GRAPH ALGORITHMS

ASSIGNMENT 01 – NetworkX Python

MUHAMMAD HARRIS
BCS203193

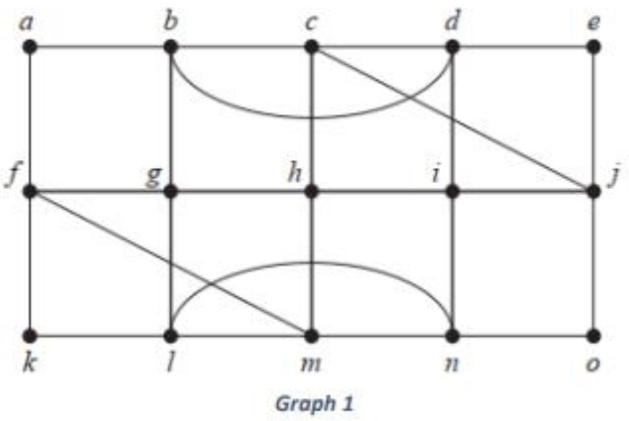
Table of Contents

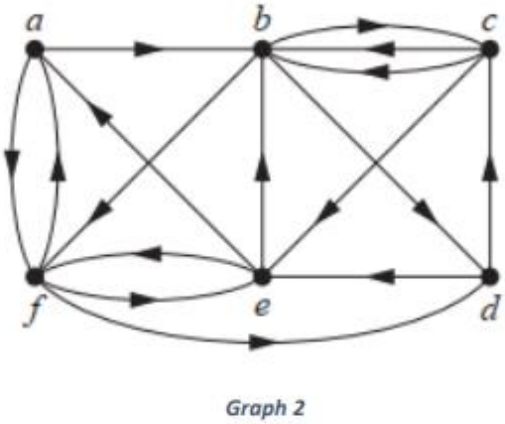
GRAPH ALGORITHMS

ASSIGNMENT 01 – NetworkX Python

QUESTION:

For the given directed and undirected graphs:





Write the PYTHON program for each graph in which:

1) Display Nodes List of the Graph
2) Display Edge List of the Graph
3) Count Connected Components of the Graph
4) Print Connected Components of the Graph
5) Display Incidence Matrix of a Graph
6) Display the Nodes degrees
7) Count Number of Edges
8) Visualize the graphs
9) Check if:
   a) Euler circuit exist or not and Graph is Eulerian
   b) Euler path exists or not
   c) Hamilton Path
   d) Perform Depth First and Breadth First Traversal on graph

Coding Standard to Follow:

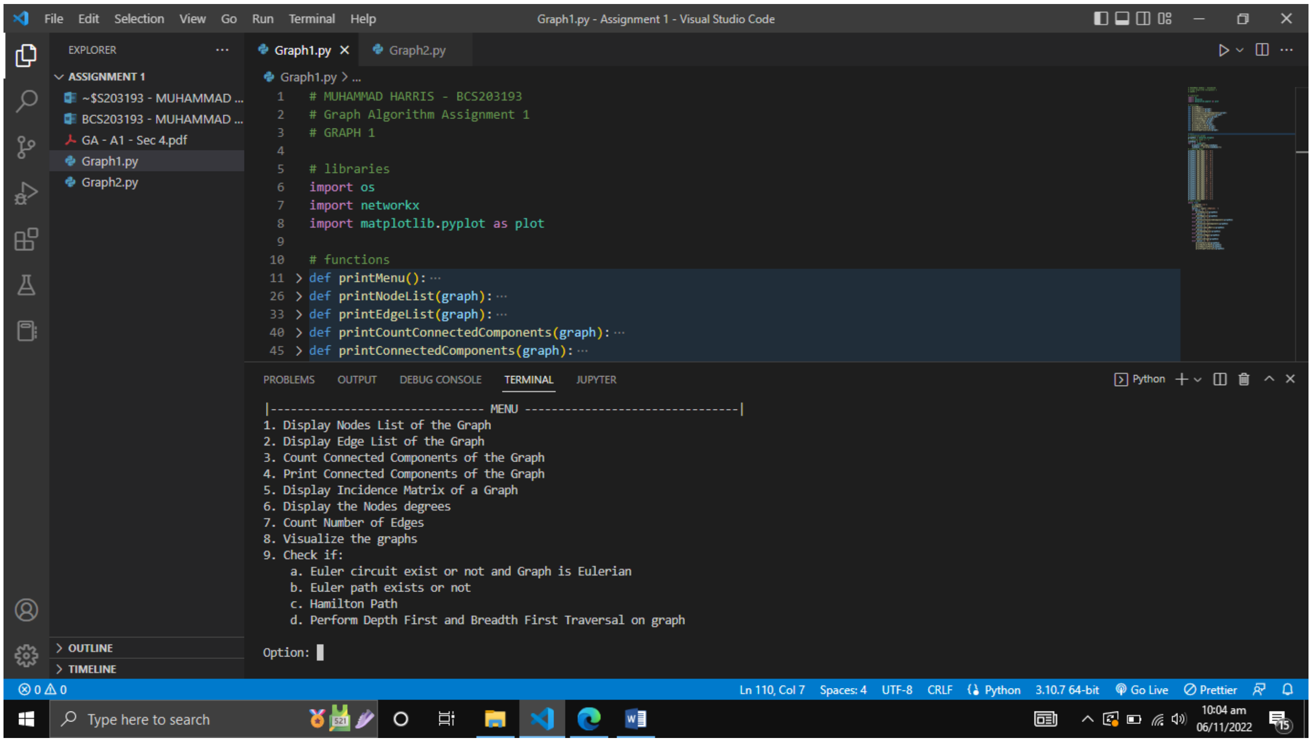
- For each graph create different Python Script (.py) will all the functionalities to implement
- There should be proper menu for user to perform the task
- Use proper comments in code for understanding
- Print output of each task with proper messages

GRAPH 1:

Code of Graph1.py

```
# MUHAMMAD HARRIS - BCS203193
# Graph Algorithm Assignment 1
# GRAPH 1

# libraries
import os
import networkx
import matplotlib.pyplot as plot

# functions
def printMenu():
    print("""|-------------------------------- MENU --------------------------------|
1. Display Nodes List of the Graph
2. Display Edge List of the Graph
3. Count Connected Components of the Graph
4. Print Connected Components of the Graph
5. Display Incidence Matrix of a Graph
6. Display the Nodes degrees
7. Count Number of Edges
8. Visualize the graphs
9. Check if:
    a. Euler circuit exist or not and Graph is Eulerian
    b. Euler path exists or not
    c. Hamilton Path
    d. Perform Depth First and Breadth First Traversal on graph""")
def printNodeList(graph):
    os.system('cls')
    nodeList = graph.nodes() # get node list of graph
    print('Node List of Graph:')
    for x in nodeList:
        print(x, end=', ') # print each node
    input('\n\npress enter to return...')
def printEdgeList(graph):
    os.system('cls')
    edgeList = graph.edges() # get edge list of graph
    print('Edge List of Graph:')
    for x in edgeList:
        print(x[0], '<->', x[1], end=', ') # print each edge
    input('\n\npress enter to return...')
def printCountConnectedComponents(graph):
    os.system('cls')
    count = networkx.number_connected_components(graph) # get number of connected components
    print('Number of Connected Components in Graph: ', count) # print count
    input('\npress enter to return...')
def printConnectedComponents(graph):
    os.system('cls')
    connectedComponents = [len(c) for c in sorted(networkx.connected_components(graph),
key=len, reverse=True)] # get connected components
    print('Length of Connected Components in Sorted Order:')
    for x in connectedComponents:
        print(x) # print length of each connected component
    input('\npress enter to return...')
def printIncidenceMatrix(graph):
    os.system('cls')
    incidenceMatrix = networkx.to_numpy_matrix(graph) # get incidence matix
    incidenceMatrixString =
str(incidenceMatrix).replace('[','').replace(']','').replace('.','')
    print('Incidence Matrix of Graph: ')
    print('',incidenceMatrixString) # print incidence matrix
    input('\npress enter to return...')
def printNodeDegrees(graph):
    os.system('cls')
    print('Degree of all Nodes in Graph:')
    nodeN = 'a'
    for x in range(15):
        print(nodeN, ' = ', graph.degree(nodeN)) # print degree of each node
        nodeN = chr(ord(nodeN)+1)
    input('\npress enter to return...')
def printCountEdges(graph):
    os.system('cls')
    print('Number of Edges in Graph: ', networkx.number_of_edges(graph)) # print number
of edges in graph
    input('\npress enter to return...')
def visualizeGraph(graph):
    os.system('cls')
    networkx.draw(graph, with_labels = True) # draw graph with labels
    plot.show()
    input('press enter to return...')
def printIsEulerian(graph):
    os.system('cls')
    if networkx.is_eulerian(graph): # check if graph is euler or not
        print('a. Graph is Euler Graph.')
    else:
        print('a. Graph is not Euler.')
def printHasEulerPath(graph):
    if networkx.has_eulerian_path(graph): # check if graph has euler path or not
        print('b. Eulerian Path or Circuit Exists.')
    else:
        print('b. Eulerian Path or Circuit does not Exist.')
def printHamiltonPath(graph):
    try:
        hamiltonPath = 
networkx.algorithms.tournament.hamiltonian_path(networkx.DiGraph(graph)) # get hamilton 
path
        print('c. Hamilton Path: ', )
        print('\t', end='')
        for x in hamiltonPath:
            if x != 'a':
                print(' ->', end=' ')
            print(x, end='') # print hamilton path if exists
    except:
        print('c. Hamilton Path does not exist.') # else print exception
def printGraphTraversals(graph):
    print('\nd. Graph Traversals (source node = a)')
    print('Breath First Search: visualized using matplotlib')
    BFStree = networkx.bfs_tree(graph, 'a') # create BFS tree
    networkx.draw(BFStree, with_labels=True)
    plot.show() # visualize BFS tree
    print('Depth First Search: visualized using matlplotlib')
    DFStree = networkx.dfs_tree(graph, 'a') # create DFS tree
    networkx.draw(DFStree, with_labels=True)
    plot.show() # visualize DFS tree
    input('\npress enter key to return...')

# main
# initialize graph
graphOne = networkx.Graph()
# creating nodes a - o
nodeName = 'a'
for x in range(15):
    graphOne.add_node(nodeName)
    nodeName = chr(ord(nodeName)+1)
# creating edges
graphOne.add_edge('a','b')
graphOne.add_edge('a','f')
graphOne.add_edge('b','g')
graphOne.add_edge('b','c')
graphOne.add_edge('b','d')
graphOne.add_edge('c','d')
graphOne.add_edge('c','j')
graphOne.add_edge('c','h')
graphOne.add_edge('d','e')
graphOne.add_edge('d','i')
graphOne.add_edge('e','j')
graphOne.add_edge('f','g')
graphOne.add_edge('f','k')
graphOne.add_edge('f','m')
graphOne.add_edge('g','h')
graphOne.add_edge('g','l')
graphOne.add_edge('h','i')
graphOne.add_edge('h','m')
graphOne.add_edge('i','j')
graphOne.add_edge('i','n')
graphOne.add_edge('j','o')
graphOne.add_edge('k','l')
graphOne.add_edge('l','m')
graphOne.add_edge('l','n')
graphOne.add_edge('m','n')
graphOne.add_edge('n','o')
# menu system
while True:
    os.system('cls')
    printMenu()
    option = input('\nOption: ')
    if option == '1':
        printNodeList(graphOne)
    elif option == '2':
        printEdgeList(graphOne)
    elif option == '3':
        printCountConnectedComponents(graphOne)
    elif option == '4':
        printConnectedComponents(graphOne)
    elif option == '5':
        printIncidenceMatrix(graphOne)
    elif option == '6':
        printNodeDegrees(graphOne)
    elif option == '7':
        printCountEdges(graphOne)
    elif option == '8':
        visualizeGraph(graphOne)
    elif option == '9':
        printIsEulerian(graphOne)
        printHasEulerPath(graphOne)
        printHamiltonPath(graphOne)
        printGraphTraversals(graphOne)
```

