

PracticalExam1

January 25, 2020

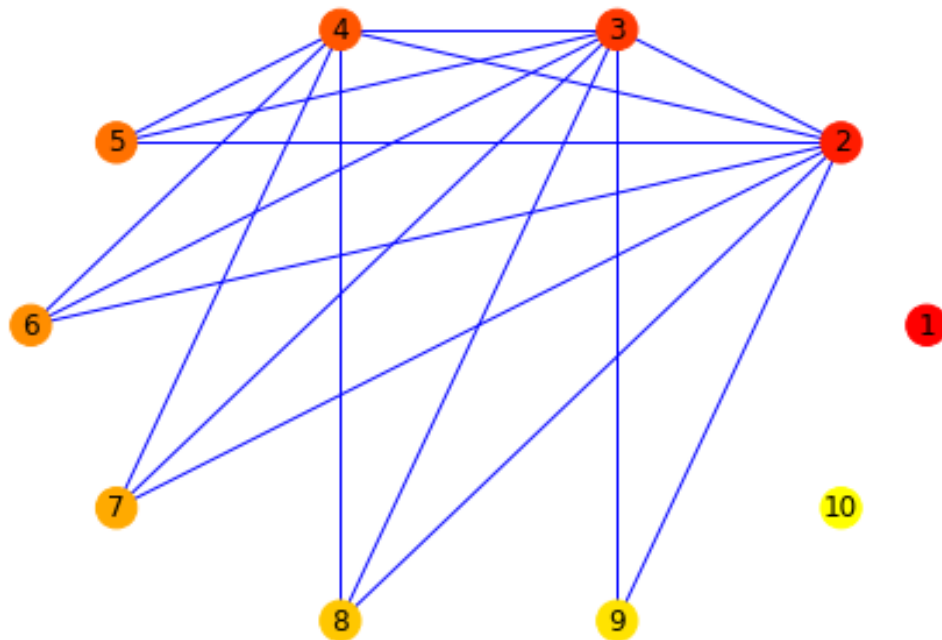
1 Graph Theory Practical Exam I

by Snigdha Jamwal

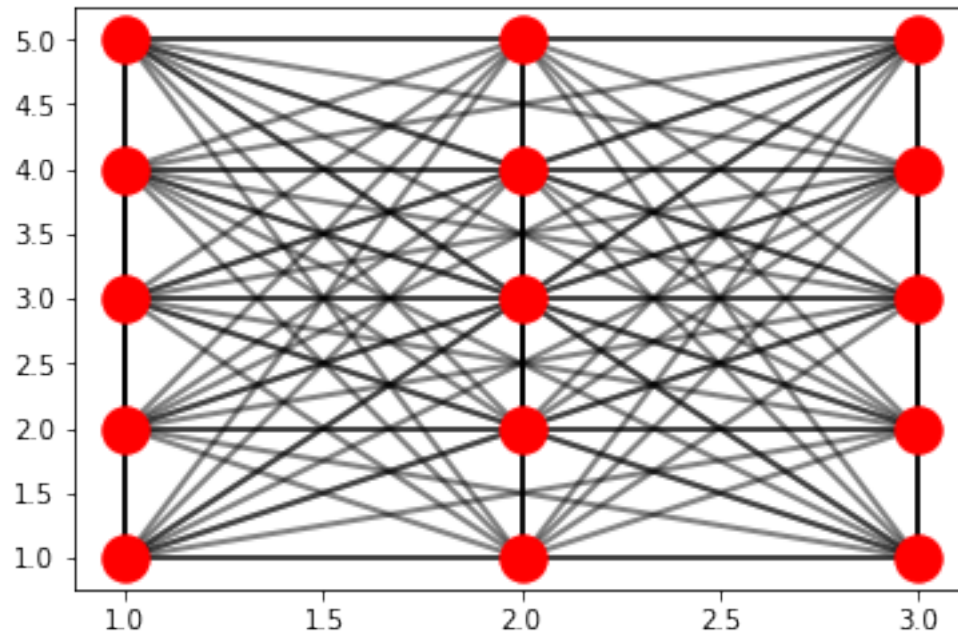
```
In [1]: import networkx as nx
import matplotlib.pyplot as plt
```

```
In [61]: # 1
G=nx.Graph()
G.add_nodes_from(range(1,11))
n=0
for i in range(1,11):
    for j in range(i+1,10):
        n=n+1
        if n>20:
            break
        else:
            G.add_edge(i+1,j)

nx.draw_circular(G,with_labels=True,node_color=range(10),edge_color="b",cmap=plt.cm.a
plt.show()
```



```
In [22]: # 2
G=nx.complete_graph(15)
pos={0:(1,1),
     1:(1,2),
     2:(1,3),
     3:(1,4),
     4:(1,5),
     5:(2,1),
     6:(2,2),
     7:(2,3),
     8:(2,4),
     9:(2,5),
     10:(3,1),
     11:(3,2),
     12:(3,3),
     13:(3,4),
     14:(3,5)}
nx.draw_networkx_nodes(G,pos)
nx.draw_networkx_edges(G,pos,alpha=0.5,width=2)
plt.show()
print("Adjacency List",nx.to_dict_of_lists(G))
print("Adjacency Matrix",nx.adj_matrix(G).todense())
```



