AI LAB
ASSIGNMENT 5

by

AMMAR JAMIL (01-134231-010)

Submitted to: DR ARSHAD FARHAD
Date: 30/09/2025

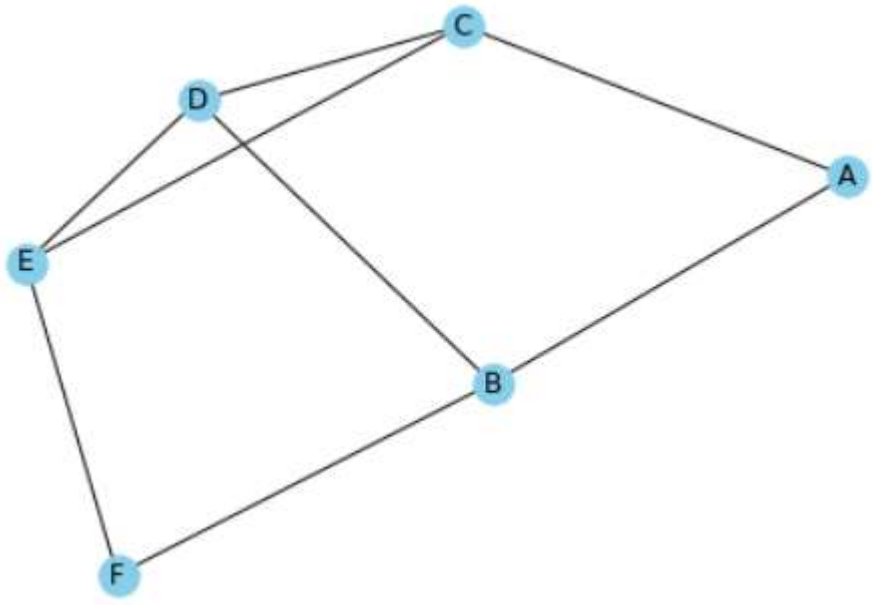
DEPARTMENT OF COMPUTER SCIENCE

BAHRIA UNIVERSITY ISLAMABAD E-8

Task 1: Construct an undirected graph representing a social network with at least six nodes (e.g., users) and eight edges (e.g., friendships). Utilize NetworkX to add nodes and edges, compute the degree of each node, and identify any isolated nodes if present. Visualize the graph using Matplotlib with custom node colors and labels. Also save figure in pdf with width = 6 inches and height = 4 inches. (plt.figure(figsize=(6, 4)) creates a figure of width 6 inches and height 4 inches.)

Code:

```
import networkx as nx
import matplotlib.pyplot as plt
G = nx.Graph()
G.add_edges_from([  ("A","B"), ("A","C"), ("B","D"), ("C","D"),
("C","E"), ("D","E"), ("E","F"), ("B","F")
])
plt.figure(figsize=(6,4))
nx.draw(G, with_labels=True, node_color="lightblue")
plt.savefig("figure1.pdf")
plt.show()
```



Task 2: Develop a directed weighted graph simulating a transportation network, incorporating at least five nodes (e.g., cities) and seven edges with weights representing distances or costs. Employ NetworkX to add weighted edges and calculate the shortest path between two specified nodes using Dijkstra's algorithm

(via nx.shortest_path). Generate a visualization with Matplotlib, including edge labels for weights and arrows for direction. Also save figure in pdf.

Code

```
import networkx as nx
import matplotlib.pyplot as plt
G = nx.DiGraph()
G.add_weighted_edges_from([  ("C1","C2",1), ("C2","C3,2), ("C3","C4",3),
("C4","C5",4), ("C1","C3",5), ("C2","C5",6),
("C5","C1",7)
])
path = nx.shortest_path(G, source="C1", target="C5", weight="weight")
print("Path :", path)
plt.figure(figsize=(6,4))
pos = {  "C1": (0, 1),  "C2": (2, 0), "C3": (2, 4),  "C4": (1, 2),  "C5": (3, 0)
}
nx.draw(G, pos, with_labels=True, node_color="lightblue", arrows=True)
labels = nx.get_edge_attributes(G, 'weight')
nx.draw_networkx_edge_labels(G, pos, edge_labels=labels)
plt.savefig("figure.pdf")
plt.show()
```

Task 3: Generate a random graph using NetworkX's built-in functions, such as an Erdos-Renyi graph with 10 nodes and a probability of 0.3 for edge creation. Compute the number of connected components and visualize the graph with Matplotlib..

Code:

```
import networkx as nx
import matplotlib.pyplot as plt
G = nx.erdos_renyi_graph(n=10, p=0.3)
components = nx.number_connected_components(G)
print("Total connected components are:", components)
```

```
plt.figure(figsize=(6,4))
nx.draw(G, with_labels=True, node_color="lightblue")
plt.savefig("figure3.pdf")
plt.show()
```