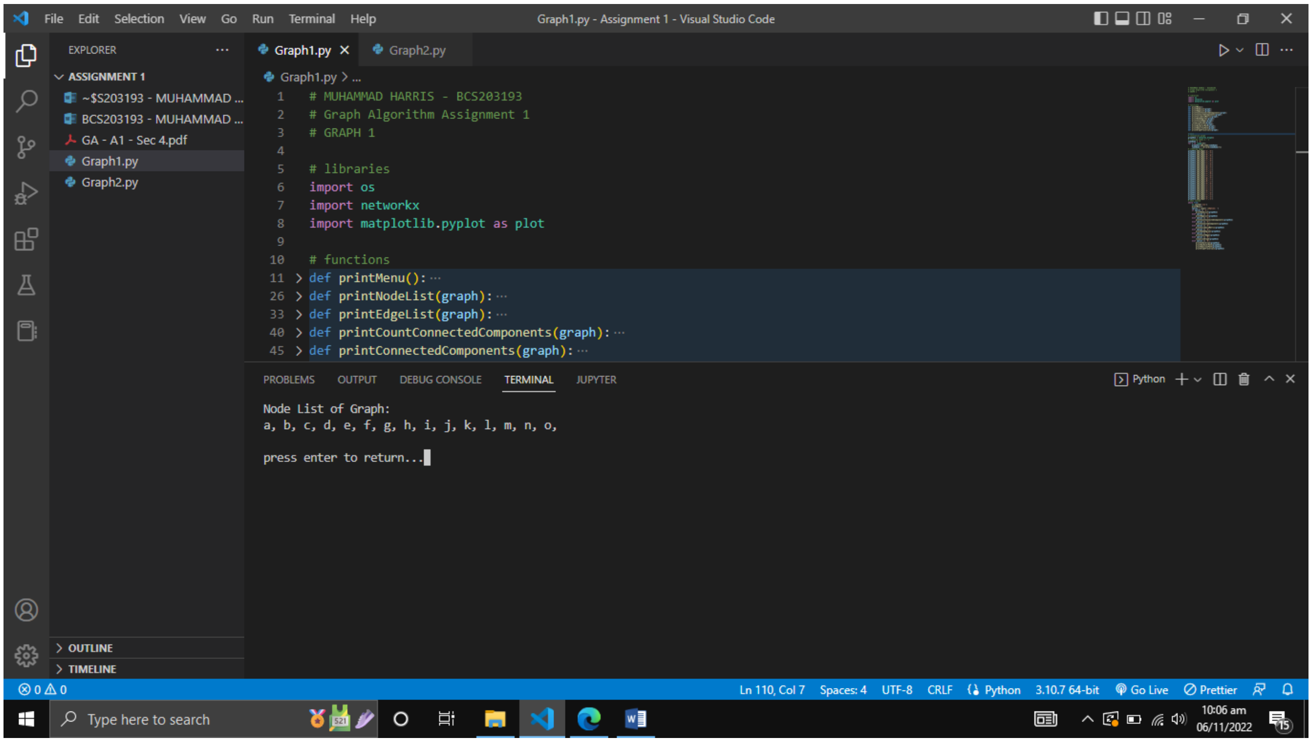
Outputs of Graph1.py


Graph1.py - Assignment 1 - Visual Studio Code
EXPLORER
ASSIGNMENT 1
~$S203193 - MUHAMMAD ...
BCS203193 - MUHAMMAD ...
GA - A1 - Sec 4.pdf
Graph1.py
Graph2.py
OUTLINE
TIMELINE
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER
Ln 110, Col 7 Spaces: 4 UTF-8 CRLF Python 3.10.7 64-bit Go Live Prettier
Type here to search
10:04 am
06/11/2022

```
# MUHAMMAD HARRIS - BCS203193
# Graph Algorithm Assignment 1
# GRAPH 1

# libraries
import os
import networkx
import matplotlib.pyplot as plot

# functions
def printMenu():...
def printNodeList(graph):...
def printEdgeList(graph):...
def printCountConnectedComponents(graph):...
def printConnectedComponents(graph):...
```

```
|------------------------------ MENU ------------------------------|
1. Display Nodes List of the Graph
2. Display Edge List of the Graph
3. Count Connected Components of the Graph
4. Print Connected Components of the Graph
5. Display Incidence Matrix of a Graph
6. Display the Nodes degrees
7. Count Number of Edges
8. Visualize the graphs
9. Check if:
    a. Euler circuit exist or not and Graph is Eulerian
    b. Euler path exists or not
    c. Hamilton Path
    d. Perform Depth First and Breadth First Traversal on graph

Option: 
```

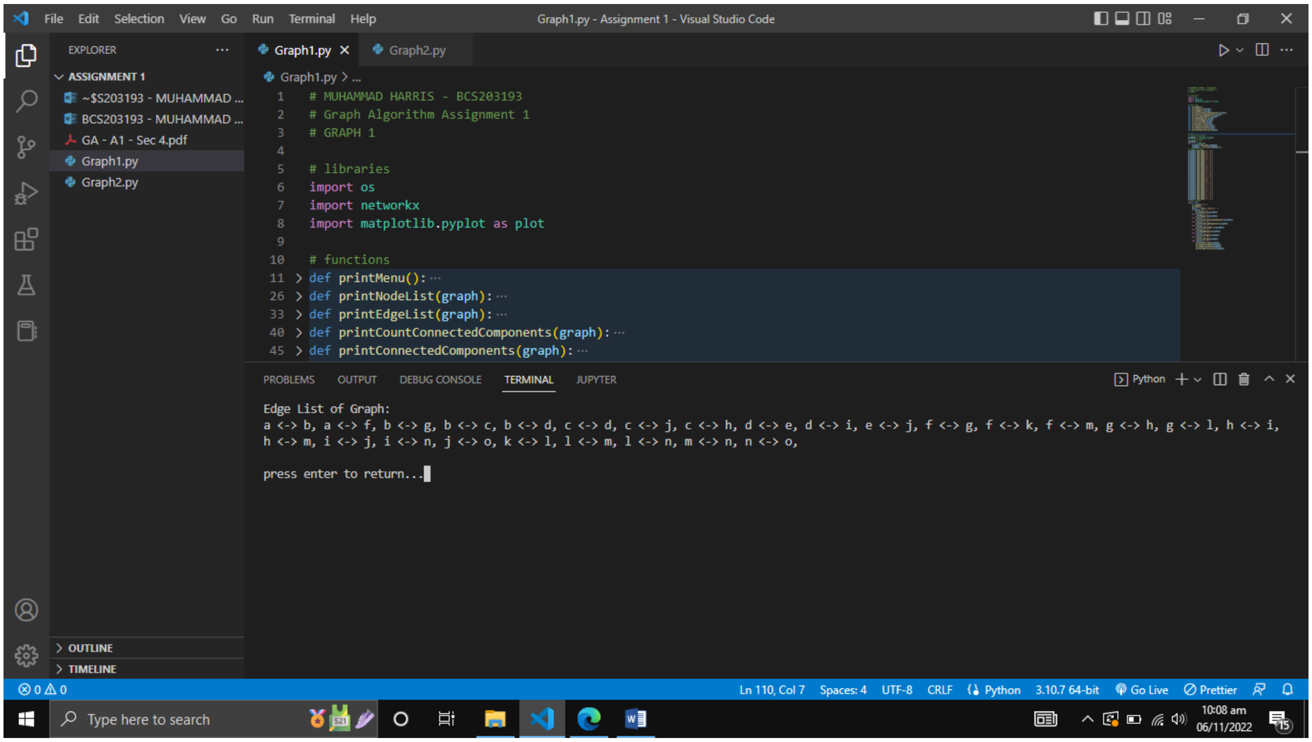
Output 1: main menu


Graph1.py - Assignment 1 - Visual Studio Code
EXPLORER
ASSIGNMENT 1
~$S203193 - MUHAMMAD ...
BCS203193 - MUHAMMAD ...
GA - A1 - Sec 4.pdf
Graph1.py
Graph2.py
OUTLINE
TIMELINE
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER
Ln 110, Col 7 Spaces: 4 UTF-8 CRLF Python 3.10.7 64-bit Go Live Prettier
Type here to search
10:06 am
06/11/2022

```
# MUHAMMAD HARRIS - BCS203193
# Graph Algorithm Assignment 1
# GRAPH 1

# libraries
import os
import networkx
import matplotlib.pyplot as plot

# functions
def printMenu():...
def printNodeList(graph):...
def printEdgeList(graph):...
def printCountConnectedComponents(graph):...
def printConnectedComponents(graph):...
```

```
Node List of Graph:
a, b, c, d, e, f, g, h, i, j, k, l, m, n, o,

press enter to return...
```

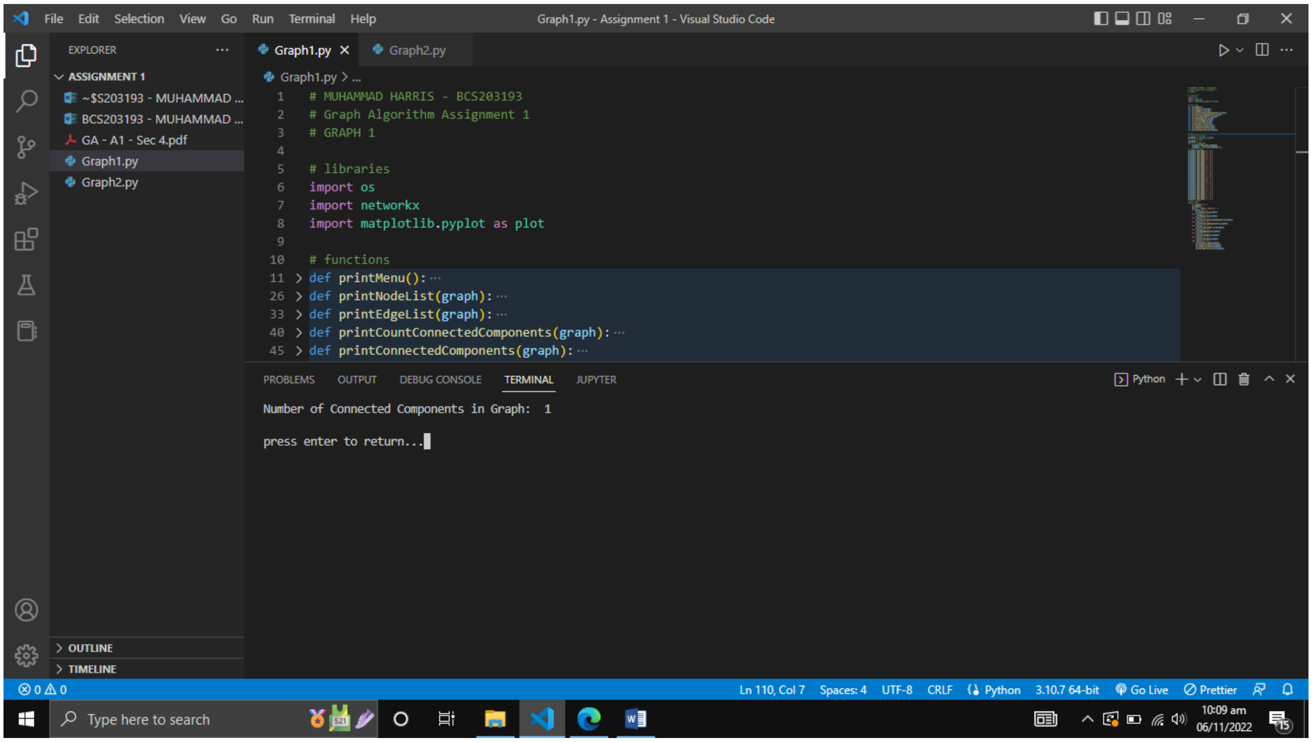
Output 2: node list of graph


Graph1.py - Assignment 1 - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
ASSIGNMENT 1
~$S203193 - MUHAMMAD ...
BCS203193 - MUHAMMAD ...
GA - A1 - Sec 4.pdf
Graph1.py
Graph2.py
OUTLINE
TIMELINE
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER
Python
Ln 110, Col 7 Spaces: 4 UTF-8 CRLF Python 3.10.7 64-bit Go Live Prettier
Type here to search
10:08 am
06/11/2022

```
# MUHAMMAD HARRIS - BCS203193
# Graph Algorithm Assignment 1
# GRAPH 1

# libraries
import os
import networkx
import matplotlib.pyplot as plot

# functions
def printMenu():...
def printNodeList(graph):...
def printEdgeList(graph):...
def printCountConnectedComponents(graph):...
def printConnectedComponents(graph):...
```

```
Edge List of Graph:
a <-> b, a <-> f, b <-> g, b <-> c, b <-> d, c <-> d, c <-> j, c <-> h, d <-> e, d <-> i, e <-> j, f <-> g, f <-> k, f <-> m, g <-> h, g <-> l, h <-> i,
h <-> m, i <-> j, i <-> n, j <-> o, k <-> l, l <-> m, l <-> n, m <-> n, n <-> o,

press enter to return...
```

