c = Counter([e[1] for e in g.edges(data=True) if e[2]["type"] == "follows"])
popular_users = [ (u, f) for (u, f) in c.most_common() if f>1 ]
print("Number of popular users", len(popular_users))

print("Top 10 popular users", popular_users[:10])
```

```
Number of popular users 5
Top 10 popular users [('rowers7(user)', 3), ('semanurkps(user)', 3), ('deepc
hatterjeevns(user)', 2), ('KC2016(user)', 2), ('lillaszulyovszky(user)', 2)]
```

d. Remove the super node from the graph and calculate the centrality measures

In [11]:

```
h = g.copy()
h.remove_node('Machine_Learning_Bootcamp(repo)')
```

4. Visualise the created interest graph

a. Create a subgraph from the original interest graph- select the user nodes and get the information about the updated graph

In [24]:

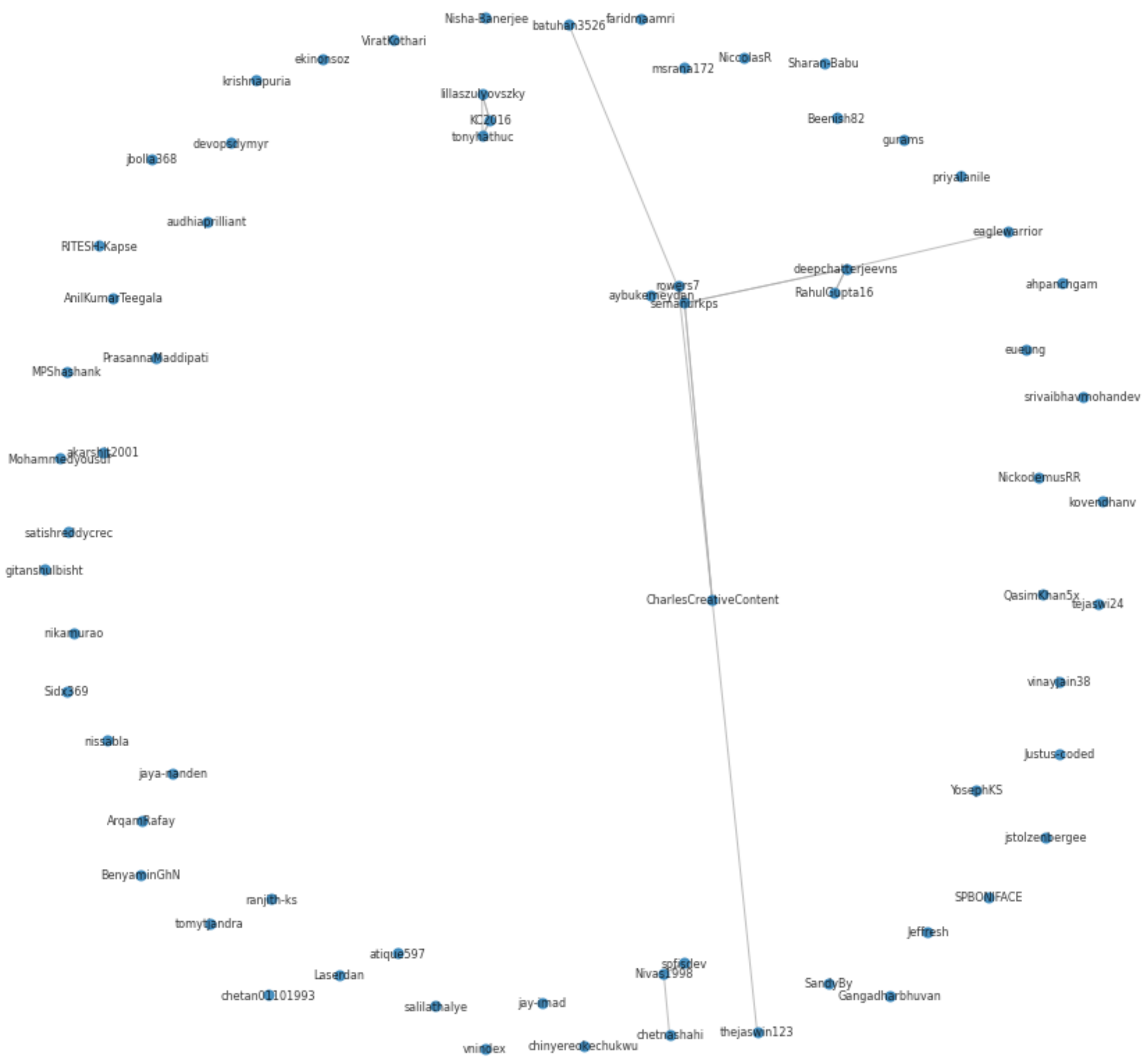
```
mtsw_users = [n for n in g if g.node[n]["type"] == "user"]
h = g.subgraph(mtsw_users)
print("Stats on the extracted subgraph")
print(nx.info(h))
```

```
Stats on the extracted subgraph
Name:
Type: SubDiGraph
Number of nodes: 67
Number of edges: 21
Average in degree:   0.3134
Average out degree:   0.3134
```

b. Visualise the extracted graph using matplotlib and networkx

In [12]:

```python
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
%matplotlib inline

fig = plt.figure(figsize=(15,15))
ax = fig.add_subplot(111)
labels = dict([(n, n.split('(user)')[0]) for n in h.nodes()])
nx.draw(h, pos=nx.spring_layout(h),arrows=False, ax=ax, node_size=50,
edge_color='#aaaaaa', alpha=0.8, labels=labels, font_size=8)
```



Nisha-Banerjee
batuhan3526
faridmaamri
ViratKothari
ekinonsoz
krishnapuria
msrana172
NicolasR
Sharan-Babu
lillaszulyovszky
KC2016
tonyhathuc
Beenish82
gurams
devopsdymyr
jbolla368
priyalanile
audhiabrilliant
RITESH-Kapse
eaglewarrior
deepchatterjeevns
rowers7
aybukemeydan
semanurkps
RahulGupta16
ahpanchgam
AnilKumarTeegala
euelung
PrasannaMaddipati
MPShashank
srivaibhavmohandev
akarshit2001
Mohammedyousuf
NickodemusRR
kovendhanv
satishreddycrec
gitanshulbisht
CharlesCreativeContent
QasimKhan5x
tejaswi24
nikamurao
vinayjain38
Sidx369
nissabla
Justus-coded
jaya-nanden
YosephKS
ArqamRafay
jstolzenbergee
BenyaminGhN
SPBONIFACE
ranjith-ks
tomytjandra
Jeffresh
atique597
sprlsdev
Nivas1998
Lasendan
SandyBy
Gangadharbhuvan
chetan01101993
salilathalye
jay-imad
chetnashahi
thejaswin123
vnindex
chinyereokechukwu