

```

In [5]: import numpy as np
import random
import matplotlib.pyplot as plt
from tqdm import tqdm
from collections import deque

def plot_degree_distribution(graph, scale='linear', color='#D95319', alph
plt.close()
num_nodes = len(graph)
degrees = [len(neighbors) for neighbors in graph.values()]
max_degree = max(degrees) if degrees else 0

degree_counts = [0] * (max_degree + 1)
for degree in degrees:
    degree_counts[degree] += 1

degree_probabilities = [count / num_nodes for count in degree_counts]

plt.figure(figsize=(8, 6))
if scale == 'log':
    plt.xscale('log')
    plt.yscale('log')
    plt.title('Degree Distribution (Log-Log Scale)')
    plt.ylabel('log(P(k))')
    plt.xlabel('log(k)')
else:
    plt.title('Degree Distribution (Linear Scale)')
    plt.ylabel('P(k)')
    plt.xlabel('k')

plt.plot(range(max_degree + 1), degree_probabilities, 'o', markersize
plt.grid(True, which="both", linestyle="--")

if fit_line and scale == 'log':
    x_vals = list(range(expct_lo, expct_hi))
    y_vals = [(x ** -3) * expct_const for x in x_vals]
    plt.plot(x_vals, y_vals, color='#7f7f7f', linestyle='dashed')

plt.show()

def select_node_by_probability(graph, degree):
    degrees = {node: len(neighbors) for node, neighbors in graph.items()}
    total_degree = sum(degrees.values())

    if degree==3:
        # Proportional to the cube of the degree
        probabilities = [deg**3 / sum(deg**3 for deg in degrees.values())

    elif degree == 2:
        # Proportional to the square of the degree
        probabilities = [deg**2 / sum(deg**2 for deg in degrees.values())
    else:
        probabilities = [deg / total_degree for deg in degrees.values()]

    return np.random.choice(list(degrees.keys()), p=probabilities)

def add_edge_to_graph(graph, new_node, degree):
    existing_node = select_node_by_probability(graph, degree)

```

```

    if new_node in graph[existing_node]:
        print("Edge already exists! Retrying...")
        add_edge_to_graph(graph, new_node, degree)
    else:
        graph[new_node].add(existing_node)
        graph[existing_node].add(new_node)

def bfs_shortest_path_length(graph, start_node):
    queue = deque([(start_node, 0)])
    visited = {start_node}
    path_lengths = []

    while queue:
        node, depth = queue.popleft()
        path_lengths.append(depth)
        for neighbor in graph[node]:
            if neighbor not in visited:
                visited.add(neighbor)
                queue.append((neighbor, depth + 1))

    return sum(path_lengths) / len(path_lengths) if path_lengths else 0

def compute_characteristic_path_length(graph):
    if not graph:
        return 0
    total_length = sum(bfs_shortest_path_length(graph, node) for node in graph)
    return total_length / len(graph)

def assess_topology(graph):
    clustering_coeffs = []
    for node in graph:
        neighbors = graph[node]
        if len(neighbors) < 2:
            clustering_coeffs.append(0)
        else:
            links = sum(1 for neighbor in neighbors for other in neighbors
                        if neighbor < other)
            clustering_coeffs.append(2 * links / (len(neighbors) * (len(neighbors) - 1)))
    avg_clustering_coeff = sum(clustering_coeffs) / len(graph)
    print(f"Average Clustering Coefficient: {avg_clustering_coeff:.4f}")

    char_path_length = compute_characteristic_path_length(graph)
    print(f"Characteristic Path Length: {char_path_length:.4f}")

def barabasi_albert_simulation(degree):
    init_nodes = int(input("Enter the initial number of nodes (m_0): "))
    final_nodes = int(input("Enter the final number of nodes: "))
    m_parameter = int(input("Enter the value of m parameter (m <= m_0): "))

    if m_parameter > init_nodes:
        print("Value of m parameter can't be greater than m0, please retry")
    else:
        print("\nCreating initial graph...")
        graph = {i: set(j for j in range(init_nodes) if j != i) for i in range(init_nodes)}
        print(f"Graph initialized with {len(graph)} nodes.")

        new_node = init_nodes
        print("\nAdding nodes...")
        for _ in tqdm(range(final_nodes - init_nodes)):
            graph[new_node] = set()
            for _ in range(m_parameter):

```

```
        add_edge_to_graph(graph, new_node, degree)
        new_node += 1

    print(f"\nFinal number of nodes reached: {len(graph)}")

    plot_degree_distribution(graph)
    plot_degree_distribution(graph, scale='log', fit_line=False)
    assess_topology(graph)

if __name__ == "__main__":
    degree=1
    barabasi_albert_simulation(1)
    barabasi_albert_simulation(2)
    barabasi_albert_simulation(3)
```

Creating initial graph...
Graph initialized with 12 nodes.