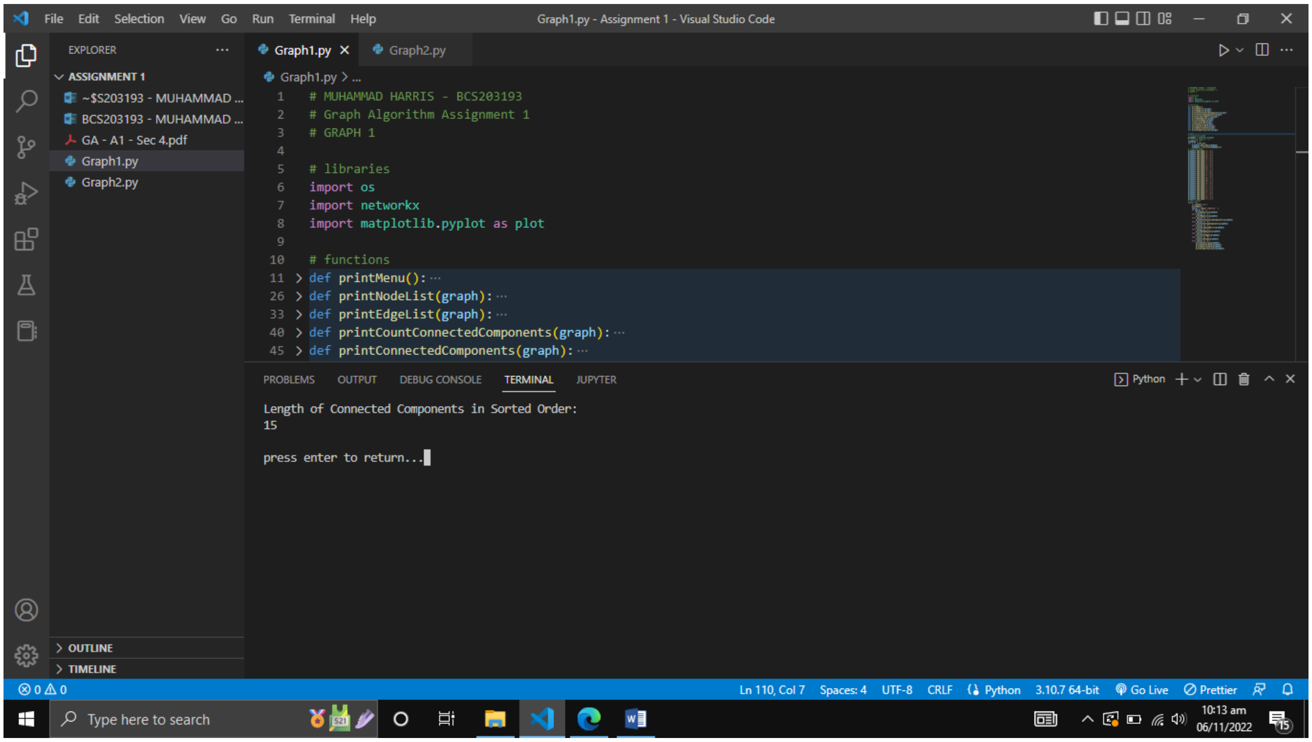
Output 3: edge list of graph


Graph1.py - Assignment 1 - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
ASSIGNMENT 1
~$S203193 - MUHAMMAD ...
BCS203193 - MUHAMMAD ...
GA - A1 - Sec 4.pdf
Graph1.py
Graph2.py
OUTLINE
TIMELINE
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER
Python
Ln 110, Col 7 Spaces: 4 UTF-8 CRLF Python 3.10.7 64-bit Go Live Prettier
Type here to search
10:09 am
06/11/2022

```
# MUHAMMAD HARRIS - BCS203193
# Graph Algorithm Assignment 1
# GRAPH 1

# libraries
import os
import networkx
import matplotlib.pyplot as plot

# functions
def printMenu():...
def printNodeList(graph):...
def printEdgeList(graph):...
def printCountConnectedComponents(graph):...
def printConnectedComponents(graph):...
```

```
Number of Connected Components in Graph:  1

press enter to return...
```

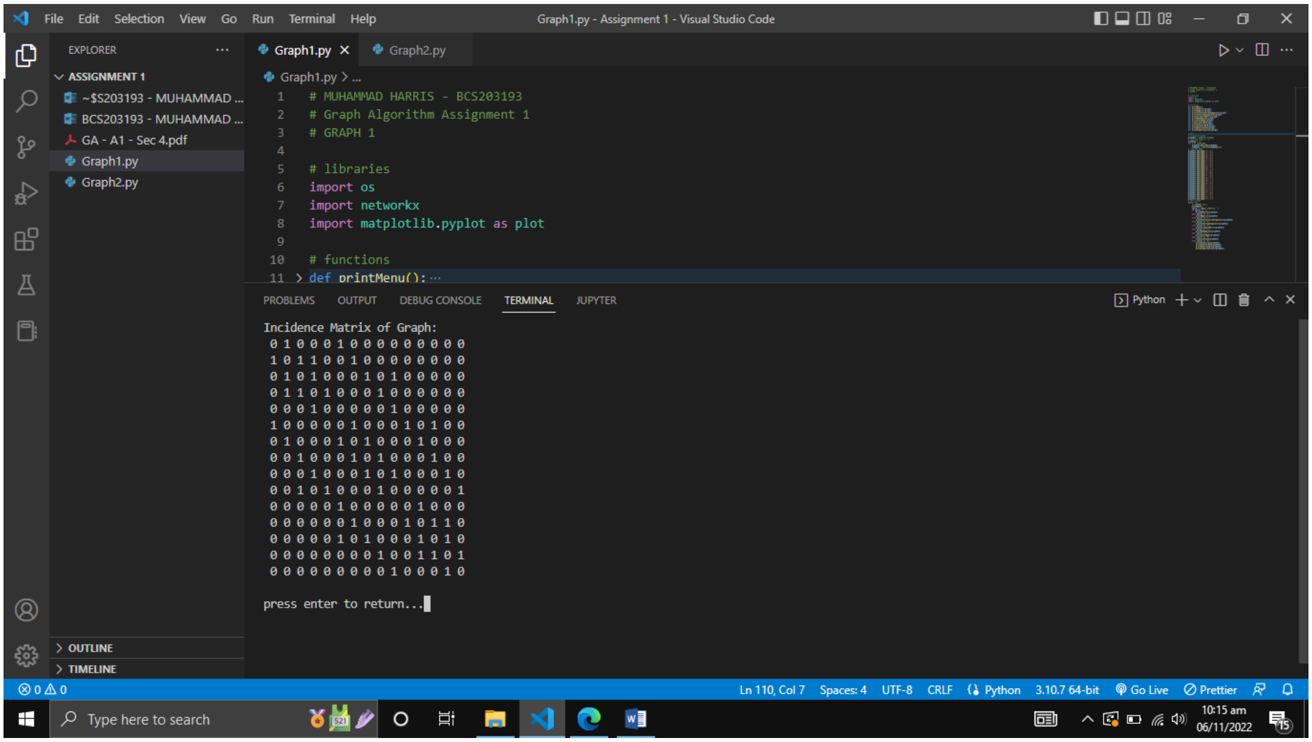
Output 4: number of connected components in graph



```
# MUHAMMAD HARRIS - BCS203193
# Graph Algorithm Assignment 1
# GRAPH 1

# libraries
import os
import networkx
import matplotlib.pyplot as plot

# functions
def printMenu(): ···
def printNodeList(graph): ···
def printEdgeList(graph): ···
def printCountConnectedComponents(graph): ···
def printConnectedComponents(graph): ···
```

```
Length of Connected Components in Sorted Order:
15

press enter to return...
```

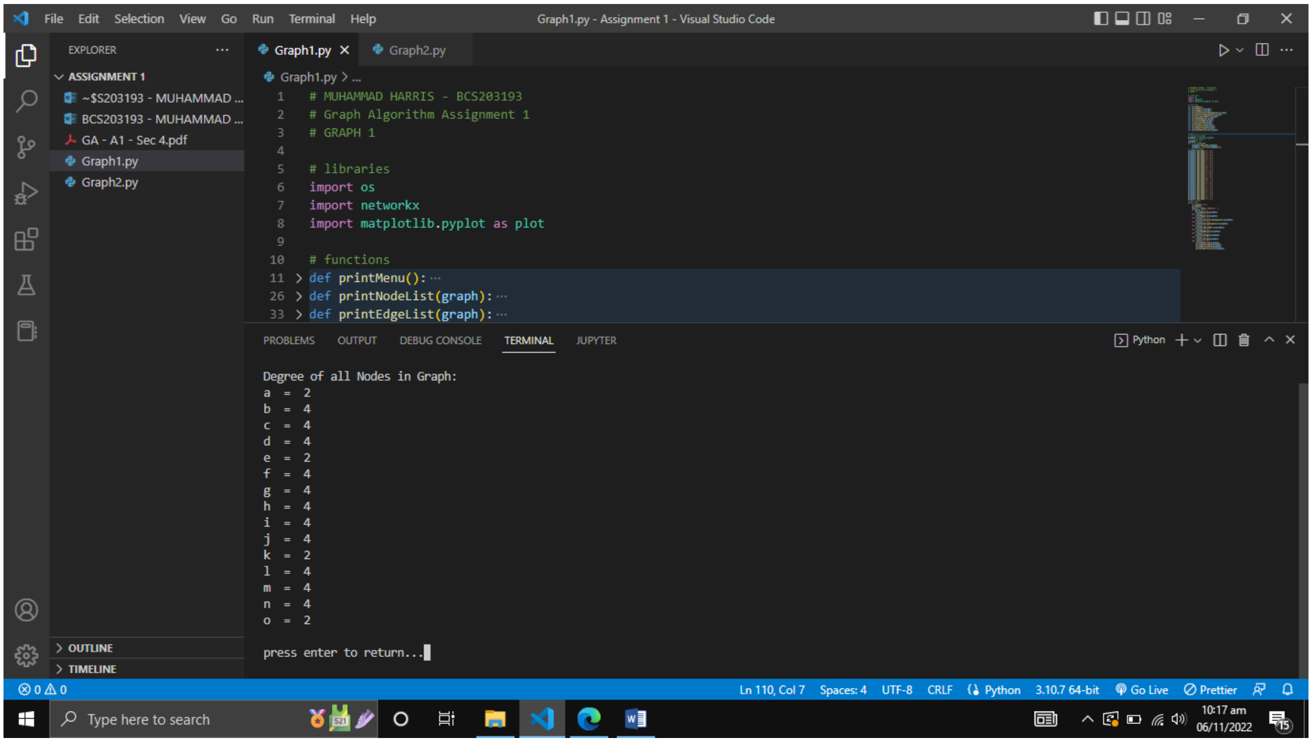
Output 5: connected components of the graph



```
# MUHAMMAD HARRIS - BCS203193
# Graph Algorithm Assignment 1
# GRAPH 1

# libraries
import os
import networkx
import matplotlib.pyplot as plot

# functions
def printMenu(): ···
```

```
Incidence Matrix of Graph:
 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0
 1 0 1 1 0 0 1 0 0 0 0 0 0 0 0
 0 1 0 1 0 0 0 1 0 1 0 0 0 0 0
 0 1 1 0 1 0 0 0 1 0 0 0 0 0 0
 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0
 1 0 0 0 0 0 1 0 0 0 1 0 1 0 0
 0 1 0 0 0 1 0 1 0 0 0 1 0 0 0
 0 0 1 0 0 0 1 0 1 0 0 0 1 0 0
 0 0 0 1 0 0 0 1 0 1 0 0 0 1 0
 0 0 1 0 1 0 0 0 1 0 0 0 0 0 1
 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0
 0 0 0 0 0 0 1 0 0 0 1 0 1 1 0
 0 0 0 0 0 1 0 1 0 0 0 1 0 1 0
 0 0 0 0 0 0 0 0 1 0 0 1 1 0 1
 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0

press enter to return...
```

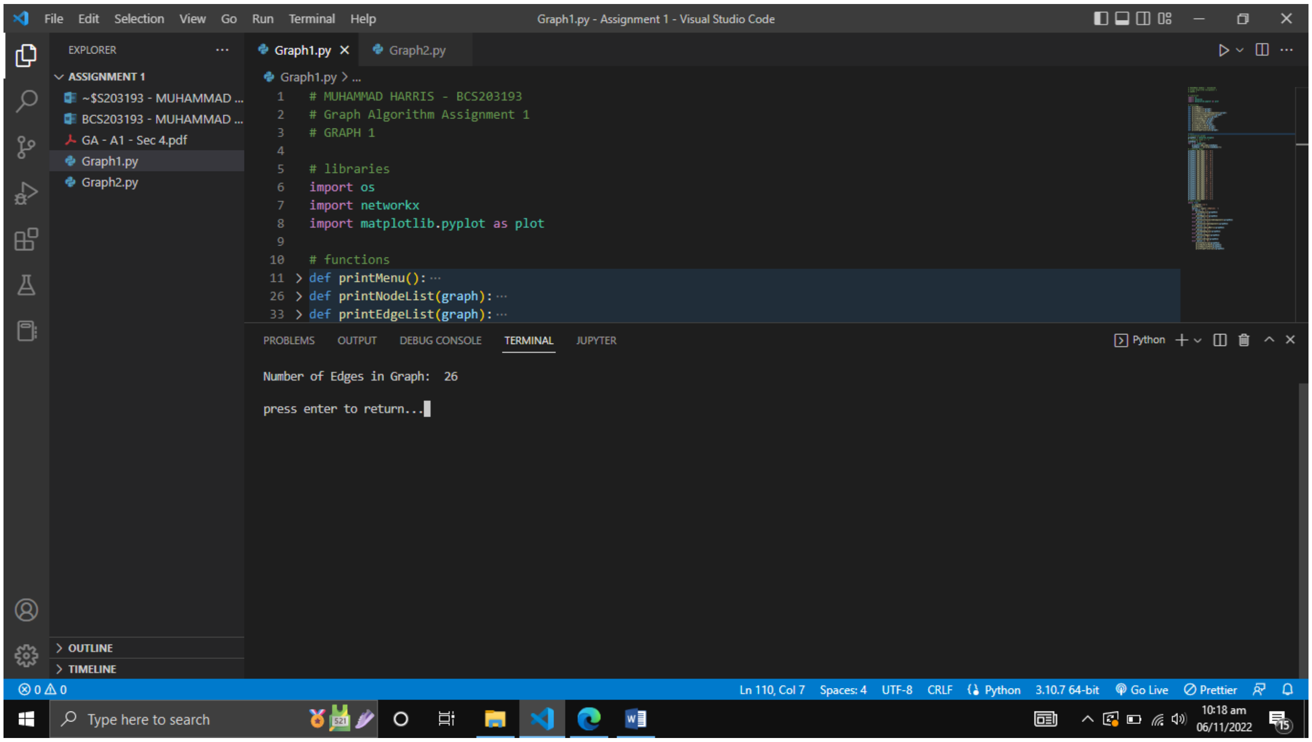
Output 6: incidence matrix of graph


File Edit Selection View Go Run Terminal Help
Graph1.py - Assignment 1 - Visual Studio Code
EXPLORER
ASSIGNMENT 1
~$S203193 - MUHAMMAD ...
BCS203193 - MUHAMMAD ...
GA - A1 - Sec 4.pdf
Graph1.py
Graph2.py
OUTLINE
TIMELINE
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER
Python
Ln 110, Col 7 Spaces: 4 UTF-8 CRLF Python 3.10.7 64-bit Go Live Prettier
Type here to search
10:17 am
06/11/2022

```python
# MUHAMMAD HARRIS - BCS203193
# Graph Algorithm Assignment 1
# GRAPH 1

# libraries
import os
import networkx
import matplotlib.pyplot as plot

# functions
def printMenu(): ···
def printNodeList(graph): ···
def printEdgeList(graph): ···
```

```
Degree of all Nodes in Graph:
a  =  2
b  =  4
c  =  4
d  =  4
e  =  2
f  =  4
g  =  4
h  =  4
i  =  4
j  =  4
k  =  2
l  =  4
m  =  4
n  =  4
o  =  2

press enter to return...
```

