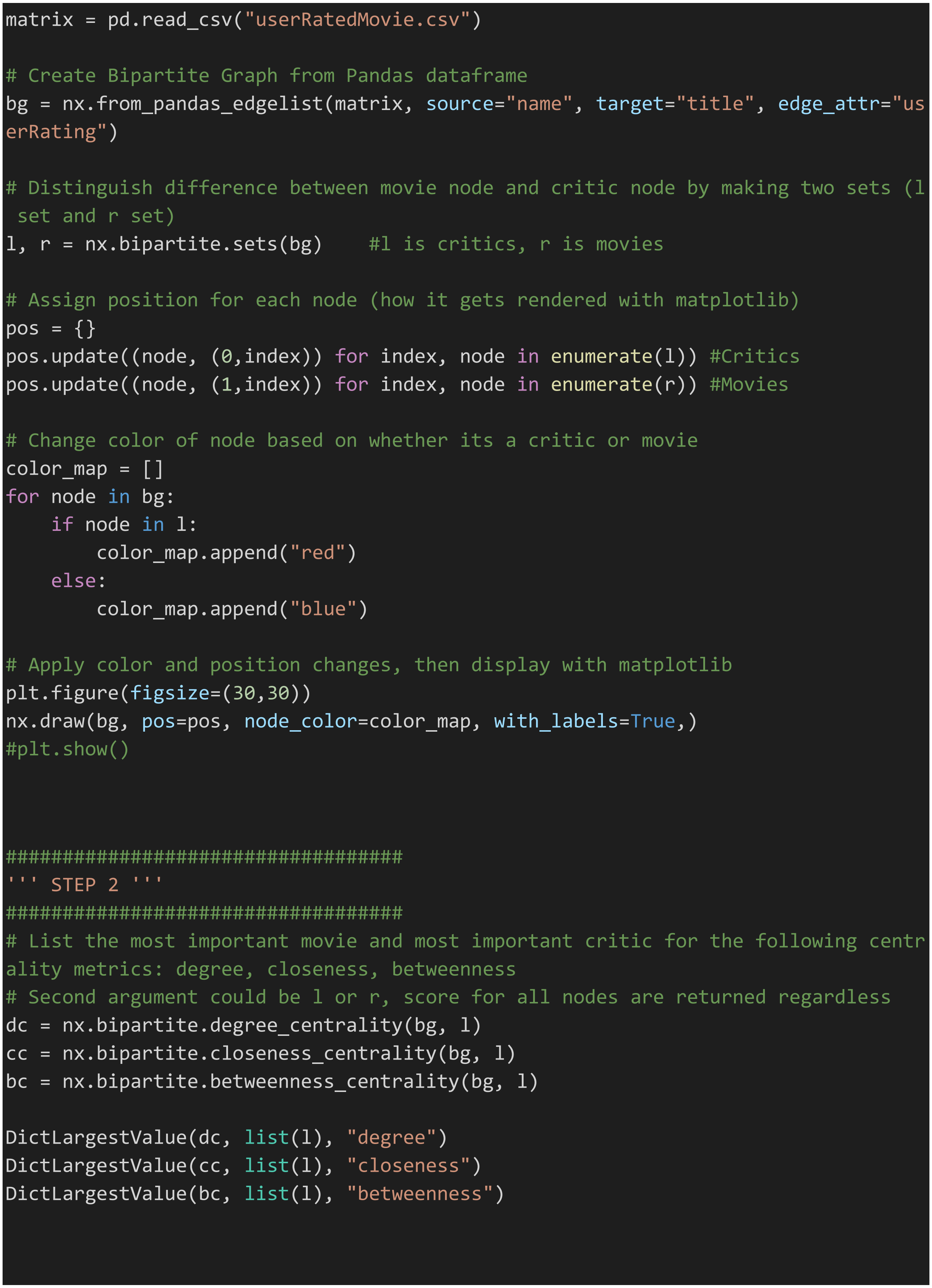
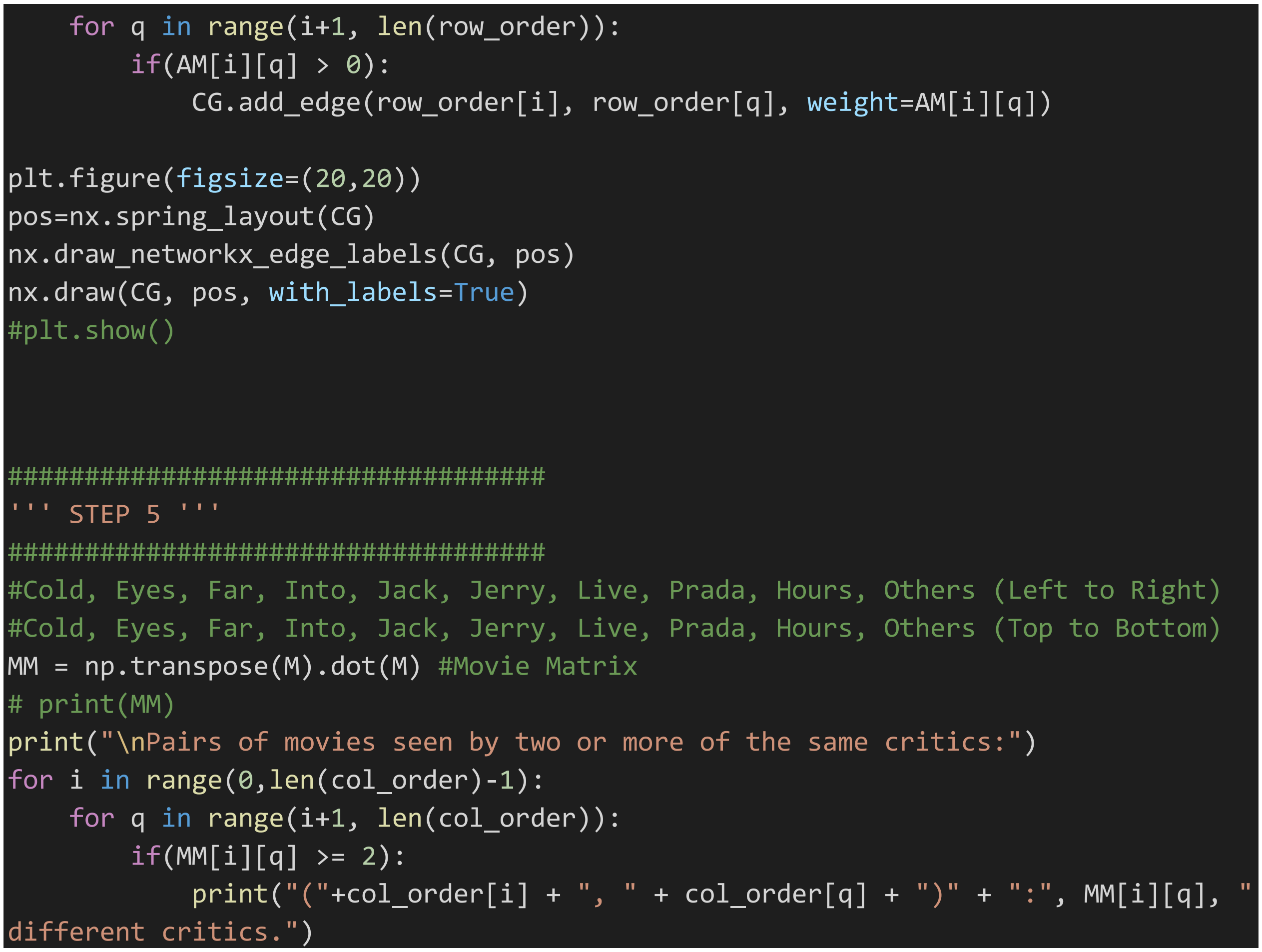
```python
import pandas as pd
import numpy as np
import networkx as nx
from networkx.algorithms.bipartite import sets, weighted_projected_graph
import matplotlib.pyplot as plt
from networkx.algorithms import bipartite


# Function that prints the key with the largest numeric value in a dict
def DictLargestValue(dictionary, critic_list, centrality_type):
    topMovieKey = ""
    largestMovieVal = -1

    topCriticKey = ""
    largestCriticVal = -1


    for key in dictionary.keys():
        if(key in critic_list):                     #This if statement takes care of the critics
            if(dictionary[key] > largestCriticVal):
                largestCriticVal = dictionary[key]
                topCriticKey = key
        else:                                       #This else statement takes care of the movies
            if(dictionary[key] > largestMovieVal):
                largestMovieVal = dictionary[key]
                topMovieKey = key

        #print(dictionary[key], key)

    print("Top critic for", centrality_type, "centrality is:", topCriticKey, "with score of", largestCriticVal)
    print("Top movie for", centrality_type, "centrality is:", topMovieKey, "with score of", largestMovieVal)
    print()






###################################
''' STEP 1 '''
###################################
# Read in the data
```



```python
matrix = pd.read_csv("userRatedMovie.csv")

# Create Bipartite Graph from Pandas dataframe
bg = nx.from_pandas_edgelist(matrix, source="name", target="title", edge_attr="userRating")

# Distinguish difference between movie node and critic node by making two sets (l set and r set)
l, r = nx.bipartite.sets(bg)    #l is critics, r is movies

# Assign position for each node (how it gets rendered with matplotlib)
pos = {}
