

## ▼ Dasgal1


```
# . Eigenvector Centrality- хоорондын харьцааг тооцолдог код бич.
import networkx as nx

# Графыг үүсгэх
G = nx.erdos_renyi_graph(10, 0.5) # 10 оройтой, 0.5 магадлалтай случай граф

# Eigenvector Centrality-г тооцоолох
centrality = nx.eigenvector_centrality(G)

# Төвлөрөлтэй оройнуудыг харуулах
print("Eigenvector Centrality: ", centrality)

# Төвлөрөлтэй оройнуудыг томъёолсон харьцааг тооцоолох
for node, score in centrality.items():
    print(f"Орой {node} - Eigenvector Centrality: {score}")
```

 Eigenvector Centrality: {0: 0.3679616425969596, 1: 0.40006502799411847, 2: 0.23286020388213274, 3: 0.2004295619194541, 4: 0.3615180790516828, 5: 0.2524661142715228, 6: 0.4038612350681437, 7: 0.27536752605995646, 8: 0.3946894136128103, 9: 0.1449515572748636}

Орой 0 - Eigenvector Centrality: 0.3679616425969596  
 Орой 1 - Eigenvector Centrality: 0.40006502799411847  
 Орой 2 - Eigenvector Centrality: 0.23286020388213274  
 Орой 3 - Eigenvector Centrality: 0.2004295619194541  
 Орой 4 - Eigenvector Centrality: 0.3615180790516828  
 Орой 5 - Eigenvector Centrality: 0.2524661142715228  
 Орой 6 - Eigenvector Centrality: 0.4038612350681437  
 Орой 7 - Eigenvector Centrality: 0.27536752605995646  
 Орой 8 - Eigenvector Centrality: 0.3946894136128103  
 Орой 9 - Eigenvector Centrality: 0.1449515572748636