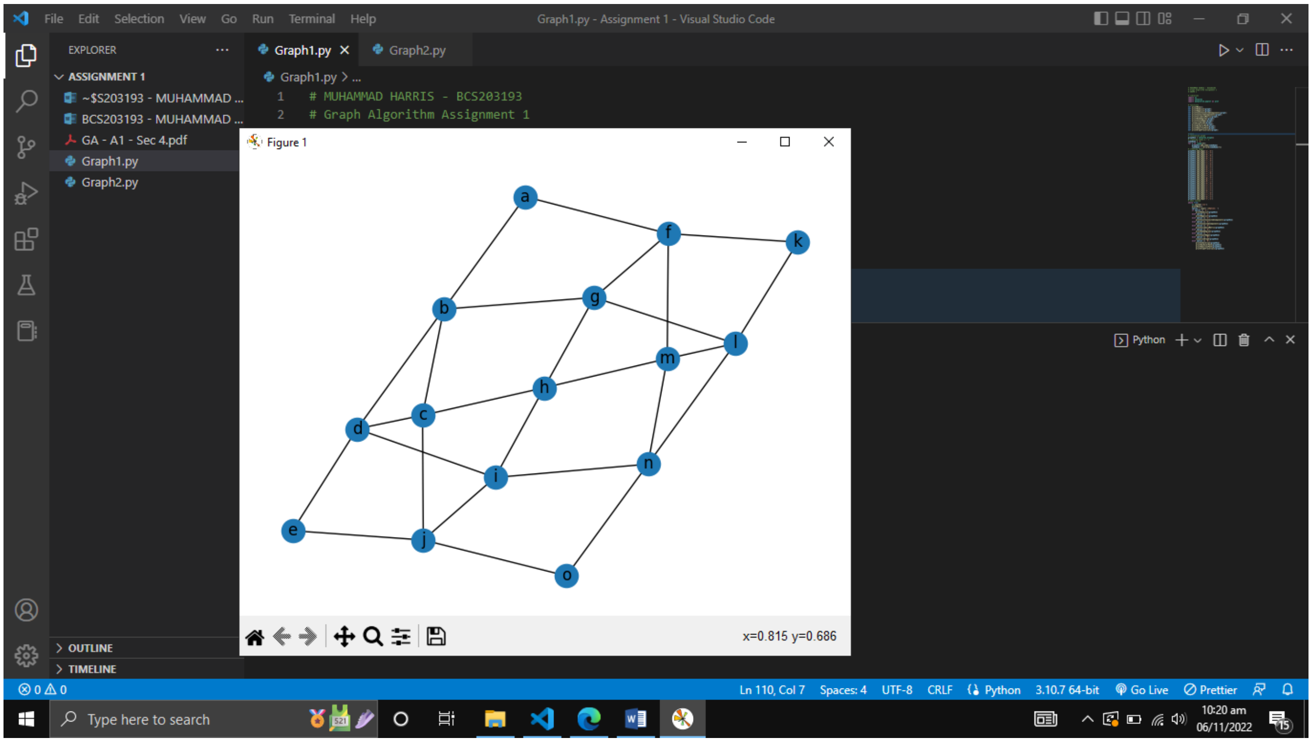
Output 7: node degrees


File Edit Selection View Go Run Terminal Help
Graph1.py - Assignment 1 - Visual Studio Code
EXPLORER
ASSIGNMENT 1
~$S203193 - MUHAMMAD ...
BCS203193 - MUHAMMAD ...
GA - A1 - Sec 4.pdf
Graph1.py
Graph2.py
OUTLINE
TIMELINE
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER
Python
Ln 110, Col 7 Spaces: 4 UTF-8 CRLF Python 3.10.7 64-bit Go Live Prettier
Type here to search
10:18 am
06/11/2022

```python
# MUHAMMAD HARRIS - BCS203193
# Graph Algorithm Assignment 1
# GRAPH 1

# libraries
import os
import networkx
import matplotlib.pyplot as plot

# functions
def printMenu(): ···
def printNodeList(graph): ···
def printEdgeList(graph): ···
```

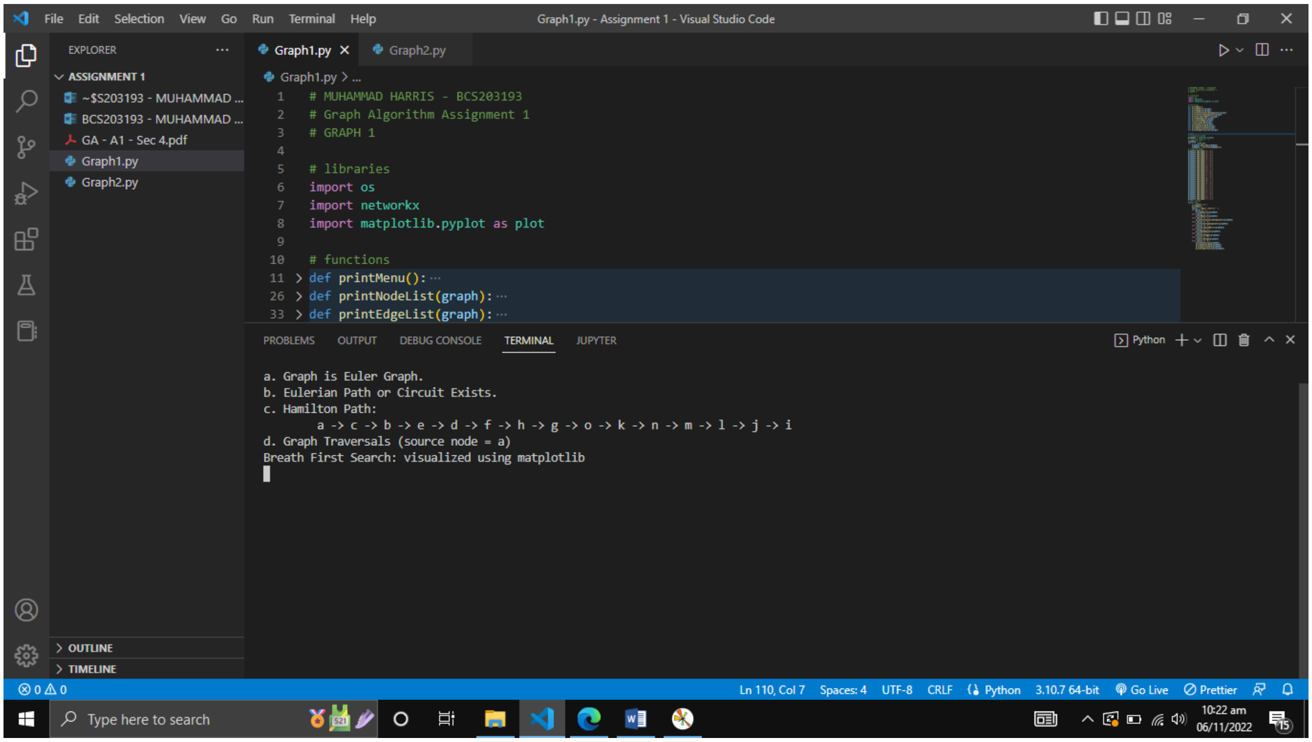
```
Number of Edges in Graph:  26

press enter to return...
```

Output 8: number of edges in graph



```
# MUHAMMAD HARRIS - BCS203193
# Graph Algorithm Assignment 1
```

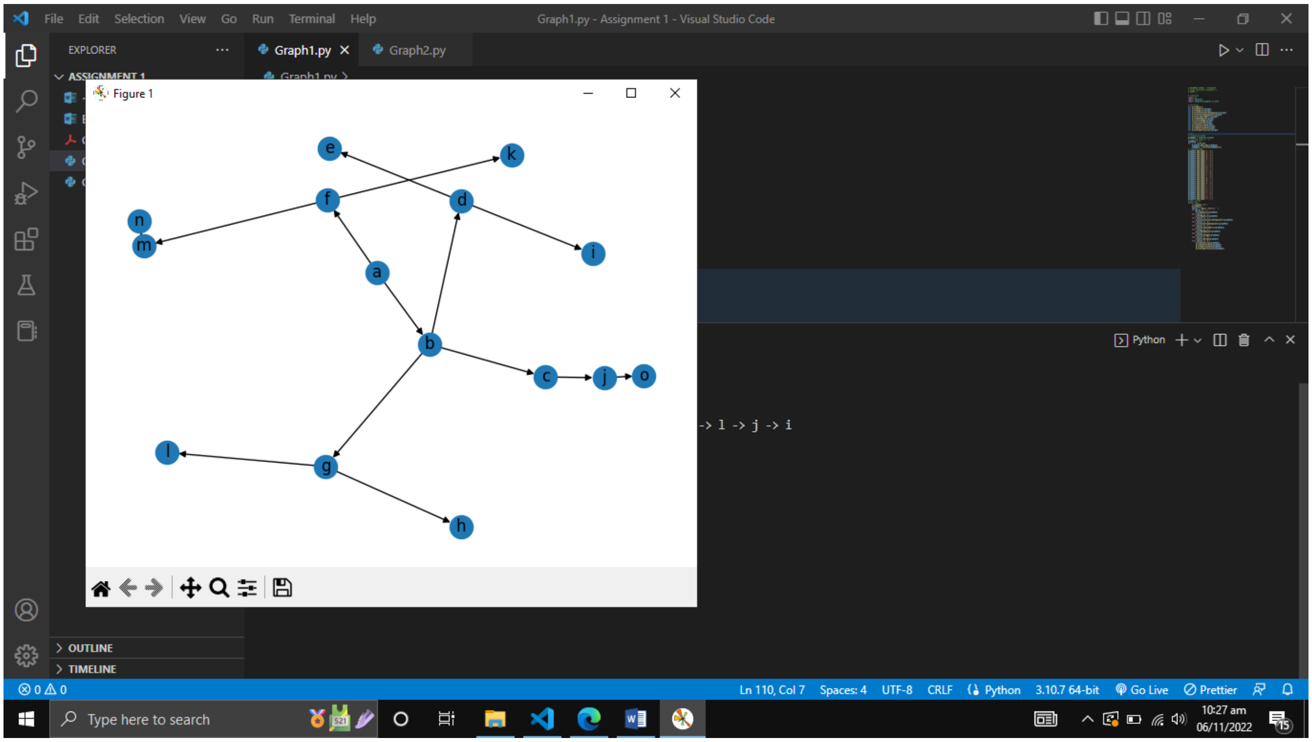
Output 9: visualization of graph


File Edit Selection View Go Run Terminal Help
Graph1.py - Assignment 1 - Visual Studio Code
EXPLORER
ASSIGNMENT 1
~$S203193 - MUHAMMAD ...
BCS203193 - MUHAMMAD ...
GA - A1 - Sec 4.pdf
Graph1.py
Graph2.py
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER
OUTLINE
TIMELINE
Ln 110, Col 7 Spaces: 4 UTF-8 CRLF Python 3.10.7 64-bit Go Live Prettier
Type here to search
10:22 am
06/11/2022

```python
# MUHAMMAD HARRIS - BCS203193
# Graph Algorithm Assignment 1
# GRAPH 1

# libraries
import os
import networkx
import matplotlib.pyplot as plot

# functions
def printMenu():···
def printNodeList(graph):···
def printEdgeList(graph):···
```