pos.update((node, (0,index)) for index, node in enumerate(l)) #Critics
pos.update((node, (1,index)) for index, node in enumerate(r)) #Movies

# Change color of node based on whether its a critic or movie
color_map = []
for node in bg:
    if node in l:
        color_map.append("red")
    else:
        color_map.append("blue")

# Apply color and position changes, then display with matplotlib
plt.figure(figsize=(30,30))
nx.draw(bg, pos=pos, node_color=color_map, with_labels=True,)
#plt.show()



###################################
''' STEP 2 '''
###################################
# List the most important movie and most important critic for the following centrality metrics: degree, closeness, betweenness
# Second argument could be l or r, score for all nodes are returned regardless
dc = nx.bipartite.degree_centrality(bg, l)
cc = nx.bipartite.closeness_centrality(bg, l)
bc = nx.bipartite.betweenness_centrality(bg, l)

DictLargestValue(dc, list(l), "degree")
DictLargestValue(cc, list(l), "closeness")
DictLargestValue(bc, list(l), "betweenness")
```

```
###################################
''' STEP 3 '''
###################################
# Make a bipartite matrix
row_order = sorted(list(l)) #Rows are critics
col_order = sorted(list(r)) #Cols are movies
numpyMatrix = bipartite.biadjacency_matrix(bg, row_order, column_order=col_order)

# Create an event by actor matrix to determine movies that have been seen by 3 or more critics
M = numpyMatrix.A   #.A gets us an ndarray object
#print(row_order)   #Edna, Homer, Krusty, Lisa, Marge, Moe, Ned                 (Top to bottom)
#print(col_order)   #Cold, Eyes, Far, Into, Jack, Jerry, Live, Prada, Hours, Others (Left to Right)

timesViewed = 0 #Increment number of times a movie has been seen for each movie. Reset to zero for each movie.
print("Movies seen by three or more critics: ")
