import networkx as nx  
  
# Generate a connected network with N nodes and S edges  
network = nx.gnm\_random\_graph(900, 12000)  
while not nx.is\_connected(network):  
 network = nx.gnm\_random\_graph(900, 12000)  
  
# Remove a certain percentage of edges  
removed\_edges = int(0.0 \* network.number\_of\_edges())  
edges\_to\_remove = list(network.edges())[:removed\_edges]  
network.remove\_edges\_from(edges\_to\_remove)  
  
# Calculate CC  
clustering\_coefficient = nx.average\_clustering(network)  
  
# Calculate RSLCC  
largest\_connected\_component = max(nx.connected\_components(network), key=len)  
relative\_size = len(largest\_connected\_component) / len(network)  
  
# Calculate INR  
isolated\_nodes = list(nx.isolates(network))  
isolated\_nodes\_ratio = len(isolated\_nodes) / len(network)  
  
# Calculate NE  
efficiency = nx.global\_efficiency(network)  
  
# Print Results  
print("Network Clustering Coefficient:", clustering\_coefficient)  
print("Largest Connected Component Relative Size:", relative\_size)  
print("Isolated Nodes Ratio:", isolated\_nodes\_ratio)  
print("Network Efficiency:", efficiency)