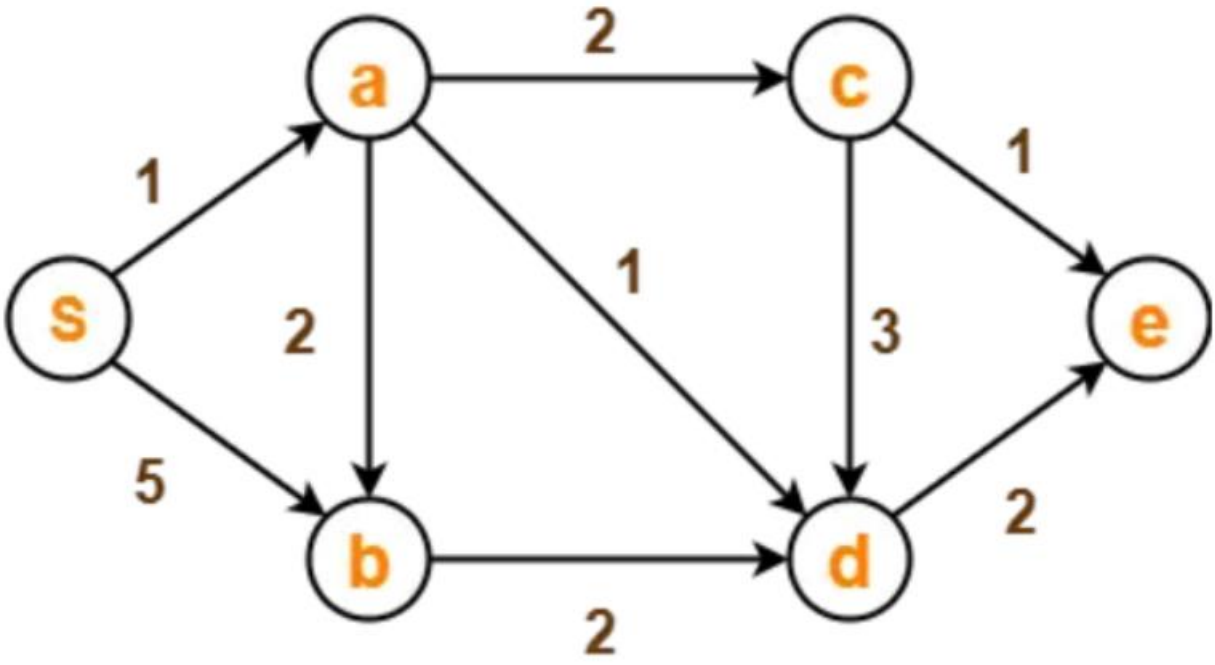
22AIE203 – DATA STRUCTURES & ALGORITHMS 2

ASSIGNMENT

Name : Anuvind MP

Roll no.: AM.EN.U4AIE22010



Using Dijkstra's Algorithm, find the shortest distance from source vertex 'S' to the remaining vertices in the given graph.

Code :

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

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

     def Edge(self, u, v):
          if v in self.adj_matrix[u]:
               return self.adj_matrix[u][v]
```

```
            return None

    def childNode(self, s):
            return self.adj_matrix[s]

    def vertices(self):
            return list(self.adj_matrix.keys())

graph = Graph(matrix)

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.childNode(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'))
```

Output :

```
{'S': 0, 'A': 1, 'D': 2, 'B': 3, 'C': 3, 'E': 4}
PS D:\BTECH\BTECH S03\DSA 2\LABSHEET\LAB 2>
```