for i in range(len(col_order)):
    timesViewed = 0
    for q in range(len(row_order)):
        timesViewed = timesViewed + M[q][i]

    #Output the name of the film if it has been viewed by three or more critics.
    if(timesViewed >= 3):
        print(col_order[i], "has been viewed by", timesViewed, "critics.")



###################################
''' STEP 4 '''
###################################
# Make an undirected graph of the critics. Edges between critics have a number attribute that represents the # of movies seen in common
# This graph will be made by taking values from an actor-to-actor matrix

#Edna, Homer, Krusty, Lisa, Marge, Moe, Ned         (Top to bottom)
#Edna, Homer, Krusty, Lisa, Marge, Moe, Ned         (Left to Right)
AM = M.dot(np.transpose(M))

CG = nx.Graph()
# Compute number of similar movies seen between critics by looking at actor-to-actor matrix (upper right triangle of matrix)
for i in range(0,len(row_order)-1):
```



```
    for q in range(i+1, len(row_order)):
        if(AM[i][q] > 0):
            CG.add_edge(row_order[i], row_order[q], weight=AM[i][q])

plt.figure(figsize=(20,20))
pos=nx.spring_layout(CG)
nx.draw_networkx_edge_labels(CG, pos)
nx.draw(CG, pos, with_labels=True)
#plt.show()



###################################
''' STEP 5 '''
###################################
#Cold, Eyes, Far, Into, Jack, Jerry, Live, Prada, Hours, Others (Left to Right)
#Cold, Eyes, Far, Into, Jack, Jerry, Live, Prada, Hours, Others (Top to Bottom)
MM = np.transpose(M).dot(M) #Movie Matrix
# print(MM)
print("\nPairs of movies seen by two or more of the same critics:")
for i in range(0,len(col_order)-1):
    for q in range(i+1, len(col_order)):
        if(MM[i][q] >= 2):
            print("("+col_order[i] + ", " + col_order[q] + ")" + ":", MM[i][q], "
different critics.")
```