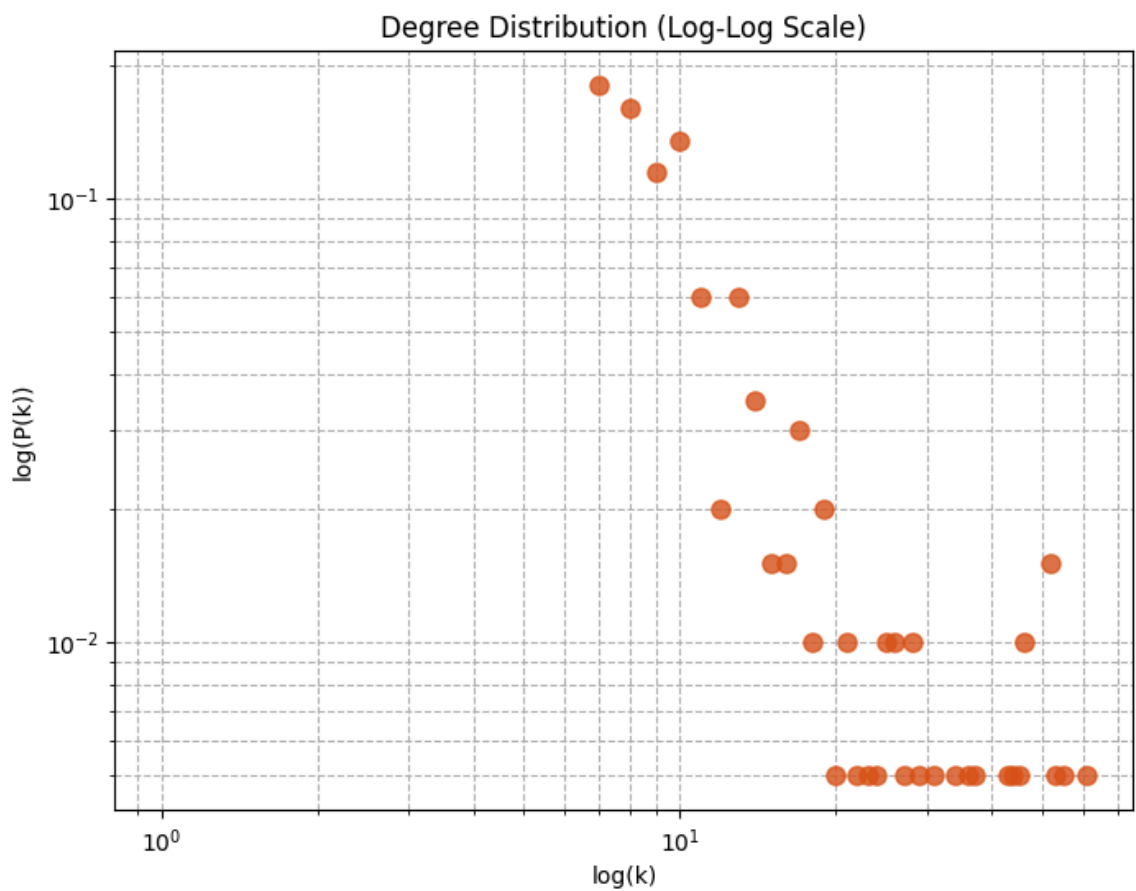
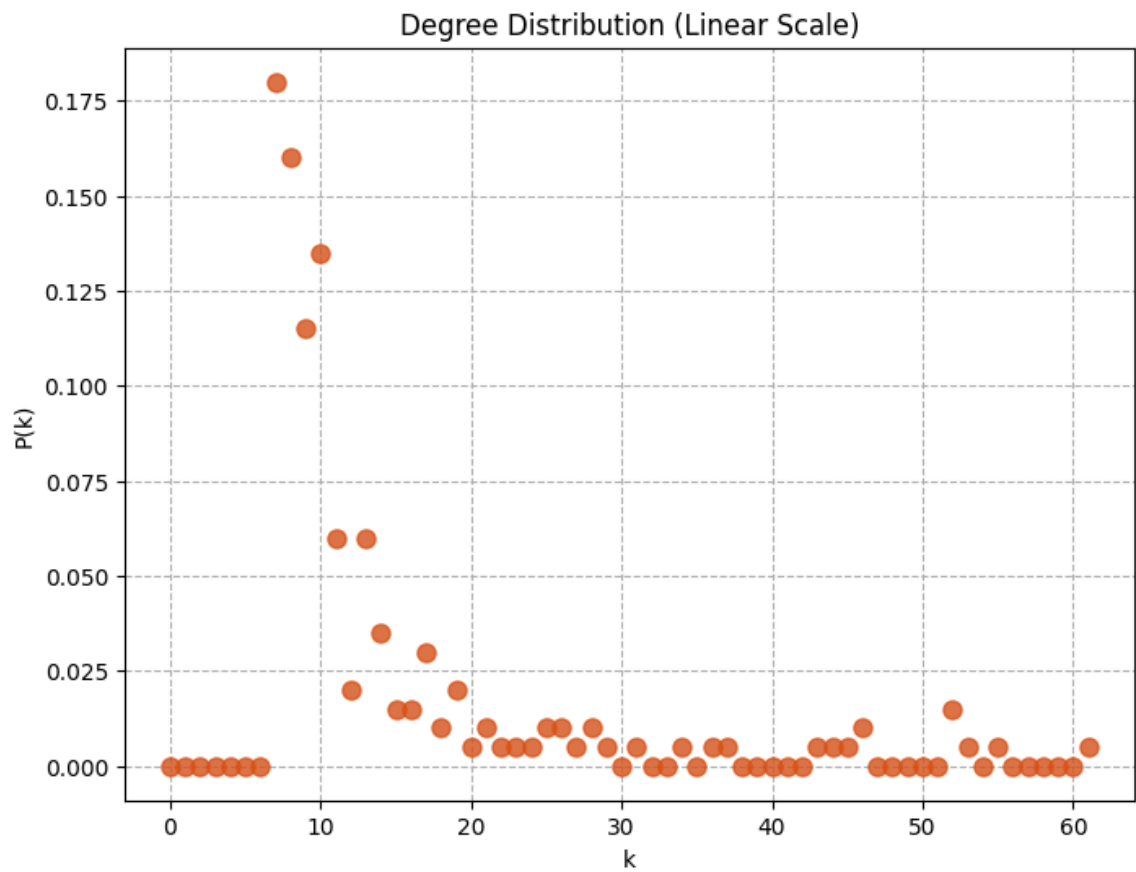
Adding nodes...