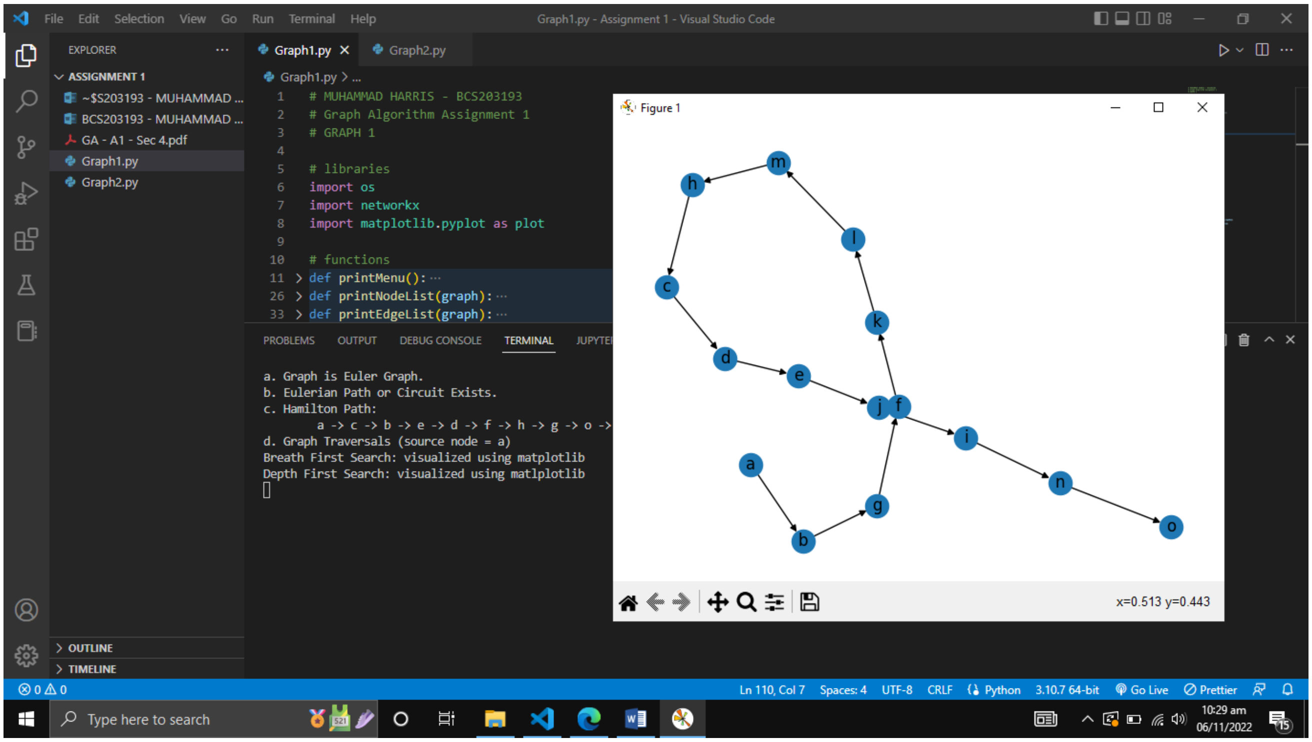
```
a. Graph is Euler Graph.
b. Eulerian Path or Circuit Exists.
c. Hamilton Path:
        a -> c -> b -> e -> d -> f -> h -> g -> o -> k -> n -> m -> l -> j -> i
d. Graph Traversals (source node = a)
Breath First Search: visualized using matplotlib
```

Output 10: eulerian & hamilton path


File Edit Selection View Go Run Terminal Help
Graph1.py - Assignment 1 - Visual Studio Code
EXPLORER
ASSIGNMENT 1
Graph1.py
Graph2.py
Figure 1
e
k
f
d
n
m
i
a
b
c
j
o
l
g
h
OUTLINE
TIMELINE
Python
Ln 110, Col 7 Spaces: 4 UTF-8 CRLF Python 3.10.7 64-bit Go Live Prettier
Type here to search
10:27 am
06/11/2022

```
-> l -> j -> i
```

Output 11: BFS traversal



```python
# MUHAMMAD HARRIS - BCS203193
# Graph Algorithm Assignment 1
# GRAPH 1

# libraries
import os
import networkx
import matplotlib.pyplot as plot

# functions
def printMenu():···
def printNodeList(graph):···
def printEdgeList(graph):···
```

```
a. Graph is Euler Graph.
b. Eulerian Path or Circuit Exists.
c. Hamilton Path:
        a -> c -> b -> e -> d -> f -> h -> g -> o ->
d. Graph Traversals (source node = a)
Breath First Search: visualized using matplotlib
Depth First Search: visualized using matlplotlib
```

Output 12: DFS traversal

Graph 2:

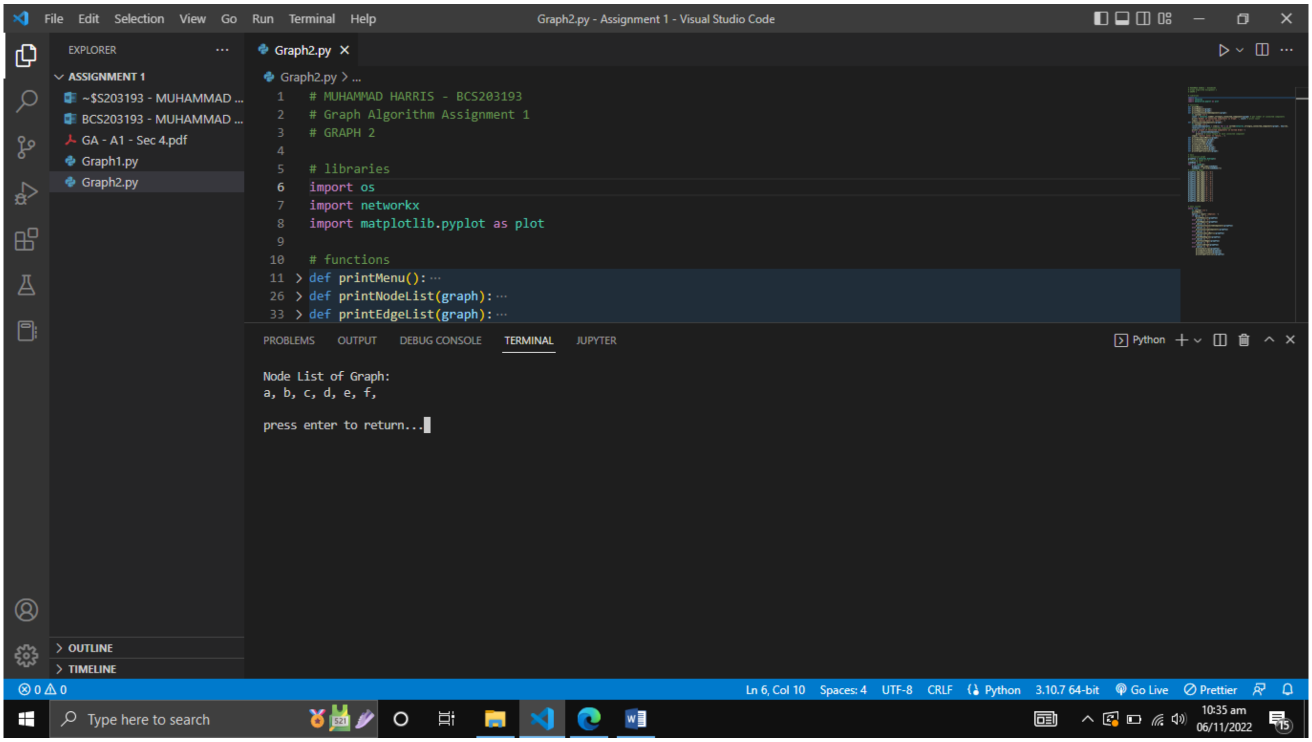
Code of Graph2.py

```
# MUHAMMAD HARRIS - BCS203193
# Graph Algorithm Assignment 1
# GRAPH 2

# libraries
import os
import networkx
import matplotlib.pyplot as plot

# functions
def printMenu():
    print("""|-------------------------------- MENU --------------------------------|
1. Display Nodes List of the Graph
2. Display Edge List of the Graph
3. Count Connected Components of the Graph
4. Print Connected Components of the Graph
5. Display Incidence Matrix of a Graph
6. Display the Nodes degrees
7. Count Number of Edges
8. Visualize the graphs
9. Check if:
    a. Euler circuit exist or not and Graph is Eulerian
    b. Euler path exists or not
    c. Hamilton Path
    d. Perform Depth First and Breadth First Traversal on graph""")
def printNodeList(graph):
    os.system('cls')
    nodeList = graph.nodes() # get node list of graph
    print('Node List of Graph:')
    for x in nodeList:
        print(x, end=', ') # print each node
    input('\n\npress enter to return...')
def printEdgeList(graph):
    os.system('cls')
    edgeList = graph.edges() # get edge list of graph
    print('Edge List of Graph:')
    for x in edgeList:
        print(x[0], '->', x[1], end=', ') # print each edge
    input('\n\npress enter to return...')
def printCountConnectedComponents(graph):
    os.system('cls')
    count = networkx.number_strongly_connected_components(graph) # get number of
connected components
```

```
    print('Number of Connected Components in Graph: ', count) # print count
    input('\npress enter to return...')
def printConnectedComponents(graph):
    os.system('cls')
    connectedComponents = [len(c) for c in 
sorted(networkx.strongly_connected_components(graph), key=len, reverse=True)] # get 
connected components
    print('Length of Connected Components in Sorted Order:')
    for x in connectedComponents:
        print(x) # print length of each connected component
    input('\npress enter to return...')
def printIncidenceMatrix(graph):
    os.system('cls')
    incidenceMatrix = networkx.to_numpy_matrix(graph) # get incidence matix
    incidenceMatrixString = 
str(incidenceMatrix).replace('[','').replace(']','').replace('.','')
    print('Incidence Matrix of Graph: ')
    print('',incidenceMatrixString) # print incidence matrix
    input('\npress enter to return...')
def printNodeDegrees(graph):
    os.system('cls')
    print('in-Degree of all Nodes in Graph:')
    nodeN = 'a'
    for x in range(6):
        print(nodeN, ' = ', graph.in_degree(nodeN)) # print degree of each node
        nodeN = chr(ord(nodeN)+1)
    print('out-Degree of all Nodes in Graph:')
    nodeN = 'a'
    for x in range(6):
        print(nodeN, ' = ', graph.out_degree(nodeN)) # print degree of each node
        nodeN = chr(ord(nodeN)+1)
    input('\npress enter to return...')
def printCountEdges(graph):
    os.system('cls')
    print('Number of Edges in Graph (parallel edges ignored): ', 
networkx.number_of_edges(graph)) # print number of edges in graph
    input('\npress enter to return...')
def visualizeGraph(graph):
    os.system('cls')
    networkx.draw(graph, with_labels = True) # draw graph with labels
    plot.show()
    input('press enter to return...')
def printIsEulerian(graph):
    os.system('cls')
    if networkx.is_eulerian(graph): # check if graph is euler or not
```

```
        print('a. Graph is Euler Graph.')
    else:
        print('a. Graph is not Euler.')
def printHasEulerPath(graph):
    if networkx.has_eulerian_path(graph): # check if graph has euler path or not
        print('b. Eulerian Path or Circuit Exists.')
    else:
        print('b. Eulerian Path or Circuit does not Exist.')
def printHamiltonPath(graph):
    try:
        hamiltonPath = networkx.algorithms.tournament.hamiltonian_path(graph) # get 
hamilton path
        print('c. Hamilton Path: ', )
        print('\t', end='')
        for x in hamiltonPath:
            if x != 'a':
                print(' ->', end=' ')
            print(x, end='') # print hamilton path if exists
    except:
        print('c. Hamilton Path does not exist.') # else print exception
def printGraphTraversals(graph):
    print('\nd. Graph Traversals (source node = a)')
    print('Breath First Search: visualized using matplotlib')
    BFStree = networkx.bfs_tree(graph, 'a') # create BFS tree
    networkx.draw(BFStree, with_labels=True)
    plot.show() # visualize BFS tree
    print('Depth First Search: visualized using matlplotlib')
    DFStree = networkx.dfs_tree(graph, 'a') # create DFS tree
    networkx.draw(DFStree, with_labels=True)
    plot.show() # visualize DFS tree
    input('\npress enter key to return...')

# main
# initialize graph
graphTwo = networkx.DiGraph()
# creating nodes a - f
nodeName = 'a'
for x in range(6):
    graphTwo.add_node(nodeName)
    nodeName = chr(ord(nodeName)+1)
# creating edges
graphTwo.add_edge('a','b')
graphTwo.add_edge('a','f')
graphTwo.add_edge('b','c')
graphTwo.add_edge('b','d')
graphTwo.add_edge('b','f')
graphTwo.add_edge('c','b')
graphTwo.add_edge('c','b')
graphTwo.add_edge('c','e')
graphTwo.add_edge('d','c')
graphTwo.add_edge('d','e')
graphTwo.add_edge('e','a')
graphTwo.add_edge('e','b')
graphTwo.add_edge('e','f')
graphTwo.add_edge('f','a')
graphTwo.add_edge('f','d')
graphTwo.add_edge('f','e')


# menu system
while True:
    os.system('cls')
    printMenu()
    option = input('\nOption: ')
    if option == '1':
        printNodeList(graphTwo)
    elif option == '2':
        printEdgeList(graphTwo)
    elif option == '3':
        printCountConnectedComponents(graphTwo)
    elif option == '4':
        printConnectedComponents(graphTwo)
    elif option == '5':
        printIncidenceMatrix(graphTwo)
    elif option == '6':
        printNodeDegrees(graphTwo)
    elif option == '7':
        printCountEdges(graphTwo)
    elif option == '8':
        visualizeGraph(graphTwo)
    elif option == '9':
        printIsEulerian(graphTwo)
        printHasEulerPath(graphTwo)
        printHamiltonPath(graphTwo)
        printGraphTraversals(graphTwo)
```