Program Output:

Centrality Measures

```
Top critic for degree centrality is: Marge Simpson with score of 0.6000000000000001

Top movie for degree centrality is: Cold Mountain with score of 0.42857142857142855


Top critic for closeness centrality is: Marge Simpson with score of 0.6470588235294118

Top movie for closeness centrality is: Into the Woods with score of 0.6756756756756757


Top critic for betweenness centrality is: Marge Simpson with score of 0.3364035087719298

Top movie for betweenness centrality is: Cold Mountain with score of 0.19469928644240572
```

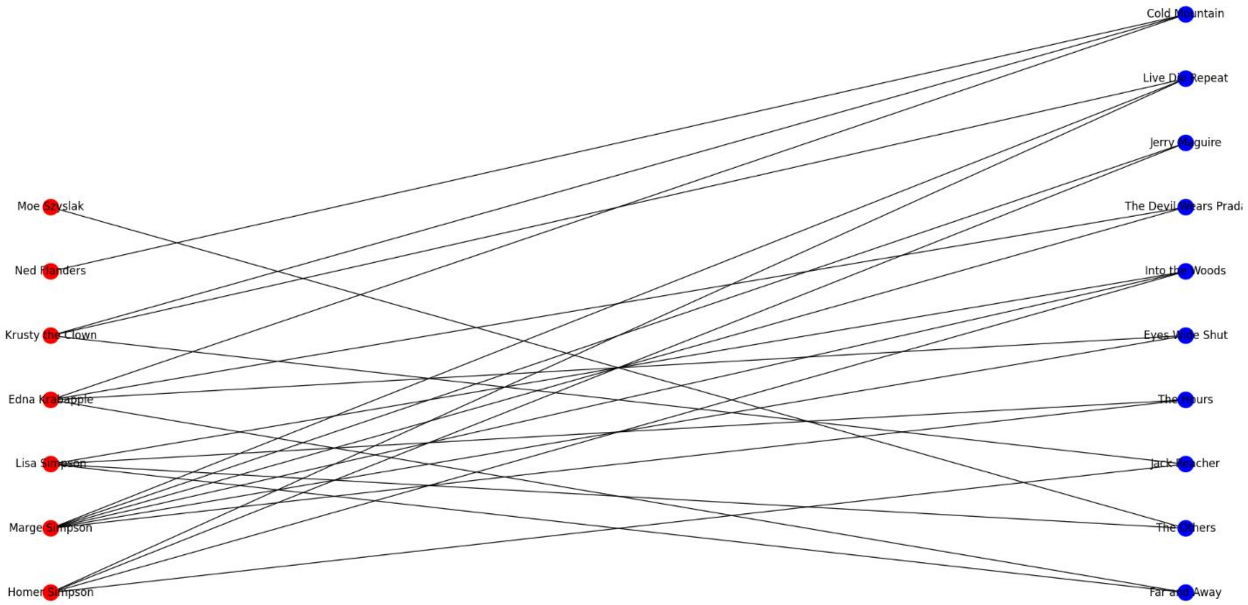
Movies seen by three or more critics:

```
Cold Mountain has been viewed by 3 critics.

Into the Woods has been viewed by 3 critics.

Live Die Repeat has been viewed by 3 critics.
```

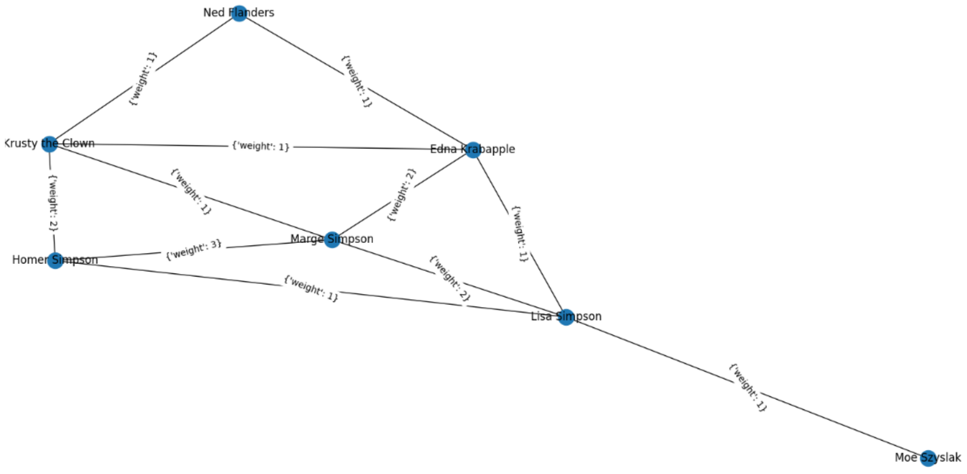
Pairs of movies seen by two or more of the same critics:

```
(Eyes Wide Shut, The Devil Wears Prada): 2 different critics.

(Into the Woods, Jerry Maguire): 2 different critics.

(Into the Woods, Live Die Repeat): 2 different critics.

(Into the Woods, The Hours): 2 different critics.

(Jack Reacher, Live Die Repeat): 2 different critics.

(Jerry Maguire, Live Die Repeat): 2 different critics.
```

Bipartite Graph Below:



Watched Movies in Common Below:


Ned Flanders
{'weight': 1}
{'weight': 1}
Krusty the Clown
{'weight': 1}
Edna Krabapple
{'weight': 1}
{'weight': 2}
{'weight': 2}
{'weight': 1}
Marge Simpson
Homer Simpson
{'weight': 3}
{'weight': 2}
{'weight': 1}
Lisa Simpson
{'weight': 1}
Moe Szyslak