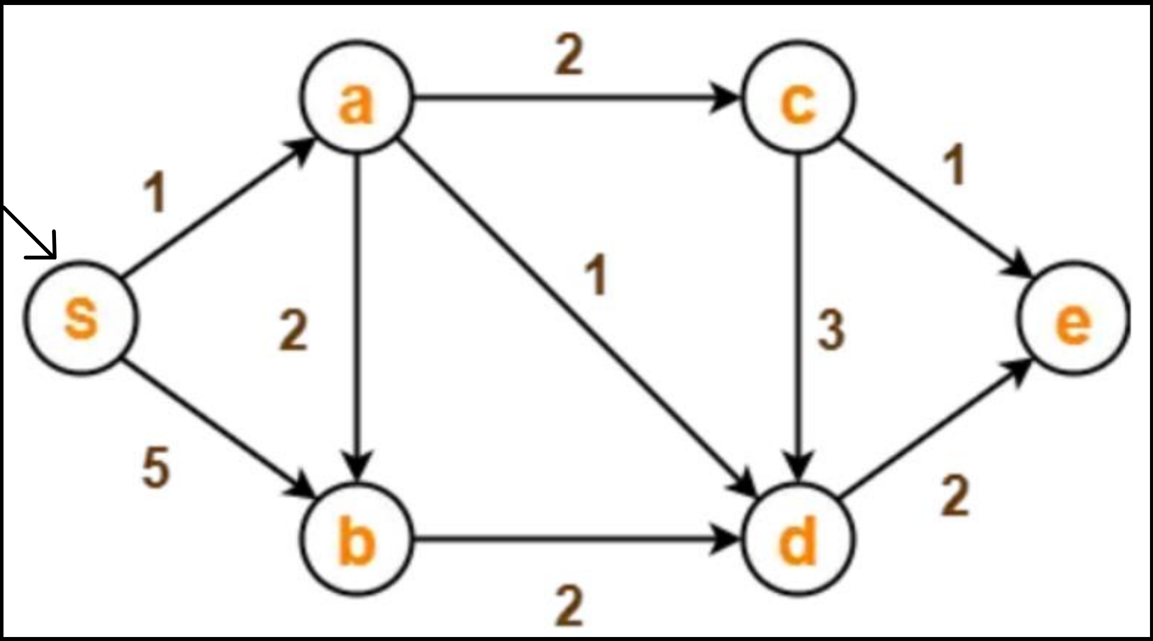
Name: Girish S

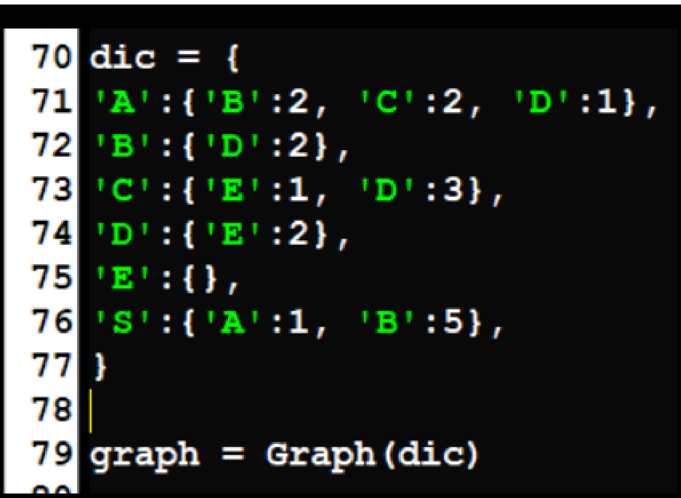
Roll No.: AM.EN.U4AIE22044

-22AIE203-

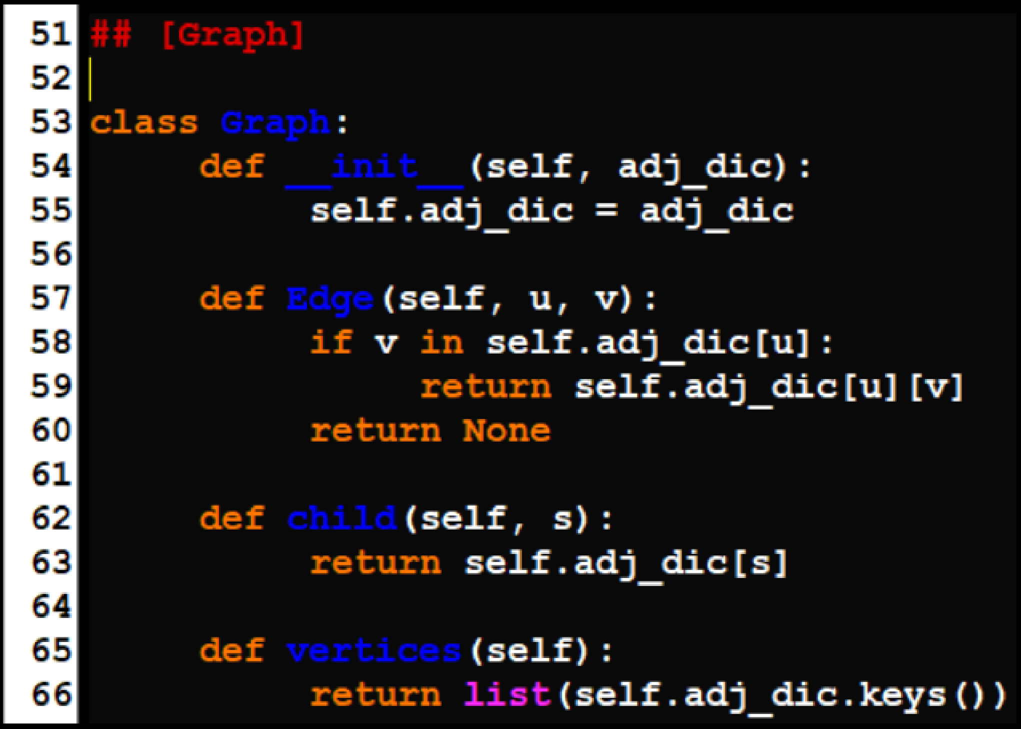
DATA STRUCTURES AND ALGORITHMS -2

ASSIGNMENT 2 – DIJKSTRA’S ALGORITHM



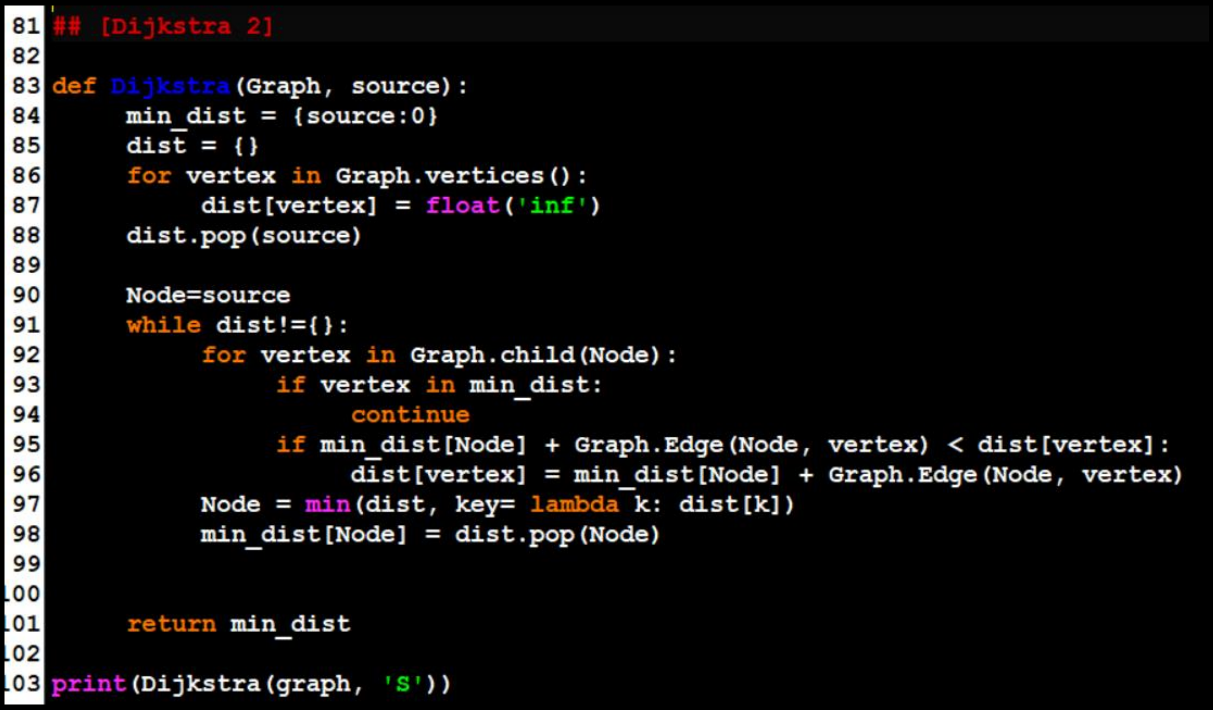


```
dic = {
'A':{'B':2, 'C':2, 'D':1},
'B':{'D':2},
'C':{'E':1, 'D':3},
'D':{'E':2},
'E':{},
'S':{'A':1, 'B':5},
}

graph = Graph(dic)
```

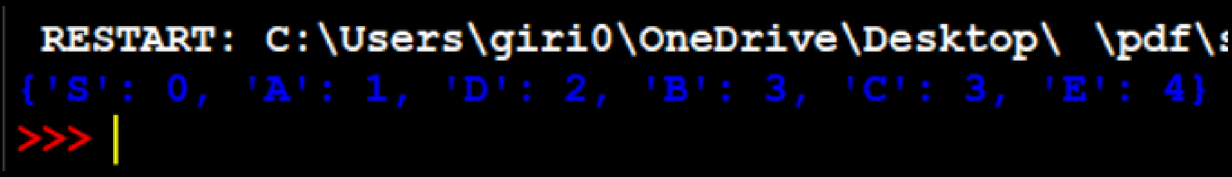


```python
## [Graph]

class Graph:
    def __init__(self, adj_dic):