## ▼ Dasgal2

```
# ER (Erdos-Rényi) санамсаргүй графыг colab дээр үүсгэж, түүний хэлбэрүүд болох subcritical,
#critical, supercritical, connected regime хэрхэн үүсдэг талаар тус бүрд нь тайлбарлана уу.
import networkx as nx
import matplotlib.pyplot as plt

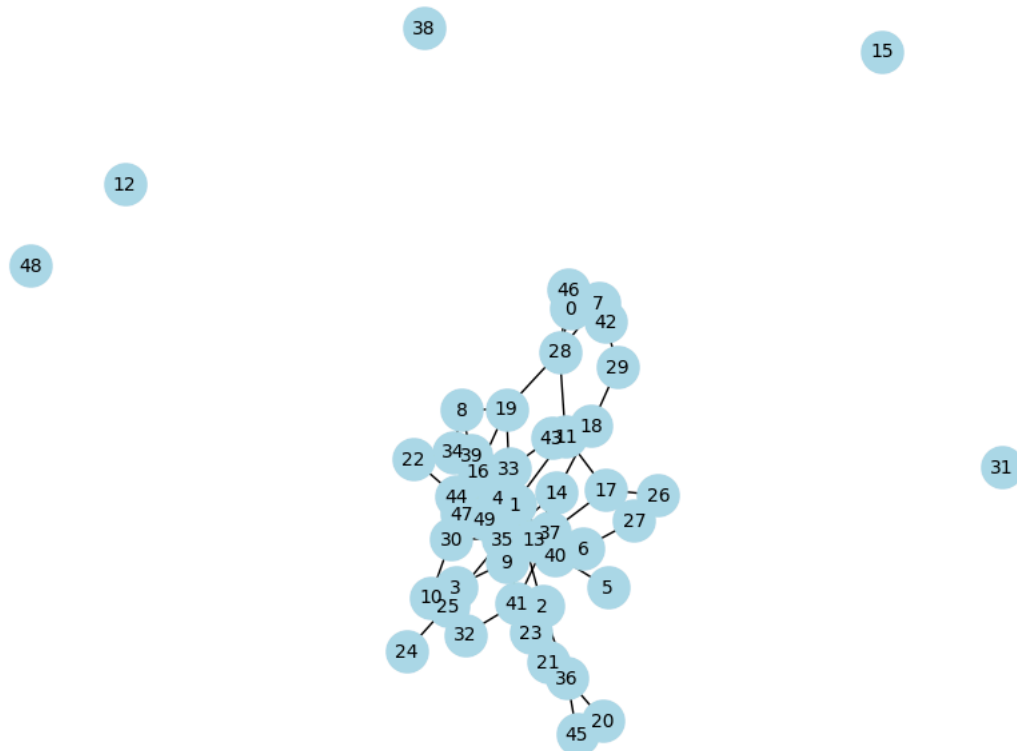
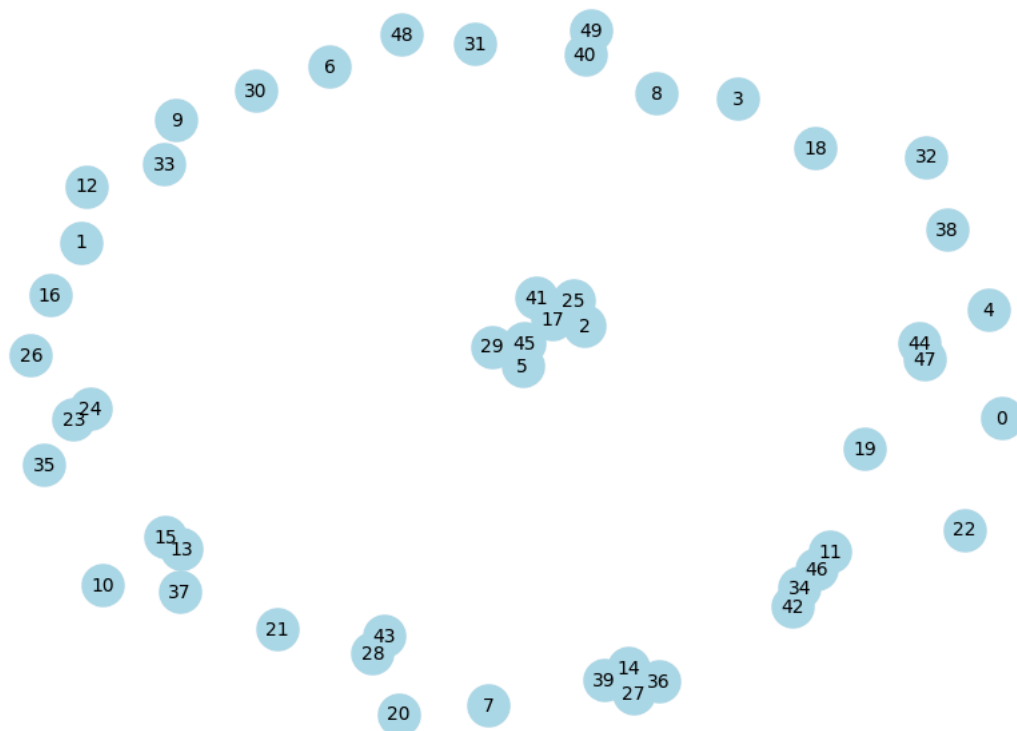
# ER граф үүсгэх болон харуулах функц
def plot_er_graph(n, p):
    G = nx.erdos_renyi_graph(n, p)
    plt.figure(figsize=(8, 6))
    nx.draw(G, with_labels=True, node_color='lightblue', font_size=10, node_size=500)
    plt.title(f"Erdős-Rényi Graph: n={n}, p={p}")