Adjacency List {0: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14], 1: [0, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14], 2: [0, 1, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14], 3: [0, 1, 2, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14], 4: [0, 1, 2, 3, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14], 5: [0, 1, 2, 3, 4, 6, 7, 8, 9, 10, 11, 12, 13, 14], 6: [0, 1, 2, 3, 4, 5, 7, 8, 9, 10, 11, 12, 13, 14], 7: [0, 1, 2, 3, 4, 5, 6, 8, 9, 10, 11, 12, 13, 14], 8: [0, 1, 2, 3, 4, 5, 6, 7, 9, 10, 11, 12, 13, 14], 9: [0, 1, 2, 3, 4, 5, 6, 7, 8, 10, 11, 12, 13, 14], 10: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 11, 12, 13, 14], 11: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 13, 14], 12: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 13, 14], 13: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 14], 14: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13]}

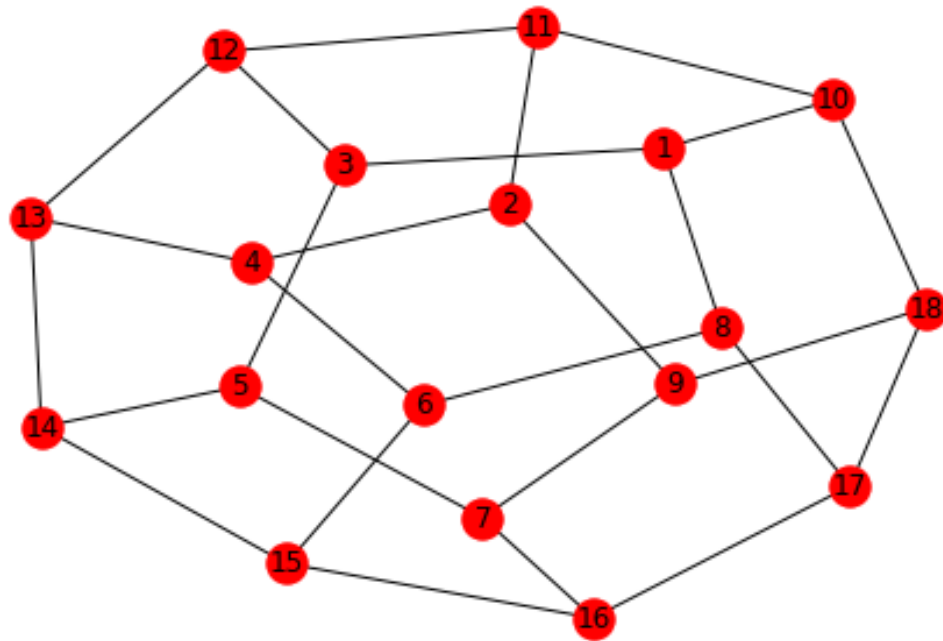
Adjacency Matrix [[0 1 1 1 1 1 1 1 1 1 1 1 1 1 1]

```
[1 0 1 1 1 1 1 1 1 1 1 1 1 1 1]
[1 1 0 1 1 1 1 1 1 1 1 1 1 1 1]
[1 1 1 0 1 1 1 1 1 1 1 1 1 1 1]
[1 1 1 1 0 1 1 1 1 1 1 1 1 1 1]
[1 1 1 1 1 0 1 1 1 1 1 1 1 1 1]
[1 1 1 1 1 1 0 1 1 1 1 1 1 1 1]
[1 1 1 1 1 1 1 0 1 1 1 1 1 1 1]
[1 1 1 1 1 1 1 1 0 1 1 1 1 1 1]
[1 1 1 1 1 1 1 1 1 0 1 1 1 1 1]
[1 1 1 1 1 1 1 1 1 1 0 1 1 1 1]
[1 1 1 1 1 1 1 1 1 1 1 0 1 1 1]
[1 1 1 1 1 1 1 1 1 1 1 1 0 1 1]
[1 1 1 1 1 1 1 1 1 1 1 1 1 0 1]
[1 1 1 1 1 1 1 1 1 1 1 1 1 1 0]
```

