NAME : RAHINI DEVI S

ROLLNO: 225229129

# SMA CIA - 1 : TWITTER COMBINED DATASET

### Import the necessary package

In [1]:
```
import pandas as pd
```

### Load the file

In [2]:
```
data = pd.read_csv('twitter_combined.txt')
data
```

Out[2]:

| | StartNode EndNode |
|---|---|
| **0** | 17116707 28465635 |
| **1** | 380580781 18996905 |
| **2** | 221036078 153460275 |
| **3** | 107830991 17868918 |
| **4** | 151338729 222261763 |
| **...** | ... |
| **2420760** | 99841247 154263215 |
| **2420761** | 99841247 194403468 |
| **2420762** | 99841247 180165101 |
| **2420763** | 99841247 253509115 |
| **2420764** | 99841247 463410501 |

2420765 rows × 1 columns

### Calculate the number of rows to keep (e.g., 50%)

In [3]:
```
file = int(len(data) * 0.4)
```

### Truncate the DataFrame

In [4]:
```
truncated_data = data[:file]
truncated_data
```

Out[4]:

| | StartNode EndNode |
|---|---|
| **0** | 17116707 28465635 |
| **1** | 380580781 18996905 |
| **2** | 221036078 153460275 |
| **3** | 107830991 17868918 |
| **4** | 151338729 222261763 |
| **...** | ... |
| **968301** | 276706356 83943787 |
| **968302** | 439788025 302847930 |
| **968303** | 194403468 18996905 |
| **968304** | 124810771 34428380 |
| **968305** | 241159821 240905997 |

968306 rows × 1 columns

### Calculate the Average Path Length

In [5]:
```
import networkx as nx
```

In [6]:
```
# Load the Twitter network from the file

G = nx.read_edgelist(truncated_data)
```

In [7]:
```
# Get the connected components

components = nx.connected_components(G)
```

In [8]:
```
# Calculate the average path length for each component

avg_lengths = []
for component in components:
    subgraph = G.subgraph(component)
    avg_lengths.append(nx.average_shortest_path_length(subgraph))
```

In [9]:
```
# Calculate the overall average path length

avg_path_length = sum(avg_lengths) / len(avg_lengths)

print(f"The average path length of the Twitter network is: {avg_path_length}")
```

```
The average path length of the Twitter network is: 1.0
```

### Calculate Diameter

In [10]:
```
# Calculate the diameter

diameter = nx.diameter(G)
```

In [11]:
```
print(f"The diameter of the Twitter network is: {diameter}")
```

```
The diameter of the Twitter network is: 1
```

### Calculate centrality measures

In [12]:
```
# Calculate degree centrality

degree_centrality = nx.degree_centrality(G)
```

In [13]:
```
# Calculate eigenvector centrality

eigenvector_centrality = nx.eigenvector_centrality(G)
```

In [14]:
```
# Print the centrality measures for the first 10 nodes

for node in list(G.nodes())[:10]:
    print(f"Node {node}: Degree Centrality = {degree_centrality[node]}, Eigenv
```

```
Node StartNode: Degree Centrality = 1.0, Eigenvector Centrality = 0.707106781
1865476
Node EndNode: Degree Centrality = 1.0, Eigenvector Centrality = 0.70710678118
65476
```

In [15]:
```
!pip uninstall community
```

```
WARNING: Skipping community as it is not installed.
```

In [16]:
```
!pip install python-louvain
```

```
Requirement already satisfied: python-louvain in c:\users\visit\anaconda3\lib
\site-packages (0.16)
Requirement already satisfied: networkx in c:\users\visit\anaconda3\lib\site-
packages (from python-louvain) (2.8.4)
Requirement already satisfied: numpy in c:\users\visit\anaconda3\lib\site-pac
kages (from python-louvain) (1.24.3)
```

In [17]:
```
import community
```

In [18]:
```
# Find communities using the Louvain algorithm

communities = community.best_partition(G)
```

In [19]:
```
# Print the communities

for node, community_id in communities.items():
    print(f"Node {node}: Community {community_id}")
```

```
Node StartNode: Community 0
Node EndNode: Community 0
```