    plt.show()

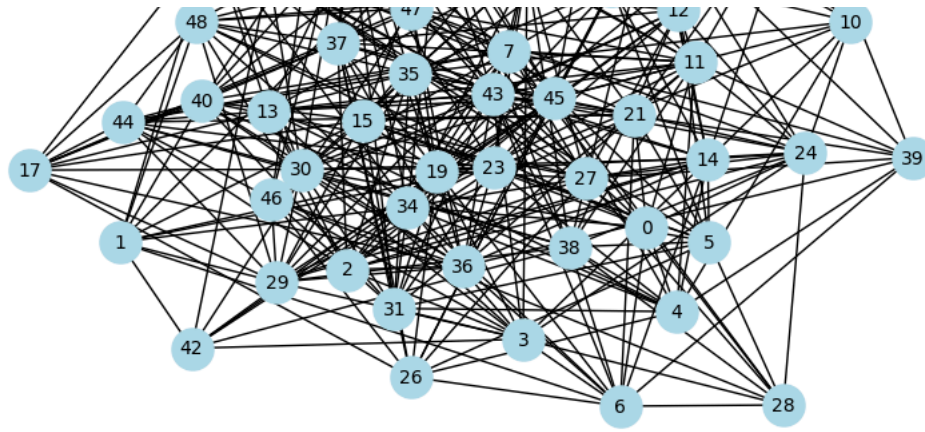
# Subcritical хэлбэр (p бага, граф нь холбогдоогүй)
n = 50 # оройн тоо
p_subcritical = 0.05 # бага магадлалаар холболт үүсэх
plot_er_graph(n, p_subcritical)

# Critical хэлбэр (p ойролцоо 1/n байх)
p_critical = 1 / n # критик утга
plot_er_graph(n, p_critical)

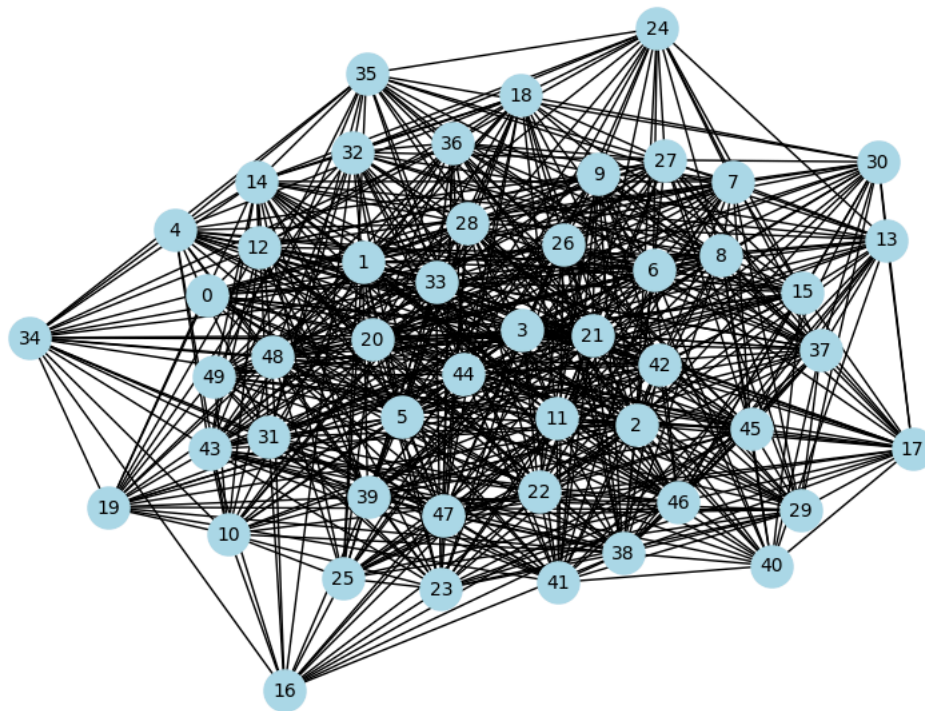
# Supercritical хэлбэр (p их, холбогдсон граф)
p_supercritical = 0.3 # өндөр магадлал
plot_er_graph(n, p_supercritical)

# Connected Regime (граф нь холбогдсон байх)
p_connected = 0.5 # холбогдсон граф үүсгэх магадлал
plot_er_graph(n, p_connected)
```