100%|██████████| 188/188 [00:00<00:00, 1257.48it/s]

[illegible]

[illegible]

```
Final number of nodes reached: 200
```



Average Clustering Coefficient: 0.1765
Characteristic Path Length: 2.2517

Creating initial graph...
Graph initialized with 12 nodes.

Adding nodes...

[illegible]

[illegible]

[illegible]

Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...

43%|██████████ | 80/188 [00:00<00:00, 187.27it/s]

[illegible]

Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...

65%|██████████ | 122/188 [00:01<00:00, 79.80it/s]

[illegible]

[illegible]

```
80%|██████████ | 151/188 [00:01<00:00, 52.24it/s]
```

[illegible]

84%|██████████ | 157/188 [00:02<00:00, 45.99it/s]

[illegible]

```
91%|██████████ | 171/188 [00:02<00:00, 32.54it/s]
```

[illegible]


```
95%|██████████| 179/188 [00:03<00:00, 27.69it/s]
```

[illegible]

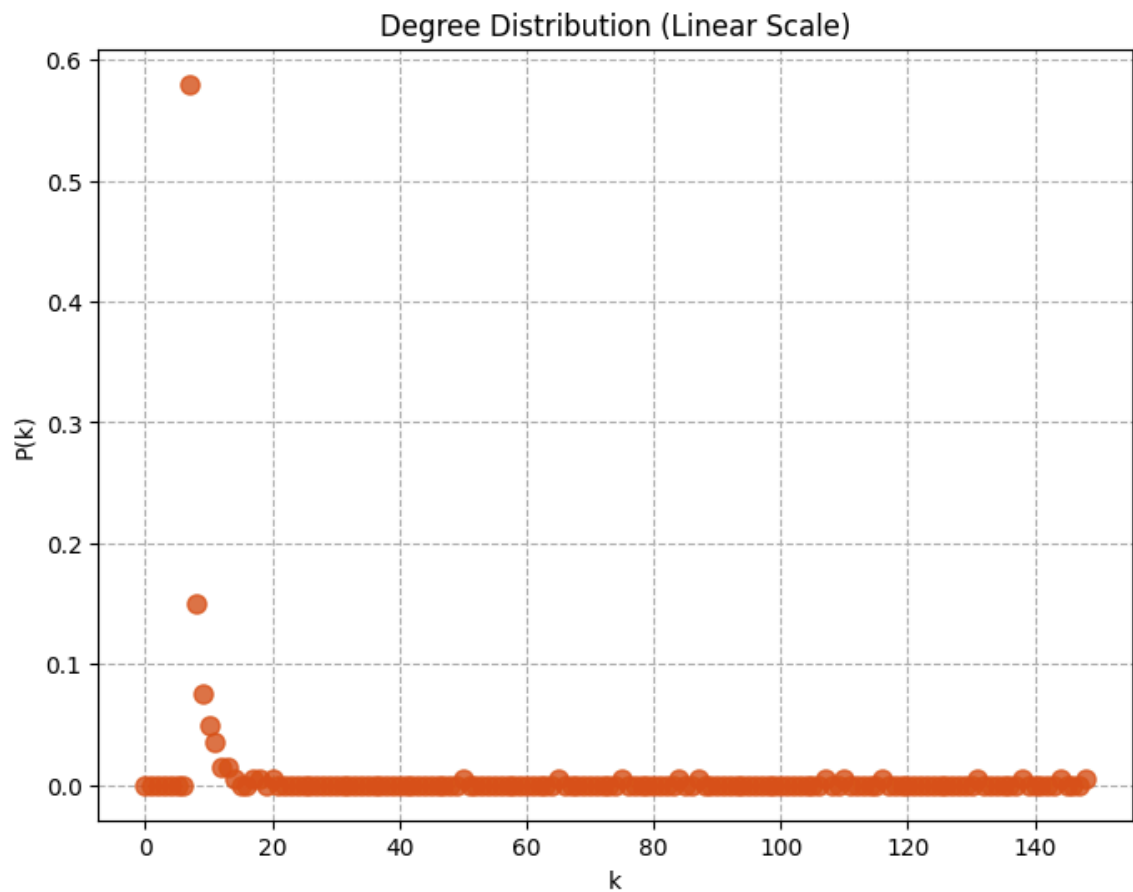
```
98%|██████████| 185/188 [00:03<00:00, 24.07it/s]
```

