In [7]:

```python
import re
import networkx as nx
```

In [ ]:

```python
import pandas as pd
df = pd.read_csv('./data/smartphone.csv', encoding='utf-8')
galexy_posts = df.get('Title') + " " + df.get('Description')
galexy_post_date = df.get('Post Date')
```

In [ ]:

```python
from konlpy.tag import Okt, Kkma, Komoran, Hannanum
tagger = Okt()

galexy_stop_words = "은 이 것 등 더 를 좀 즉 인 옹 때 만 원 이때 개 일 기 시 럭 갤 성 삼 스 폰 트 드 기 이 리 폴 사 전 마 자 플 블 가 중 북 수 팩 년 월
galexy_stop_words = galexy_stop_words.split(' ')
galexy_stop_words[0:10]
```

```
[0.019s][warning][os,thread] Attempt to protect stack guard pages failed (0x00000001697c8000-0x00000001697d4000).
[0.019s][warning][os,thread] Attempt to deallocate stack guard pages failed.
```

Out[ ]:

```
['은', '이', '것', '등', '더', '를', '좀', '즉', '인', '옹']
```

In [ ]:

```python
galexy_nouns = []
for post in galexy_posts:
    for noun in tagger.nouns(post):
        if noun not in galexy_stop_words:
            galexy_nouns.append(noun)

galexy_nouns[0:10]
```

In [ ]:

```python
from collections import Counter
num_top_nouns = 20
galexy_nouns_counter = Counter(galexy_nouns)
galexy_top_nouns = dict(galexy_nouns_counter.most_common(num_top_nouns))
```

In [ ]:

```python
galexy_sentences = []
for post in galexy_posts:
    galexy_sentences.extend(re.split('; |\.|\?|\!', post))
galexy_sentences[0:10]
```

In [ ]:

```python
galexy_sentences_nouns = []
for sentence in galexy_sentences:
    sentence_nouns = tagger.nouns(sentence)
    galexy_sentences_nouns.append(sentence_nouns)
galexy_sentences_nouns[0:10]
```

In [ ]:
```
galexy_word2id = {w: i for i, w in enumerate(galexy_top_nouns.keys())}
galexy_word2id
```

In [ ]:
```
galexy_id2word = {i: w for i, w in enumerate(galexy_top_nouns.keys())}
galexy_id2word
```

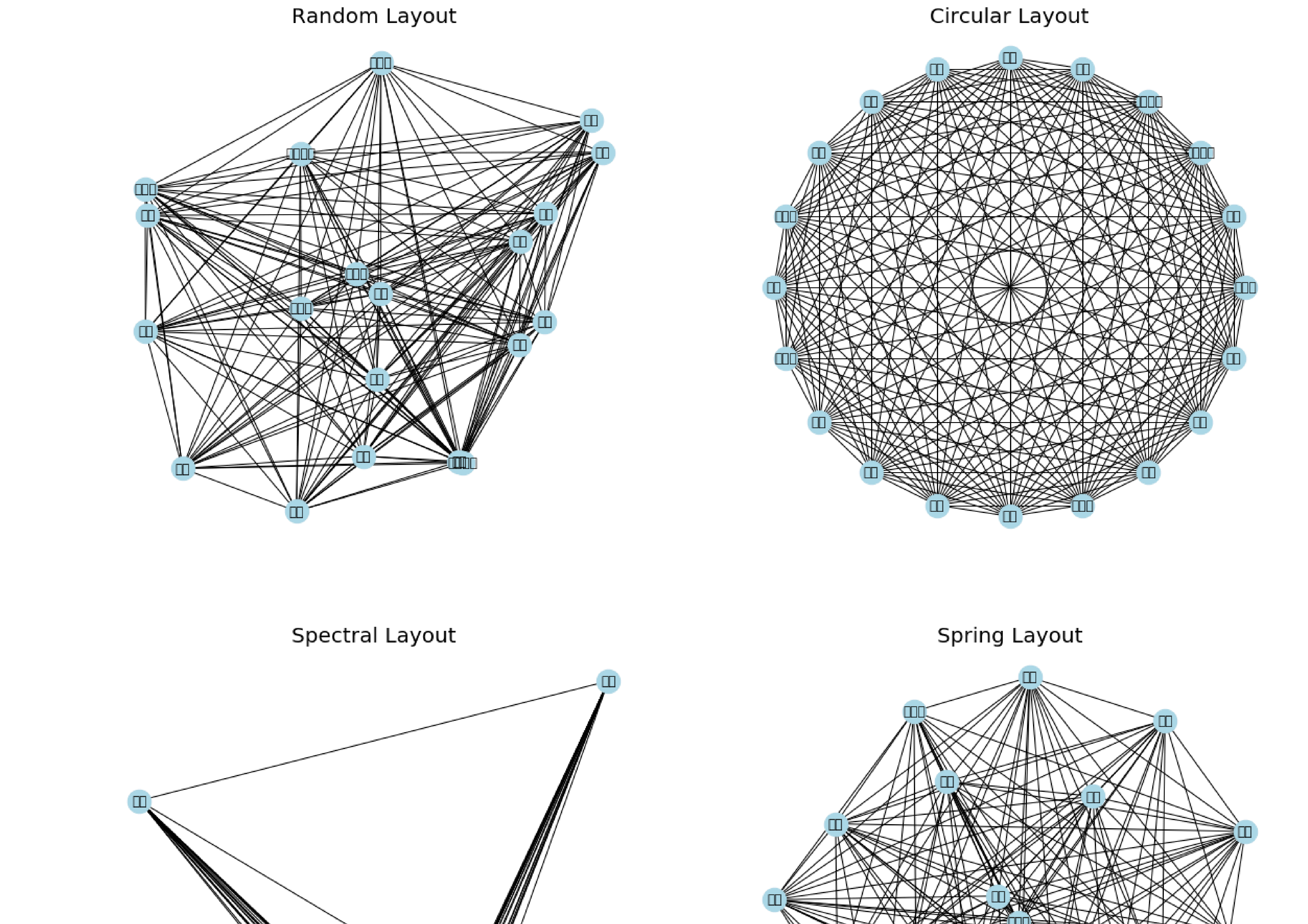
In [ ]:
```
import numpy as np
galexy_adjacent_matrix = np.zeros((num_top_nouns, num_top_nouns), int)
for sentence in galexy_sentences_nouns:
    for wi, i in galexy_word2id.items():
        if wi in sentence:
            for wj, j in galexy_word2id.items():
                if i != j and wj in sentence:
                    galexy_adjacent_matrix[i][j] += 1
galexy_adjacent_matrix
```