Erdős-Rényi Graph:  $n=50$ ,  $p=0.05$ Erdős-Rényi Graph:  $n=50$ ,  $p=0.02$ Erdős-Rényi Graph:  $n=50$ ,  $p=0.3$ 



Erdős-Rényi Graph:  $n=50$ ,  $p=0.5$



## Dasgal3

```
#Оройн тоог n = 100, хос орой болгоны холбогдох магадлалыг p = 0.06 гэж өгөн networkx санг
#ашиглан ER графыг үүсгэнэ үү. Граф ямар хэлбэрээр үүсэхийг тайлбарлана уу.
import networkx as nx
import matplotlib.pyplot as plt

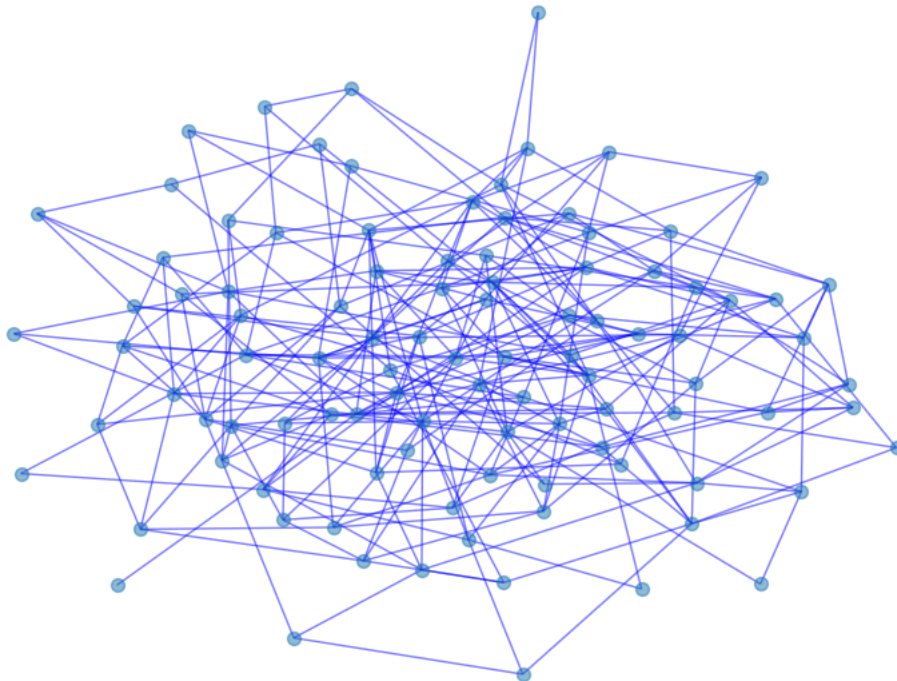
# ER граф үүсгэх
n = 100 # оройн тоо
p = 0.06 # хос орой хоорондын холбогдох магадлал

# ER граф үүсгэх
G = nx.erdos_renyi_graph(n, p)

# Графыг харуулах
plt.figure(figsize=(8, 6))
nx.draw(G, with_labels=False, node_size=50, edge_color='blue', alpha=0.5)
plt.title(f"Erdős-Rényi Graph: n={n}, p={p}")
plt.show()
```



Erdős-Rényi Graph: n=100, p=0.06



Графын хэлбэр: Subcritical, Critical, Supercritical хэлбэрүүд нь ER графыг үүсгэхэд ямар холбогдох магадлалтай утга дээр үндэслэн үүсэх вэ гэдгийг тодорхойлдог. Энэ нь оройн тоо болон холболтын магадлалтай шууд хамааралтай байдаг.