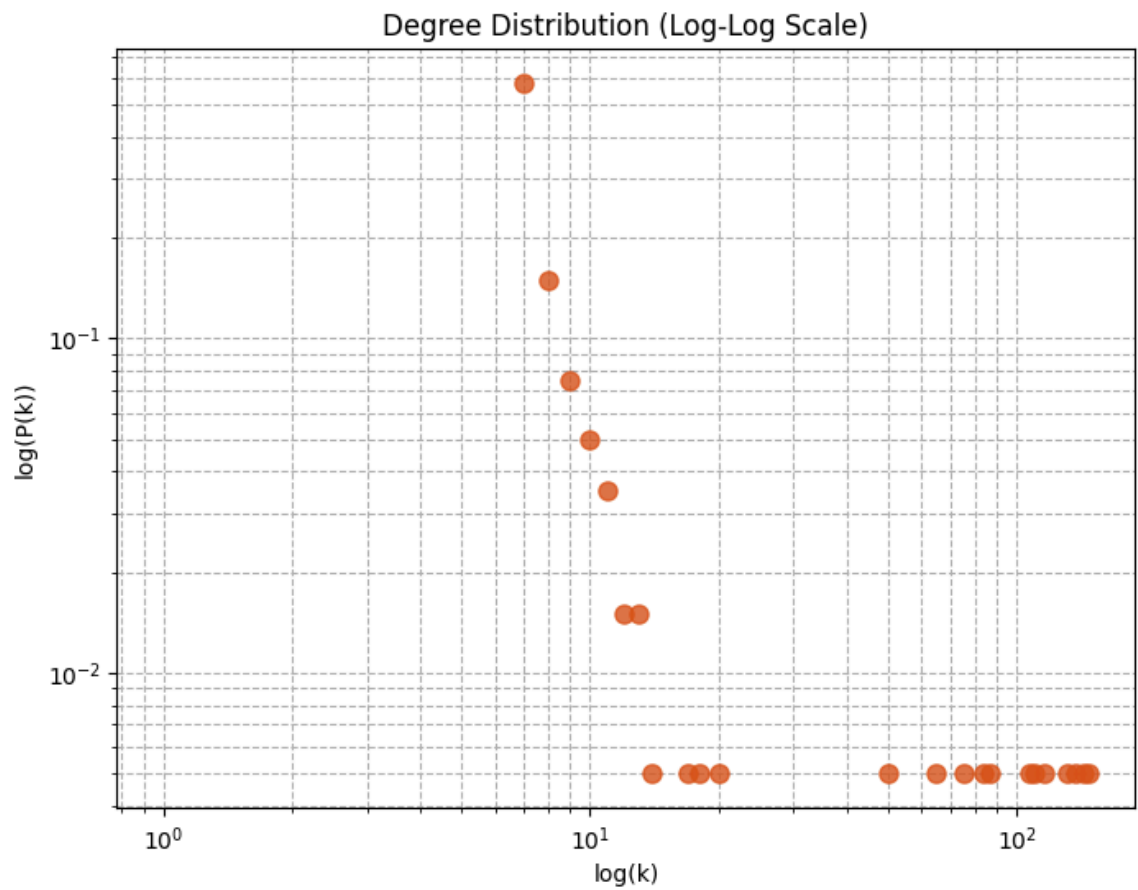
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...

100%|██████████| 188/188 [00:03<00:00, 52.70it/s]

Edge already exists! Retrying...
Edge already exists! Retrying...

Final number of nodes reached: 200





Average Clustering Coefficient: 0.7501

Characteristic Path Length: 1.9237

Creating initial graph...

Graph initialized with 12 nodes.

Adding nodes...

15% | 29/188 [00:00<00:00, 283.13it/s]

[illegible]

[illegible]

[illegible]

Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...

49%|██████████ | 93/188 [00:00<00:01, 81.78it/s]

[illegible]

[illegible]

[illegible]

Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...

60%|██████████ | 113/188 [00:01<00:01, 51.49it/s]

[illegible]

[illegible]

Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...

67%|██████████ | 126/188 [00:02<00:01, 37.81it/s]

[illegible]

[illegible]

```
70%|██████████| 131/188 [00:02<00:01, 31.55it/s]
```


[illegible]

Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...

74%|██████████ | 139/188 [00:02<00:01, 27.20it/s]

[illegible]

[illegible]

[illegible]

81% | ██████████ | 153/188 [00:03<00:01, 18.31it/s]

[illegible]

85%|██████████ | 160/188 [00:04<00:01, 16.15it/s]

[illegible]

```
86%|██████████| 162/188 [00:04<00:01, 15.40it/s]
```

[illegible]

```
89%|██████████| 168/188 [00:04<00:01, 14.48it/s]
```

Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...

90%|██████████ | 170/188 [00:04<00:01, 15.40it/s]

Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...

91%|██████████ | 172/188 [00:05<00:01, 11.51it/s]

[illegible]

```
98%|██████████| 184/188 [00:05<00:00, 12.43it/s]
```

[illegible]