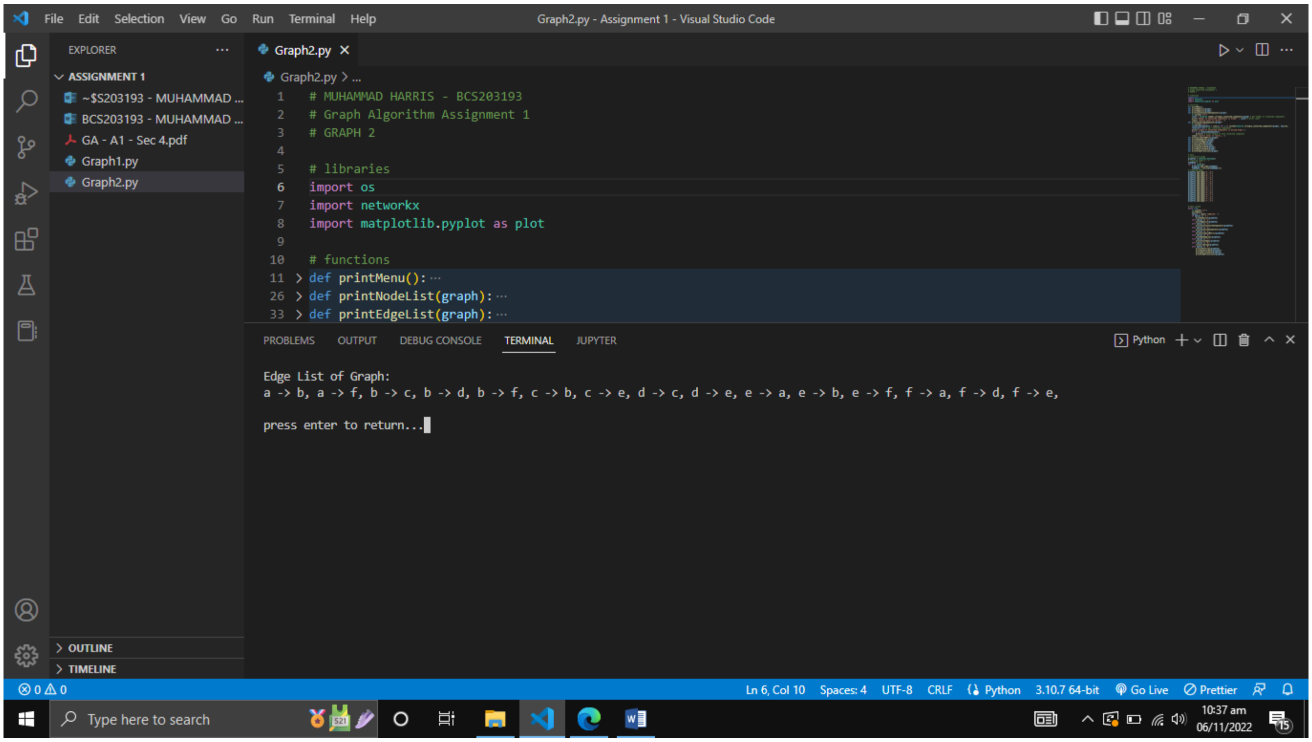
Outputs of Graph2.py



```
# MUHAMMAD HARRIS - BCS203193
# Graph Algorithm Assignment 1
# GRAPH 2

# libraries
import os
import networkx
import matplotlib.pyplot as plot

# functions
def printMenu(): ···
def printNodeList(graph): ···
def printEdgeList(graph): ···
```

```
Node List of Graph:
a, b, c, d, e, f,

press enter to return...
```

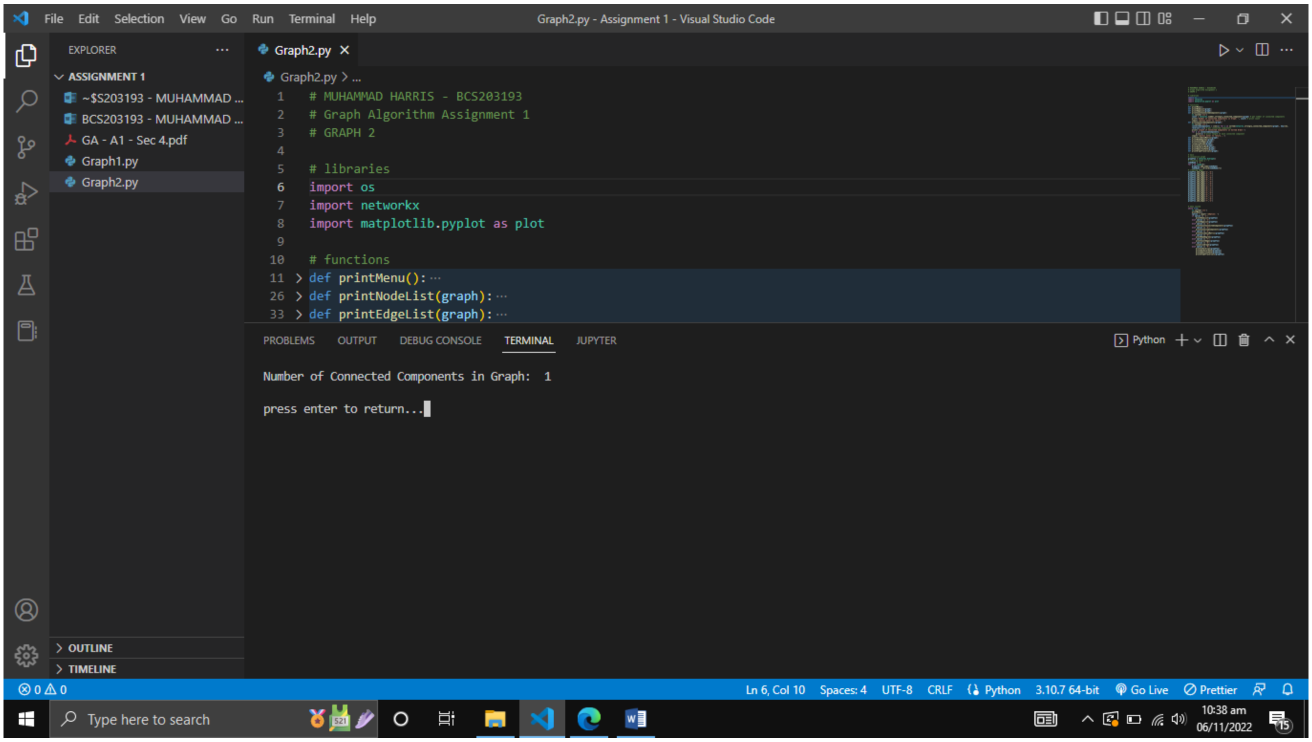
Output 1: node list of graph



```
# MUHAMMAD HARRIS - BCS203193
# Graph Algorithm Assignment 1
# GRAPH 2

# libraries
import os
import networkx
import matplotlib.pyplot as plot

# functions
def printMenu(): ···
def printNodeList(graph): ···
def printEdgeList(graph): ···
```

```
Edge List of Graph:
a -> b, a -> f, b -> c, b -> d, b -> f, c -> b, c -> e, d -> c, d -> e, e -> a, e -> b, e -> f, f -> a, f -> d, f -> e,

press enter to return...
```

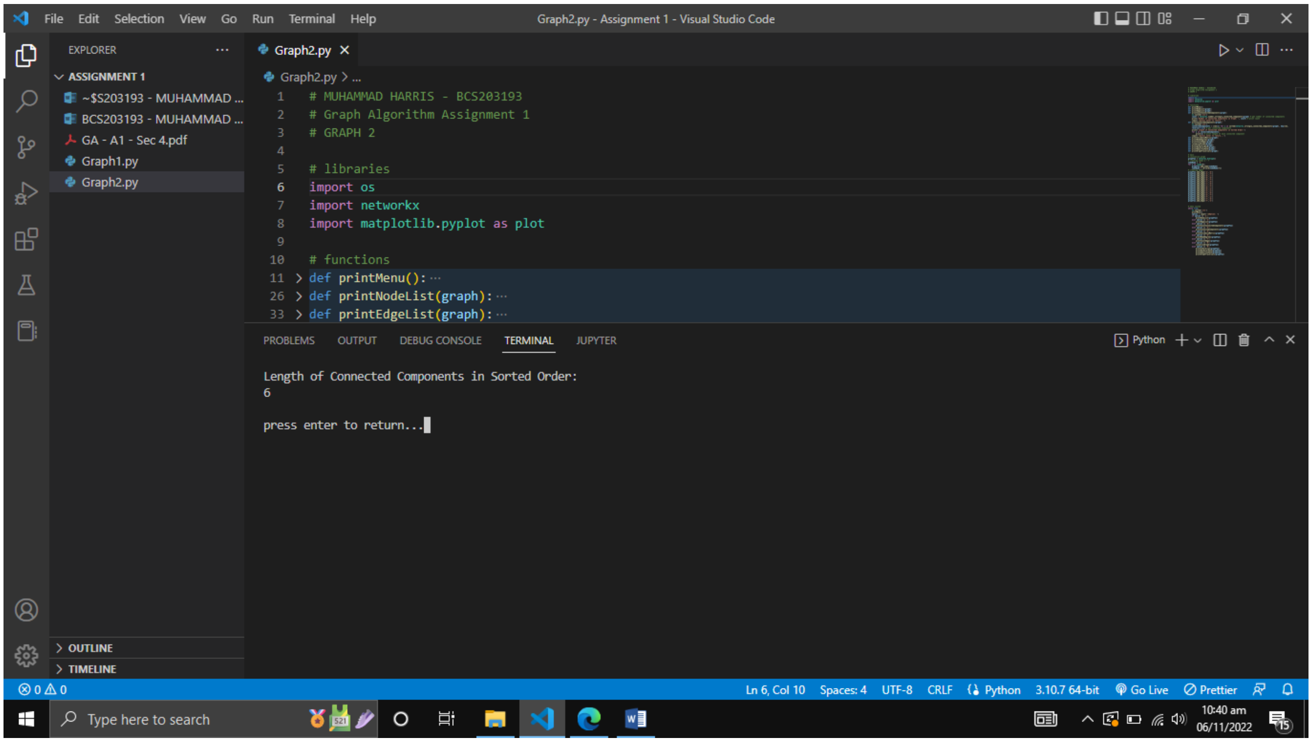
Output 2: node list of graph


File Edit Selection View Go Run Terminal Help
Graph2.py - Assignment 1 - Visual Studio Code
EXPLORER
ASSIGNMENT 1
~$S203193 - MUHAMMAD ...
BCS203193 - MUHAMMAD ...
GA - A1 - Sec 4.pdf
Graph1.py
Graph2.py
OUTLINE
TIMELINE
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER
Python
Ln 6, Col 10 Spaces: 4 UTF-8 CRLF Python 3.10.7 64-bit Go Live Prettier
Type here to search
10:38 am
06/11/2022

```python
# MUHAMMAD HARRIS - BCS203193
# Graph Algorithm Assignment 1
# GRAPH 2

# libraries
import os
import networkx
import matplotlib.pyplot as plot

# functions
def printMenu(): ···
def printNodeList(graph): ···
def printEdgeList(graph): ···
```

```
Number of Connected Components in Graph:  1

press enter to return...
```

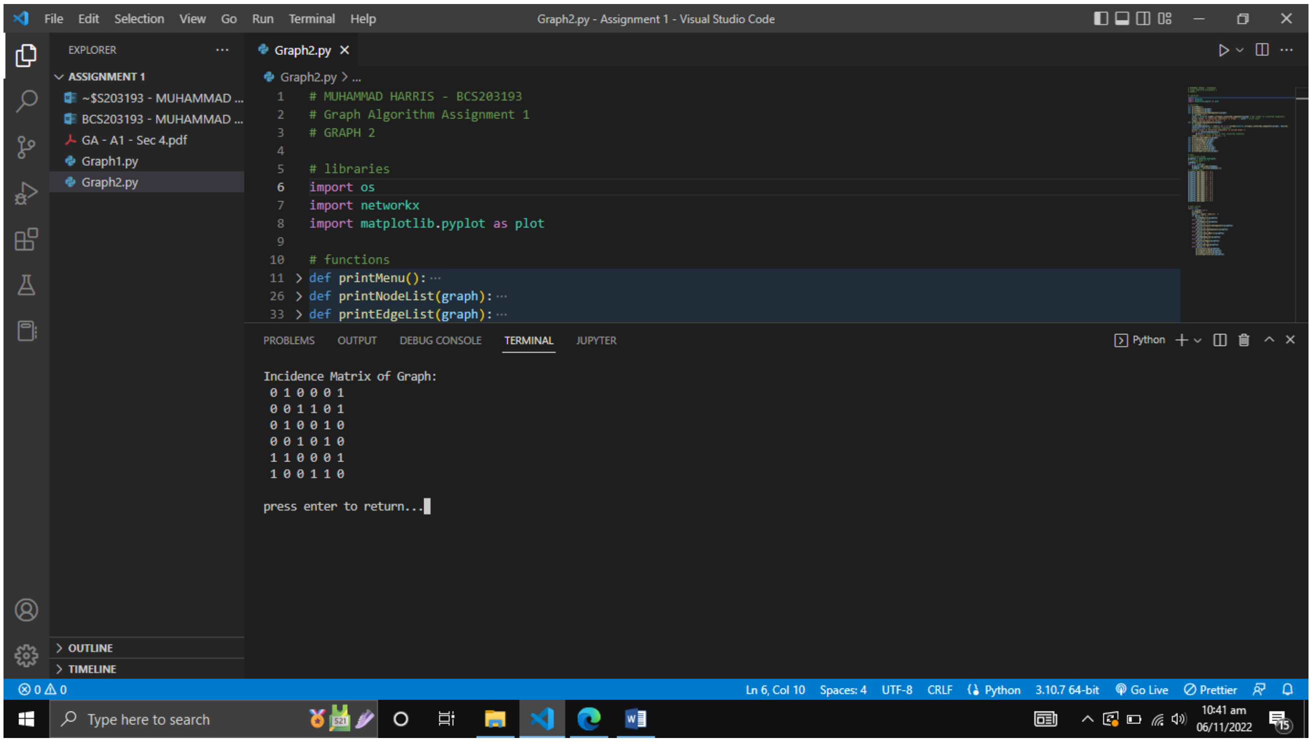
Output 3: number of connected components


