**Labsheet 4**

**Graph**

def add\_node(v):

    global node\_count

    if v in nodes :

        print(v,"is already in graph")

    else:

        node\_count = node\_count + 1

        nodes.append(v)

        for n in graph:

            n.append(0)

        temp = []

        for i in range(node\_count):

            temp.append(0)

        graph.append(temp)

def add\_edge(v1,v2,cost):

    if v1 not in nodes:

        print(v1,"is not present in graph")

    elif v2 not in nodes:

        print(v2,"is not present in graph")

    else:

        index1 = nodes.index(v1)         #all vertices of graph is store in node list

        index2 = nodes.index(v2)

        graph[index1][index2] = cost

        #graph[index2][index1] = cost     #undirected,weighted graph

def delete\_node(v):

    global node\_count

    if v not in nodes:

        print(v,"is not present in graph")

    else:

        index1 = nodes.index(v)    #store node which want to deleted

        node\_count = node\_count-1  #decrement node count

        nodes.remove(v)

        graph.pop(index1)

        for i in graph :

            i.pop(index1)

def delete\_edge(v1,v2):

    if v1 not in nodes:

        print(v1,"is not present in graph")

    elif v2 not in nodes:

        print(v2,"is not present in graph")

    else:

        index1 = nodes.index(v1)

        index2 = nodes.index(v2)

        graph[index1][index2] = 0

        graph[index2][index1] = 0

#as matrix

def print\_graph():

    for i in range(node\_count):

        for j in range(node\_count):

            print(format(graph[i][j],"<3"),end="")          #format function - to print adj matrix on shape

        print()

nodes = []   #store all nodes

graph = []   #store adj matrix

node\_count = 0

# print("Before adding nodes")

# print(nodes)

# print(graph)

add\_node("A")

add\_node("B")

add\_node("D")

add\_edge("A","B",7)

add\_edge("A","D",5)

delete\_node("D")

# print\_graph()

delete\_edge("A","B")

# print("After adding nodes")

print(nodes)

#print(graph)

print("adj matrix :")

print\_graph()