In [ ]:
```
galexy_network = nx.from_numpy_matrix(galexy_adjacent_matrix)
list(galexy_network.adjacency())
```

In [ ]:
```
import matplotlib.pyplot as plt
from matplotlib import font_manager as fm
from matplotlib import rc

font_path="./font/NanumBarunGothic.ttf"
font_name = fm.FontProperties(fname=font_path).get_name()
rc('font', family=font_name)

fig = plt.figure()
fig.set_size_inches(20, 20)
ax = fig.add_subplot(1, 1, 1)
ax.axis("off")
option = {
    'node_color' : 'lightblue',
    'node_size' : 2000,
    'size' : 2
}
nx.draw(galexy_network, labels=galexy_id2word, font_family=font_name, ax=ax, **option)
```

```
c:\python\venv\tensorflow\lib\site-packages\matplotlib\font_manager.py:1241: UserWarning: findfont: Font family ['NanumBarunGoth
ic'] not found. Falling back to DejaVu Sans.
  (prop.get_family(), self.defaultFamily[fontext]))
```

In [ ]:
```
fig = plt.figure()
fig.set_size_inches(20, 20)
option = {
    'node_color' : 'lightblue',
    'node_size' : 500,
    'size' : 100
}

plt.subplot(221)
plt.title('Random Layout', fontsize=20)
nx.draw_random(galexy_network, labels=galexy_id2word, font_family=font_name, **option)
plt.subplot(222)
plt.title('Circular Layout', fontsize=20)
nx.draw_circular(galexy_network, labels=galexy_id2word, font_family=font_name, **option)
plt.subplot(223)
plt.title('Spectral Layout',fontsize=20)
nx.draw_spectral(galexy_network, labels=galexy_id2word, font_family=font_name, **option)
plt.subplot(224)
plt.title('Spring Layout',fontsize=20)
nx.draw_spring(galexy_network, labels=galexy_id2word, font_family=font_name, **option)
```


Random Layout
Circular Layout
Spectral Layout
Spring Layout

In [ ]:
```
#Degree
nx.degree_centrality(galexy_network)
```

In [ ]:
```
#Eigenvector
nx.eigenvector_centrality(galexy_network, weight='weight')
```

In [ ]:
```
#Closeness
nx.closeness_centrality(galexy_network, distance='weight')
```

In [ ]:
```
#Current Flow Closeness
nx.current_flow_closeness_centrality(galexy_network)
```

In [ ]:
```
#Current Flow Betweenness
nx.current_flow_betweenness_centrality(galexy_network)
```

In [ ]:
```
#Communicability Betweenness
nx.communicability_betweenness_centrality(galexy_network)
```

In [ ]:
```

```