Энэхүү граф нь ямар хэлбэртэй болох вэ?  $n = 100$  оройтой,  $p = 0.06$  магадлалтай графын хувьд  $p$  нь  $1/n$ -ээс бага учир  $p \approx 1/100 = 0.01$  байдаг. Энэ тохиолдолд граф нь critical болон supercritical хооронд шилжиж эхэлдэг.

Тэгэхээр  $p = 0.06$  нь critical нөхцөлд ойрхон байгаа ч, бид supercritical нөхцөлд шилжиж байгаа гэсэн үг бөгөөд сүлжээ нь холбогдсон байж магадгүй.

Сүлжээгийн ихэнх орой нь хоорондоо холбогдсон байж магадгүй бөгөөд нэг хэсэг холбогдсон сүлжээ үүсч байгаа бөгөөд үүнийг connected гэж тодорхойлох болно.

Тиймээс, энэ граф нь connected (холбогдсон) байх боломжтой.

3. Тайлбар: Subcritical графууд нь ихэвчлэн олон оройтой боловч хоорондоо холбогдсон байдалгүй байдаг.

Critical графууд нь холбогдсон байдалтай боловч жижиг хэсгүүдэд хуваагдаж магадгүй.

Supercritical графууд нь ихэнх оройнууд нь холбогдсон байдаг бөгөөд бүх орой хоорондоо аль нэг замаар холбогдсон байдаг.

$p = 0.06$  нь supercritical нөхцөлд багтдаг бөгөөд эдгээр графууд нь ихэвчлэн бүх оройтой холбогдсон, бие даасан жижиг хэсгүүд байхгүй графууд юм.

Ингээд,  $n = 100$ ,  $p = 0.06$  утгаар үүсгэгдсэн ER граф нь connected (холбогдсон) граф болон харагдах бөгөөд энэ нь ER графын supercritical хэлбэрийн шинж тэмдэг юм.

## Dasgal4

```
# Average shortest path олох командыг бичнэ үү.
import networkx as nx

# Сүлжээ үүсгэх (жишээ нь ER граф)
n = 100 # оройн тоо
p = 0.06 # холбогдох магадлал
G = nx.erdos_renyi_graph(n, p)

# Average shortest path олох
average_shortest_path = nx.average_shortest_path_length(G)

print(f"Average shortest path length: {average_shortest_path}")
```

↗ Average shortest path length: 2.7105050505050503

## Dasgal5

```
# Average shortest path олох командыг бичнэ үү.
import networkx as nx

# Сүлжээ үүсгэх (жишээ нь ER граф)
n = 100 # оройн тоо
p = 0.06 # холбогдох магадлал
G = nx.erdos_renyi_graph(n, p)

# Average shortest path олох
average_shortest_path = nx.average_shortest_path_length(G)

print(f"Average shortest path length: {average_shortest_path}")
```

↗ Average shortest path length: 2.7674747474747474

## Тайлбар

Граф үүсгэх:

$n = 100$ : Сүлжээний оройн тоог 100 болгож тохируулсан.

$p = 0.06$ : Хоёр орой хоорондын холбогдох магадлал нь 0.06 (6%) байна. Энэ нь тухайн граф үүсэхэд орой бүр хоорондоо холбогдож буй магадлал юм.

$G = nx.erdos\_renyi\_graph(n, p)$ : ER граф-ыг үүсгэж байна. Erdős–Rényi (ER) загвар нь санамсаргүй холболттой граф юм. Энэ граф нь оройн тоо ( $n$ ) болон холбогдох магадлал ( $p$ ) дээр үндэслэн үүсдэг.

Average Shortest Path Тооцоолол:

$nx.average\_shortest\_path\_length(G)$ : Энэ нь граф  $G$ -ийн бүх орой хоорондын хамгийн богино замын уртаас авах дунджаар