Riley Payung

CDS 292

Assignment 3

Imports

In [1]:

```
import networkx as netx
import matplotlib.pyplot as plt
import numpy as np
```

Question 3.10

rho = 0.5. n = 8.

0.5 = 2m / 8*7

-> 0.5 = 2m/56

-> 0.5 = m / 28

-> 14 = m

The number of links this network has is 14.

Question 3.13

In [2]:

```
W = netx.Graph()
n = ["Sagretti","Chapin","Obrien","Parkinson","Krogh","Mitchell",
     "Porter","Strachan","Hunt(H)","Liddy","Magruder","Nixon",
     "Sturgis","Baldwin","McCord","Dean","Haldeman","Martinez",
     "LaRue","Colson","Kalmbach","Gray","Ehrlichman","Barker"]
for i in n:
    W.add_node(i);

e = [("Sagretti","Chapin"),("Krogh","Hunt(H)"),("Krogh","Liddy"),("McCord","Mitchell"),("M
agruder","Mitchell"),
     ("Magruder","Porter"),("Magruder","Strachan"),("McCord","Parkinson"),("Hunt(H)","Park
inson"),
     ("Hunt(H)","Liddy"),("Baldwin","Hunt(H)"),("McCord","Hunt(H)"),("Hunt(H)","Dean"),("D
ean","Hunt(H)"),
     ("Baldwin","Liddy"),("McCord","Liddy"),("Dean","Liddy"),("Magruder","Liddy"),("McCor
d","Magruder"),
     ("Magruder","Nixon"),("Magruder","Haldeman"),("Magruder","Dean"),("Baldwin","Sturgis"
),("Baldwin","McCord"),
     ("McCord","Dean"),("McCord","LaRue"),("Dean","Nixon"),("Dean","Haldeman"),("Dean","La
Rue"),("Dean","Martinez"),
     ("Dean","Barker"),("Dean","Ehrlichman"),("Dean","Gray"),("Dean","Kalmbach"),("McCord"
,"Colson"),("Hunt(H)","Colson"),
     ("Gray","Dean")]
for i in e:
    W.add_edge(i[0],i[1])
```

Question 3.14

In [3]:

```
NumNodes = len(n);
NumEdges = len(e);
density = (2 * NumEdges) / (NumNodes * (NumNodes - 1))
```

In [4]:

```
print (NumNodes,NumEdges,density)
```

```
24 37 0.13405797101449277
```

Question 3.17

In [5]:

```
N = netx.Graph()
for i in range(6):
    N.add_node(str(i))
for i in N.nodes():
    for j in N.nodes():
        if (i != j):
            N.add_edge(i,j);

rho = (2 * len(N.edges())) / (len(N.nodes()) * (len(N.nodes()) - 1))
```

In [6]:

```
print(N.nodes())
```

```
['0', '1', '2', '3', '4', '5']
```

In [7]:

```
print(N.edges())
```

```
[('0', '1'), ('0', '2'), ('0', '3'), ('0', '4'), ('0', '5'), ('1', '2'),
('1', '3'), ('1', '4'), ('1', '5'), ('2', '3'), ('2', '4'), ('2', '5'), ('3',
'4'), ('3', '5'), ('4', '5')]
```

In [8]:

```
print(rho)
```

```
1.0
```

Question 3.18

In [9]:

```
Net18 = netx.Graph()
for i in range(100):
    if (i != 99):
        Net18.add_edge(i,i+1)
    else:
        Net18.add_edge(0,i);

Nodes = len(Net18.nodes())
Edges = len(Net18.edges())
rho = (2 * Edges) / (Nodes * (Nodes - 1))
```

In [10]:

```
print(Net18.nodes())
```

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

In [11]:

```
print(Net18.edges())
```

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

In [12]:

```
print(rho)
```

```
0.020202020202020204
```

Question 3.20

In [13]:

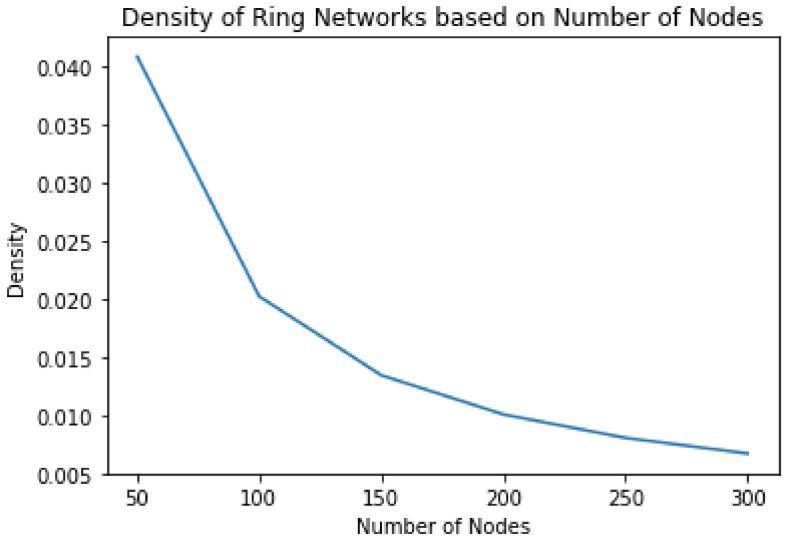
```
densityN = [50,100,150,200,250,300]

def densityRing(n):
    return (2 * n) / (n * (n-1));

densities = [];
for i in densityN:
    densities.append(densityRing(i))
```

In [14]:

```
plt.plot(densities)
plt.title("Density of Ring Networks based on Number of Nodes");
plt.xlabel("Number of Nodes");
plt.ylabel("Density")
plt.xticks(np.arange(0,6,1),densityN[:])
plt.show()
```



Question 4.1

Are the following networks valid based on each node's degree?

A) Yes

B) No, since k_2 = 3.

C) Yes, Each node has 1 connection, following a Z-pattern.

D) Yes, Nodes 1 and 3 are isolated.

E) Technically yes, but all nodes are isolated.

F) No, since K_2 = 4.

G) Yes, since this is a complete network.

Question 4.2

.

.

.

.

.

.

.

Question 4.3

.

.

.

.

.

.

.

Question 4.4

The degree can be determined by summing the elements in the row i of the adjacency matrix. SUM(a_ij). We can also do SUM(a_ji) which uses the columns instead of the rows, in which we use the symmetry property of the adjacency matrix.

Question 4.5

- You can write the adjacency matrix with all zeroes
- you can write a degree network: k_1 = 0, k_2 = 0, ..., k_n-1 = 0, k_n = 0.
- you can write a link list file with just one column. (this is not a good way to do it, since the minimal list is of two columns, but it can be done)

Question 4.6

A 3x3 Adjacency matrix since we only have 10 numbers, and the sqrt(10) is not whole (it is in fact just above 3). We are left with a remainder of 1, therefore we need to drop it.