NetworkX graph generator from DF:

```
# VER 2:
pip install networkx matplotlib

import pandas as pd
import networkx as nx
import matplotlib.pyplot as plt

# Create a DataFrame with edge data
data = {
    'from': ['A', 'A', 'B', 'C'],
    'to': ['B', 'C', 'C', 'D']
}
df = pd.DataFrame(data)

# Create a graph object
G = nx.Graph()

# Add edges from the DataFrame to the graph
for index, row in df.iterrows():
    G.add_edge(row['from'], row['to'])

# Draw the graph
nx.draw(G, with_labels=True, node_color='lightblue', edge_color='gray', node_size=2000,
font_size=15)

# Show the plot
plt.show()
```

NetworkX graph generator from DF Ver.2 w/ print coords:

```
pip install networkx matplotlib

import pandas as pd
import networkx as nx
import matplotlib.pyplot as plt

# Create a DataFrame with edge data
data = {
    'from': ['A', 'A', 'B', 'C'],
    'to': ['B', 'C', 'C', 'D']
}
df = pd.DataFrame(data)

# Create a graph object
G = nx.Graph()

# Add edges from the DataFrame to the graph
for index, row in df.iterrows():
    G.add_edge(row['from'], row['to'])

# Compute the positions of the nodes using a layout function
positions = nx.spring_layout(G)

# Print the positions of each node
print("Node positions:")
for node, pos in positions.items():
    print(f"{node}: {pos}")
```

```

# Draw the graph
nx.draw(G, pos=positions, with_labels=True, node_color='lightblue', edge_color='gray',
node_size=2000, font_size=15)

# Show the plot
plt.show()

# - - - USE XY TO LINE ArcGIS TOOL TO CREATE GEOMETRY
```

NetworkX graph generator from DF Ver.3 w/ add coords to new dataframe:

```
import pandas as pd
import networkx as nx
import matplotlib.pyplot as plt

# Create a DataFrame with edge data
data = {
    'from': ['A', 'A', 'B', 'C'],
    'to': ['B', 'C', 'C', 'D']
}
df = pd.DataFrame(data)

# Create a graph object
G = nx.Graph()

# Add edges from the DataFrame to the graph
for index, row in df.iterrows():
    G.add_edge(row['from'], row['to'])

# Compute the positions of the nodes using a layout function
positions = nx.spring_layout(G)

# Create a DataFrame to store node positions
positions_df = pd.DataFrame(positions).T.reset_index()
positions_df.columns = ['Node', 'X', 'Y']

# Print the DataFrame with node positions
print("Node positions DataFrame:")
print(positions_df)

# Draw the graph
nx.draw(G, pos=positions, with_labels=True, node_color='lightblue', edge_color='gray',
node_size=2000, font_size=15)

# Show the plot
plt.show()
```

Code sample XYToLine example (stand-alone script) (Link)
https://doc.arcgis.com/en/allsource/latest/analysis/geoprocessing-tools/data-management/xy-to-line.htm

```
# Import system modules
import arcpy
from arcpy import env

# Set local variables
input_table = r"c:\workspace\city2city.dbf"
out_lines = r"c:\workspace\flt4421.gdb\routing001"

```

```
#XY To Line
arcpy.XYToLine_management(input_table,out_lines,
                          "LOND1","LATD1","LOND2",
                          "LATD2","GEODESIC","idnum")
```