In [32]: # 3

```
G=nx.Graph()
G.add_nodes_from(range(1,19))
G.add_cycle([1,3,5,7,9,2,4,6,8])
G.add_cycle([10,11,12,13,14,15,16,17,18])
for i in range(1,10):
    G.add_edge(i,i+9)
#shells=([1,3,5,7,9,2,4,6,8],[10,11,12,13,14,15,16,17,18])
```

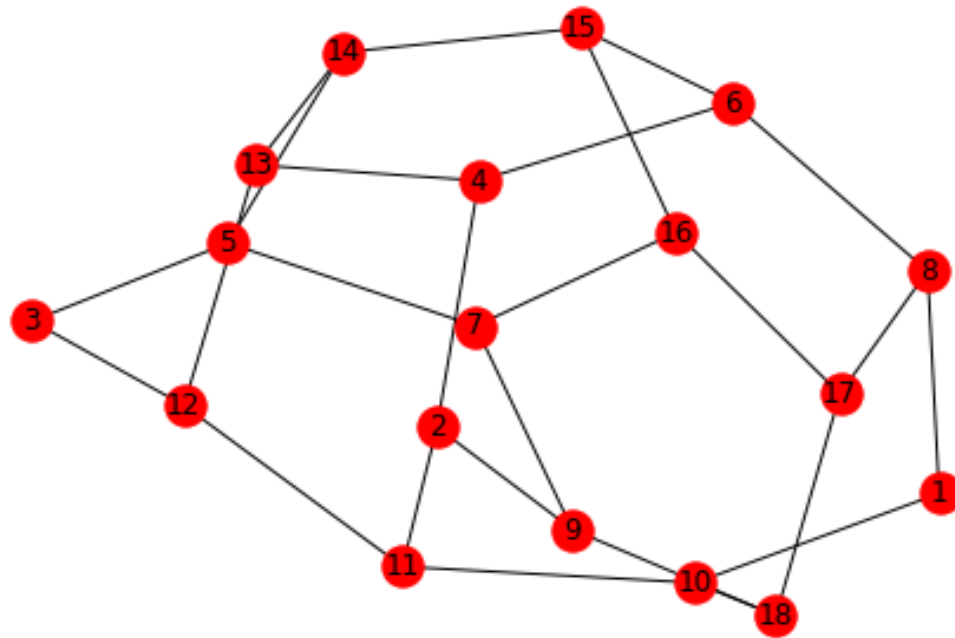
```
nx.draw(G,with_labels=True)
plt.show()
```



```
In [33]: print("Spanning Graph of G")
         G.remove_edge(1,3)
         nx.draw(G,with_labels=True)
         plt.show()
```

Spanning Graph of G

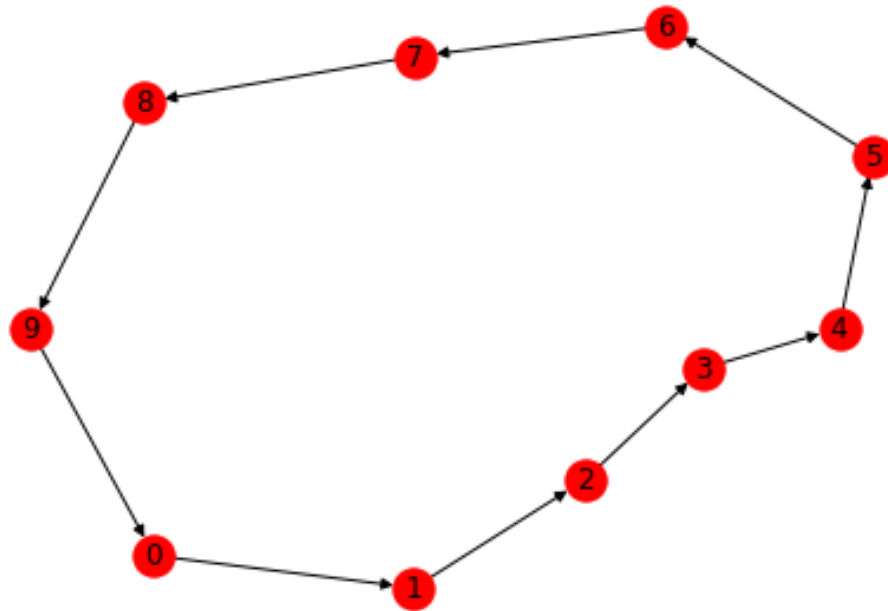
C:\Users\Snigdha\Anaconda3\lib\site-packages\networkx\drawing\nx_pylab.py:611: MatplotlibDeprecationWarning:
if cb.is_numlike(alpha):



```
In [39]: # 4
G=nx.DiGraph()
G.add_nodes_from(range(0,10))
for i in range(0,9):
    G.add_edge(i,i+1)
G.add_edge(9,0)
print("Vertices of the graph",G.nodes)
print("Edges of the graph",G.edges)
nx.draw(G,with_labels=True)
```