File Edit Selection View Go Run Terminal Help
Graph2.py - Assignment 1 - Visual Studio Code
EXPLORER
ASSIGNMENT 1
~$S203193 - MUHAMMAD ...
BCS203193 - MUHAMMAD ...
GA - A1 - Sec 4.pdf
Graph1.py
Graph2.py
OUTLINE
TIMELINE
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER
Python
Ln 6, Col 10 Spaces: 4 UTF-8 CRLF Python 3.10.7 64-bit Go Live Prettier
Type here to search
10:40 am
06/11/2022

```python
# MUHAMMAD HARRIS - BCS203193
# Graph Algorithm Assignment 1
# GRAPH 2

# libraries
import os
import networkx
import matplotlib.pyplot as plot

# functions
def printMenu(): ···
def printNodeList(graph): ···
def printEdgeList(graph): ···
```

```
Length of Connected Components in Sorted Order:
6

press enter to return...
```

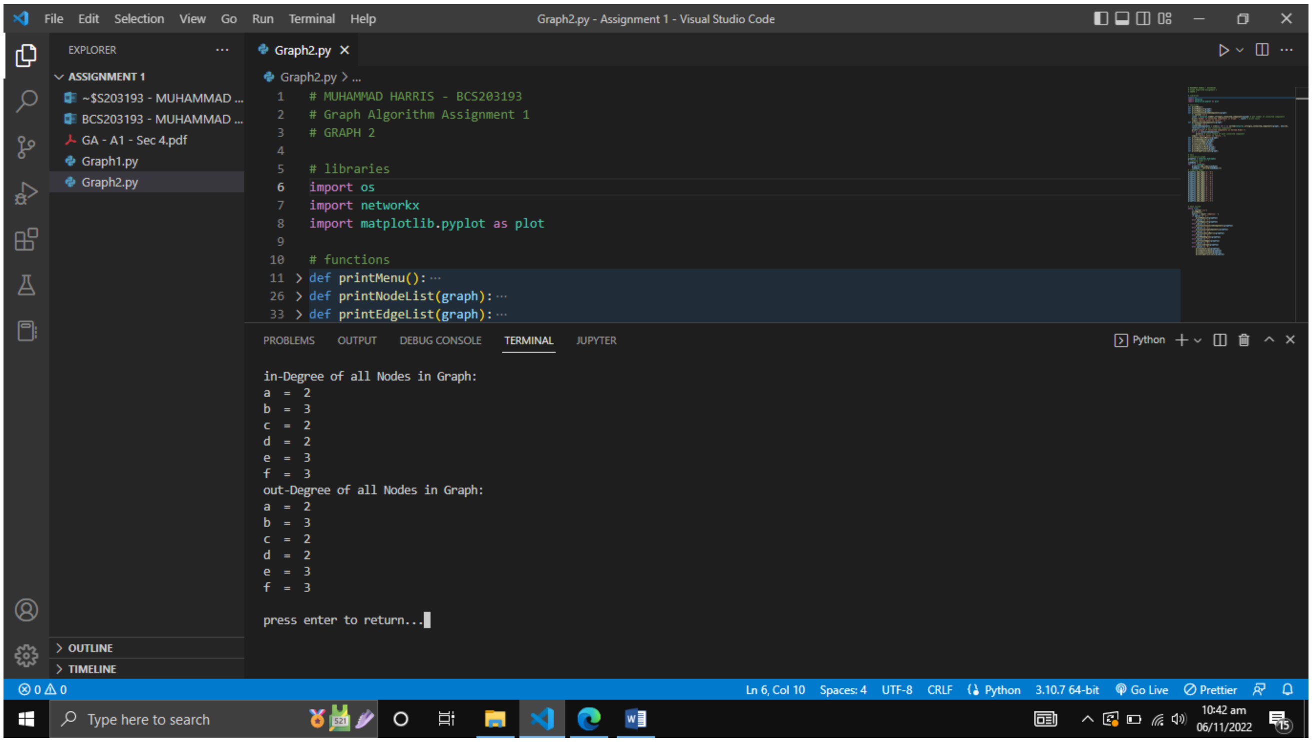
Output 4: connected components of graph


Graph2.py - Assignment 1 - Visual Studio Code
EXPLORER
ASSIGNMENT 1
~$S203193 - MUHAMMAD ...
BCS203193 - MUHAMMAD ...
GA - A1 - Sec 4.pdf
Graph1.py
Graph2.py
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER
Ln 6, Col 10 Spaces: 4 UTF-8 CRLF Python 3.10.7 64-bit Go Live Prettier
10:41 am
06/11/2022

```python
# MUHAMMAD HARRIS - BCS203193
# Graph Algorithm Assignment 1
# GRAPH 2

# libraries
import os
import networkx
import matplotlib.pyplot as plot

# functions
def printMenu(): ···
def printNodeList(graph): ···
def printEdgeList(graph): ···
```

```
Incidence Matrix of Graph:
 0 1 0 0 0 1
 0 0 1 1 0 1
 0 1 0 0 1 0
 0 0 1 0 1 0
 1 1 0 0 0 1
 1 0 0 1 1 0

press enter to return...
```

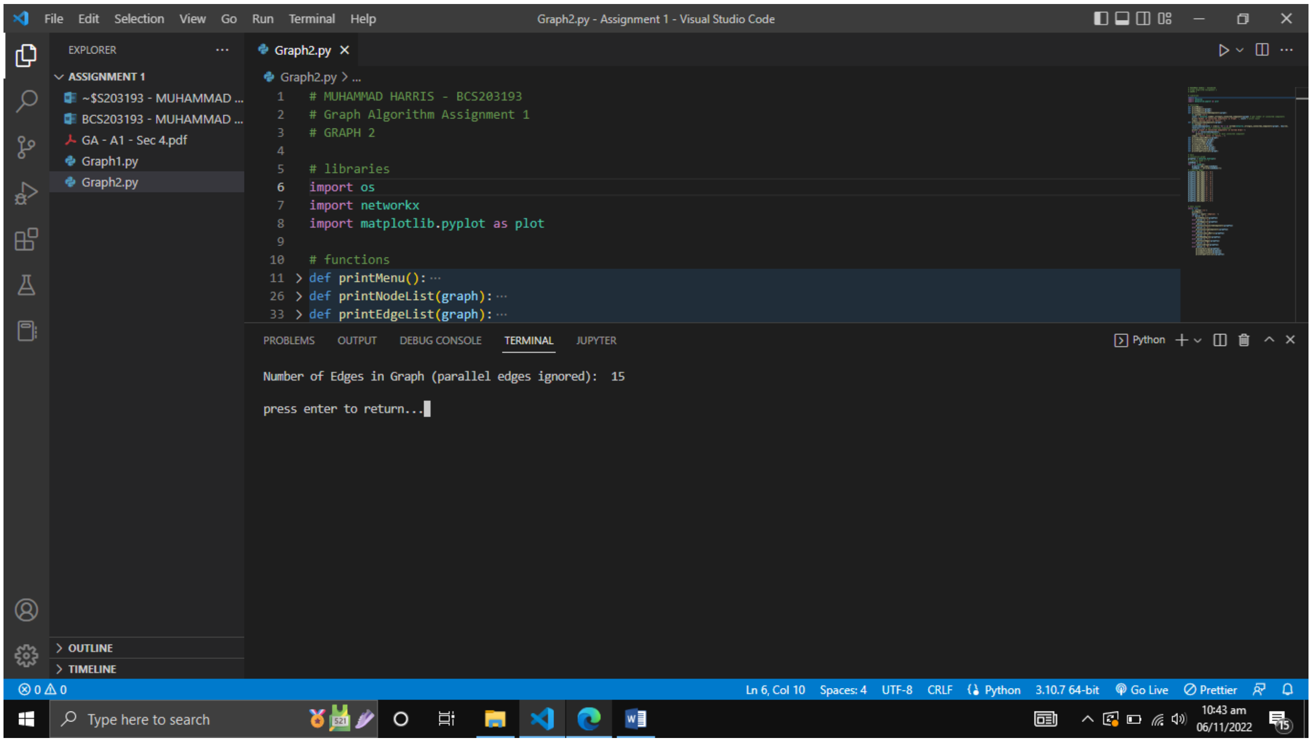
Output 5: incidence matrix of graph


Graph2.py - Assignment 1 - Visual Studio Code
EXPLORER
ASSIGNMENT 1
~$S203193 - MUHAMMAD ...
BCS203193 - MUHAMMAD ...
GA - A1 - Sec 4.pdf
Graph1.py
Graph2.py
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER
Ln 6, Col 10 Spaces: 4 UTF-8 CRLF Python 3.10.7 64-bit Go Live Prettier
10:42 am
06/11/2022

```python
# MUHAMMAD HARRIS - BCS203193
# Graph Algorithm Assignment 1
# GRAPH 2

# libraries
import os
import networkx
import matplotlib.pyplot as plot

# functions
def printMenu(): ···
def printNodeList(graph): ···
def printEdgeList(graph): ···
```

```
in-Degree of all Nodes in Graph:
a  =  2
b  =  3
c  =  2
d  =  2
e  =  3
f  =  3
out-Degree of all Nodes in Graph:
a  =  2
b  =  3
c  =  2
d  =  2
e  =  3
f  =  3

press enter to return...
```

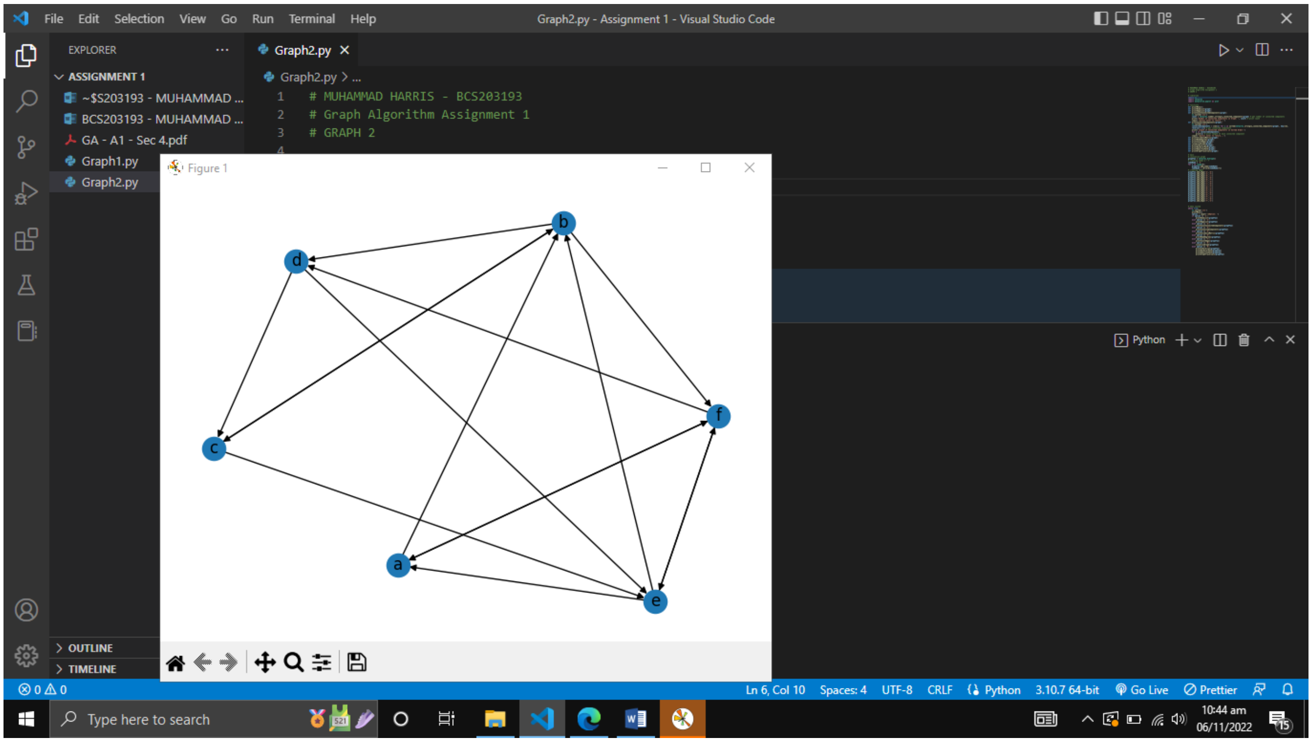
Output 6: in-degree & out-degree of nodes