        self.adj_dic = adj_dic

    def Edge(self, u, v):
        if v in self.adj_dic[u]:
            return self.adj_dic[u][v]
        return None

    def child(self, s):
        return self.adj_dic[s]

    def vertices(self):
        return list(self.adj_dic.keys())
```

1. **Perform Dijkstra's Algorithm on the given Graph using Adjacency matrix or Adjacency list**



```python
## [Dijkstra 2]

def Dijkstra(Graph, source):
    min_dist = {source:0}
    dist = {}
    for vertex in Graph.vertices():
        dist[vertex] = float('inf')
    dist.pop(source)

    Node=source
    while dist!={}:
        for vertex in Graph.child(Node):
            if vertex in min_dist:
                continue
            if min_dist[Node] + Graph.Edge(Node, vertex) < dist[vertex]:
                dist[vertex] = min_dist[Node] + Graph.Edge(Node, vertex)
        Node = min(dist, key= lambda k: dist[k])
        min_dist[Node] = dist.pop(Node)


    return min_dist

print(Dijkstra(graph, 'S'))
```



```
 RESTART: C:\Users\giri0\OneDrive\Desktop\ \pdf\
{'S': 0, 'A': 1, 'D': 2, 'B': 3, 'C': 3, 'E': 4}
>>> 
```