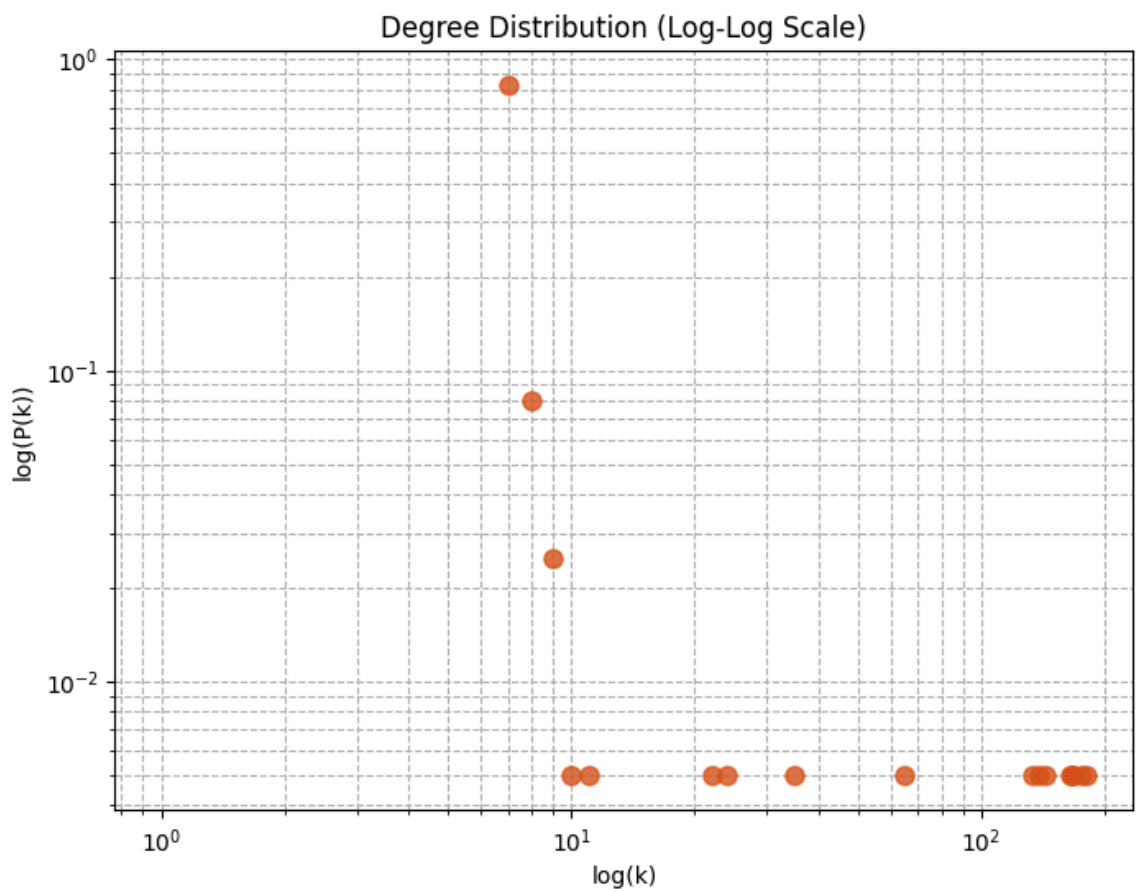
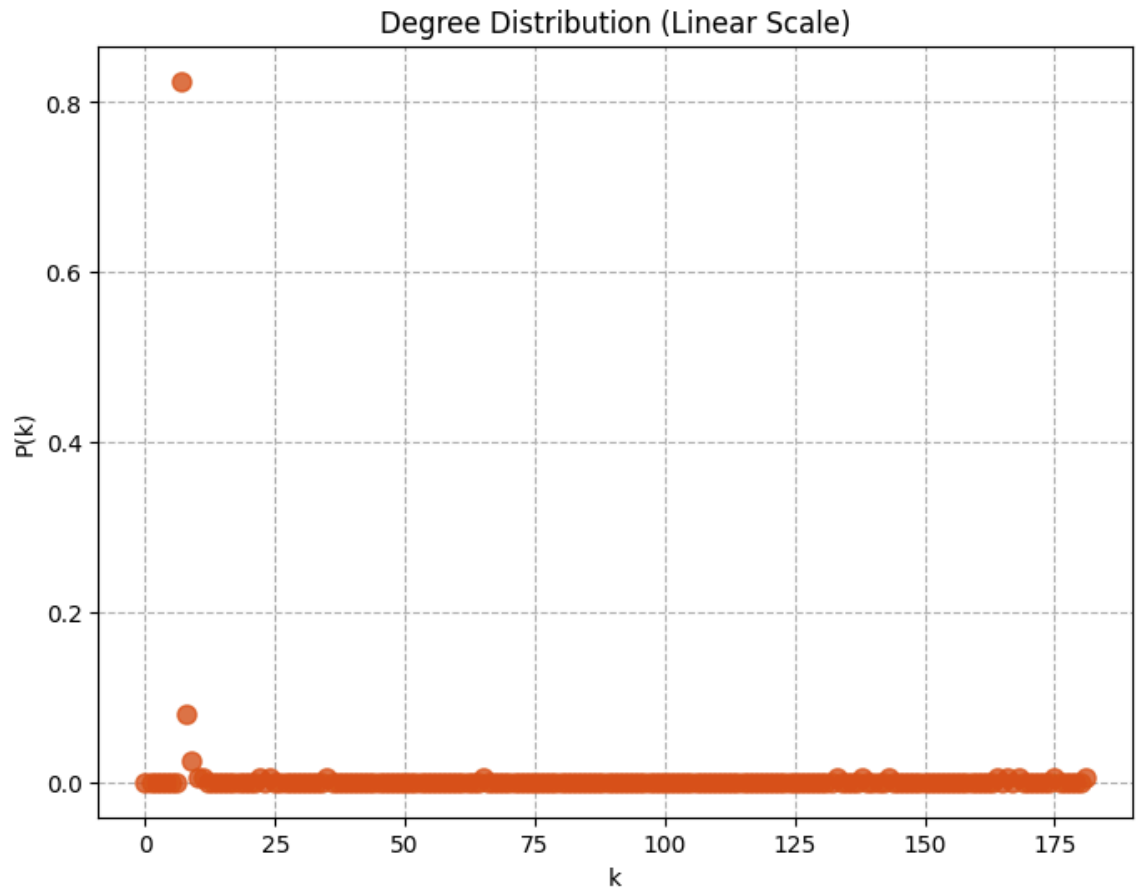
```
99%|██████████| 186/188 [00:06<00:00, 12.48it/s]
```

Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...

100%|██████████| 188/188 [00:06<00:00, 29.29it/s]

Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...
Edge already exists! Retrying...

Final number of nodes reached: 200



the coefficient is relatively low at 0.1765. This suggests that for nodes with a degree of 1, the graph has few tightly-knit clusters. In other words, most of the connections for these nodes do not form a strongly interconnected neighborhood. Characteristic Path Length: 2.2517 This metric reflects the average shortest path length between all pairs of nodes in the network. A characteristic path length of 2.2517 implies that the network is moderately connected. The paths between nodes are reasonably short on average, but there may be a larger number of intermediate nodes in some cases. This suggests that for nodes of degree 1, the network is not highly dense but still has fairly direct connections. For Degree = 2: Average Clustering Coefficient: 0.7501

A significant increase in the clustering coefficient to 0.7501 indicates that nodes with a degree of 2 have a much higher likelihood of forming tight clusters. These nodes are more connected within their neighborhoods, implying that they tend to form more triangles (three-way connections). This is expected in networks where nodes are more interconnected, contributing to a denser local structure. Characteristic Path Length: 1.9237

The path length decreases slightly to 1.9237 compared to degree=1. This shows that nodes with degree 2 are closer to each other on average. The shorter path lengths suggest that the network's overall connectivity has improved, making it more efficient for information to flow between nodes. For Degree = 3: Average Clustering Coefficient: 0.9165

The clustering coefficient continues to rise, reaching 0.9165. This high value indicates that nodes with a degree of 3 form very dense local neighborhoods with strong interconnections. There is a high probability that the neighbors of a node are also connected to each other, leading to tight-knit communities or cliques. Characteristic Path Length: 1.9210

The characteristic path length remains nearly constant (slightly lower) at 1.9210. Despite the increase in clustering, the average shortest path between nodes is still very short, indicating that the network is highly connected. The local clusters formed by nodes of degree 3 don't seem to increase the overall distance between nodes, which is a positive characteristic for communication efficiency in the network. General Observations:

Clustering Coefficient: The clustering coefficient increases significantly as the degree of the nodes increases. This shows that higher-degree nodes are more likely to form densely connected communities, which are characteristic of "small-world" networks where nodes tend to form tightly-knit local clusters.

Characteristic Path Length: The characteristic path length decreases slightly with increasing degree, suggesting that as nodes get more connected (higher degree), the overall efficiency of the network in terms of communication or information spread improves. This means that for nodes with higher degrees, the average number of intermediate nodes required to connect any two nodes is small, even though the network continues to grow in size.

In summary, as the degree increases from 1 to 3, the network becomes denser in terms of clustering, and it also becomes more efficient in terms of average path lengths. This

behavior reflects typical properties of scale-free networks, where a few highly connected nodes (hubs) create dense local structures, and the overall connectivity improves.