File Edit Selection View Go Run Terminal Help
Graph2.py - Assignment 1 - Visual Studio Code
EXPLORER
ASSIGNMENT 1
~$S203193 - MUHAMMAD ...
BCS203193 - MUHAMMAD ...
GA - A1 - Sec 4.pdf
Graph1.py
Graph2.py
OUTLINE
TIMELINE
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER
Python
Ln 6, Col 10 Spaces: 4 UTF-8 CRLF Python 3.10.7 64-bit Go Live Prettier
Type here to search
10:43 am
06/11/2022

```python
# MUHAMMAD HARRIS - BCS203193
# Graph Algorithm Assignment 1
# GRAPH 2

# libraries
import os
import networkx
import matplotlib.pyplot as plot

# functions
def printMenu(): ···
def printNodeList(graph): ···
def printEdgeList(graph): ···
```

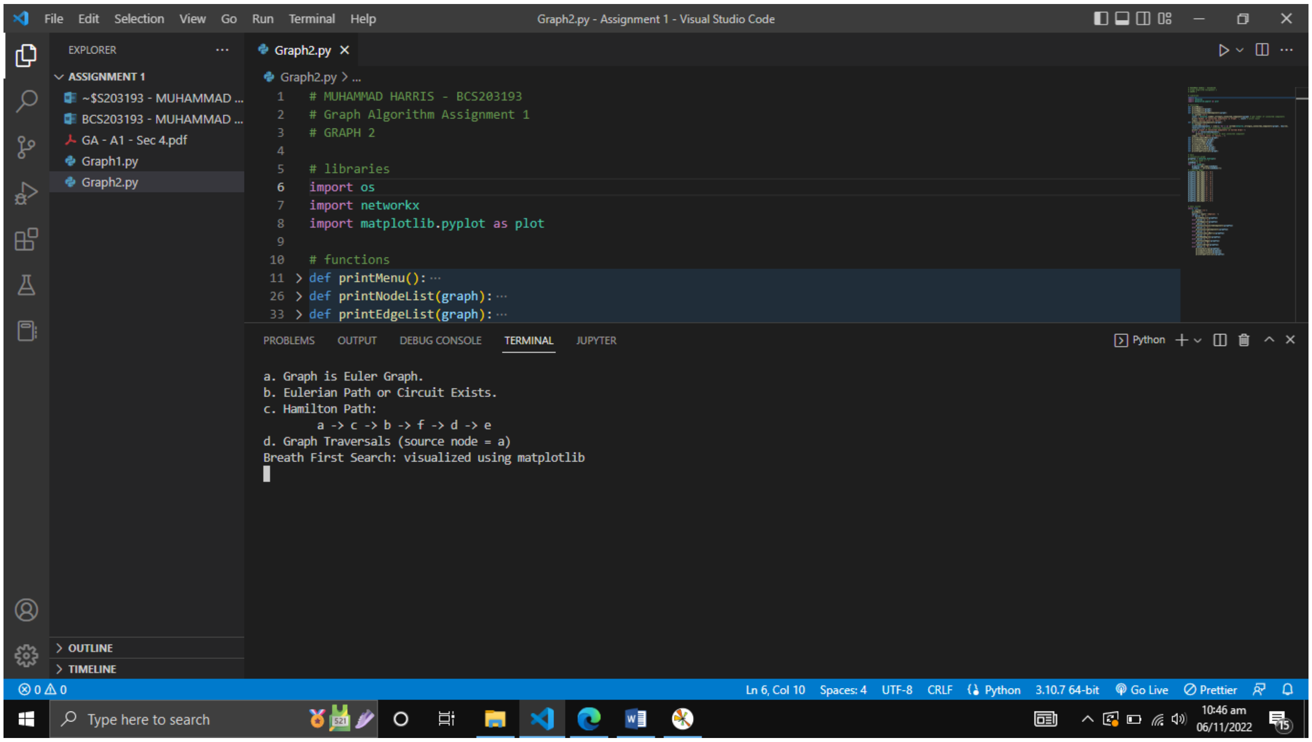
```
Number of Edges in Graph (parallel edges ignored):  15

press enter to return...
```

Output 7: number of edges in graph



```python
# MUHAMMAD HARRIS - BCS203193
# Graph Algorithm Assignment 1
# GRAPH 2
```

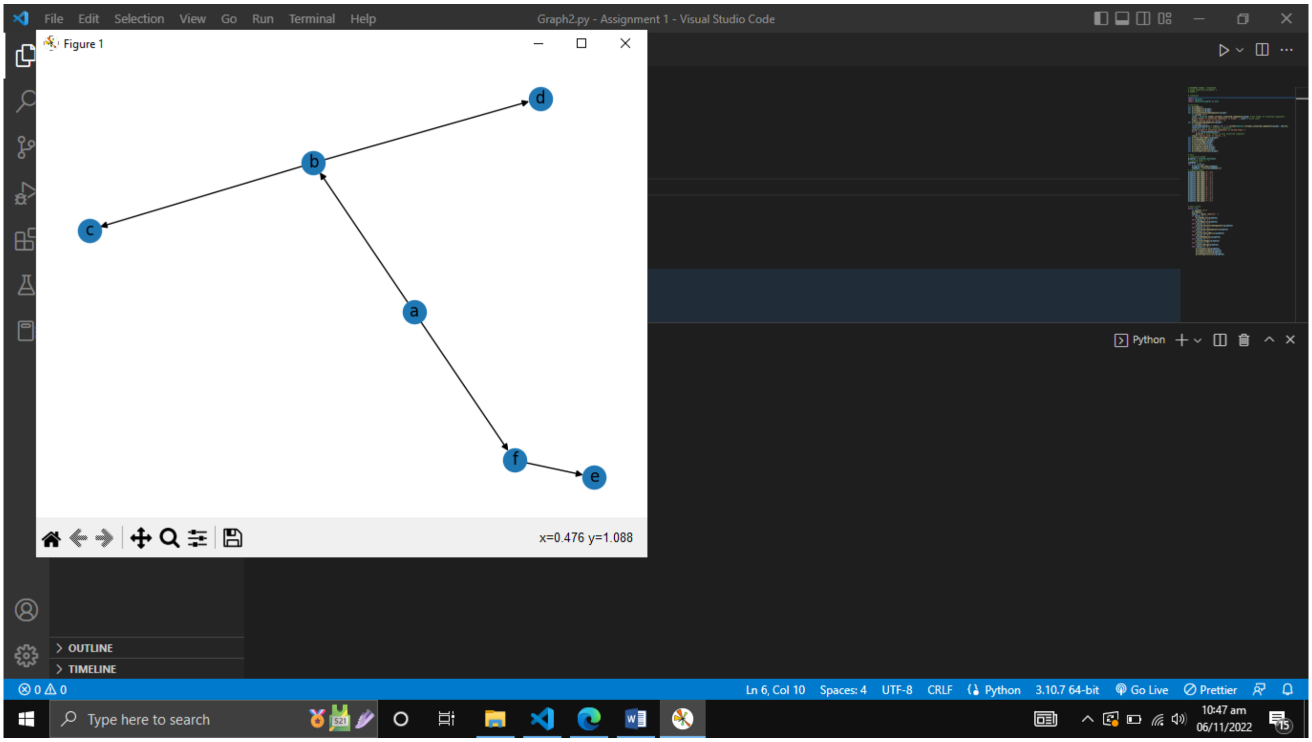
Output 8: graph visualization


File Edit Selection View Go Run Terminal Help
Graph2.py - Assignment 1 - Visual Studio Code
EXPLORER
ASSIGNMENT 1
~$S203193 - MUHAMMAD ...
BCS203193 - MUHAMMAD ...
GA - A1 - Sec 4.pdf
Graph1.py
Graph2.py
OUTLINE
TIMELINE
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER
Python
Ln 6, Col 10 Spaces: 4 UTF-8 CRLF Python 3.10.7 64-bit Go Live Prettier
Type here to search
10:46 am
06/11/2022

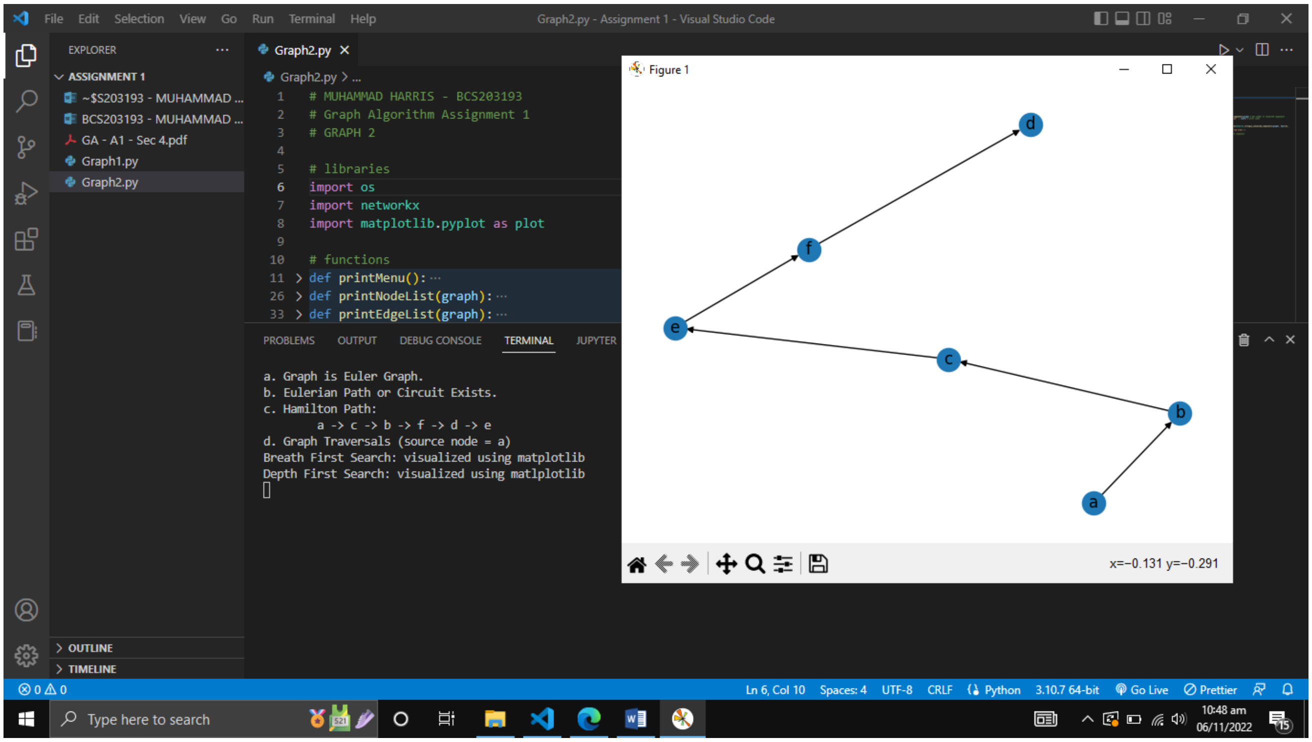
```
# MUHAMMAD HARRIS - BCS203193
# Graph Algorithm Assignment 1
# GRAPH 2

# libraries
import os
import networkx
import matplotlib.pyplot as plot

# functions
def printMenu(): ···
def printNodeList(graph): ···
def printEdgeList(graph): ···
```

```
a. Graph is Euler Graph.
b. Eulerian Path or Circuit Exists.
c. Hamilton Path:
        a -> c -> b -> f -> d -> e
d. Graph Traversals (source node = a)
Breath First Search: visualized using matplotlib
```

Output 9: eulerian & hamilton path


File Edit Selection View Go Run Terminal Help
Graph2.py - Assignment 1 - Visual Studio Code
Figure 1
d
b
c
a
f
e
x=0.476 y=1.088
OUTLINE
TIMELINE
Python
Ln 6, Col 10 Spaces: 4 UTF-8 CRLF Python 3.10.7 64-bit Go Live Prettier
Type here to search
10:47 am
06/11/2022

Output 10: BFS traversal



```python
# MUHAMMAD HARRIS - BCS203193
# Graph Algorithm Assignment 1
# GRAPH 2

# libraries
import os
import networkx
import matplotlib.pyplot as plot

# functions
def printMenu(): ···
def printNodeList(graph): ···
def printEdgeList(graph): ···
```

```
a. Graph is Euler Graph.
b. Eulerian Path or Circuit Exists.
c. Hamilton Path:
        a -> c -> b -> f -> d -> e
d. Graph Traversals (source node = a)
Breath First Search: visualized using matplotlib
Depth First Search: visualized using matlplotlib
```

Output 11: DFS traversal