Vertices of the graph [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

Edges of the graph [(0, 1), (1, 2), (2, 3), (3, 4), (4, 5), (5, 6), (6, 7), (7, 8), (8, 9), (9, 0), (4, 7), (5, 7), (5, 12), (7, 9), (7, 11), (9, 10), (10, 11), (10, 18), (11, 12), (12, 3), (13, 14), (14, 15), (15, 16), (16, 17), (17, 18), (18, 10), (16, 4), (16, 7), (13, 5), (13, 13), (14, 13), (15, 15), (16, 16), (17, 17), (18, 18)]



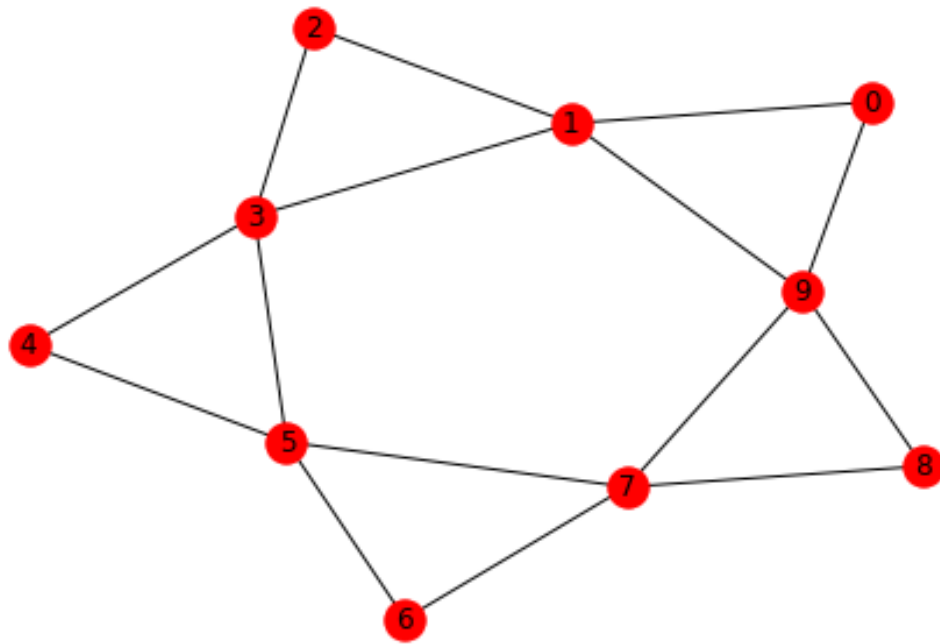
In [47]: # 5

```

G=nx.Graph()
G.add_nodes_from(range(0,10))
for i in range(1,11):
    G.add_edge(i%10,(i+1)%10)
G.add_edges_from([(1,3),(3,5),(5,7),(7,9),(1,9)])
print("Edge List", G.edges)
print("Vertex List", G.nodes)
print("Degree Sequence")
deg=[G.degree(v) for v in G]
deg.sort(reverse=True)
print(deg)
nx.draw(G,with_labels=True)

```

Edge List [(0, 9), (0, 1), (1, 2), (1, 3), (1, 9), (2, 3), (3, 4), (3, 5), (4, 5), (5, 6), (5, 7), (6, 7), (7, 8), (8, 9), (9, 0)]
Vertex List [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
Degree Sequence
[4, 4, 4, 4, 4, 2, 2, 2, 2, 2]



In []: