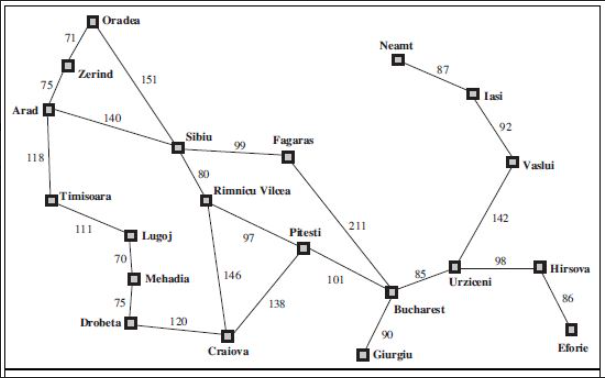
Practical No. 3

DFS & BFS Combined

**Q.1]** Implement DFS and BFS to the Romania Map in Python.

**Map:**

****

**Code:**

from collections import defaultdict

**class** Graph:

**def** \_\_init\_\_(self):

self.graph = defaultdict(list)

**def** addEdge(self, fromNode, toNode):

self.graph[fromNode].append(toNode)

self.graph[toNode].append(fromNode)

**def** doDFS(self, fromNode):

visited = set()

stack = list()

stack.append(fromNode)

print("Visited: ")

while (len(stack) > 0):

currentNode = stack[0]

stack.pop(0)

if currentNode in visited:

continue

visited.add(currentNode)

print(currentNode,end= "-> ")

for friendNode in self.graph[currentNode]:

if friendNode not in visited:

stack.insert(0, friendNode)

**def** doBFS(self, fromNode):

visited = set()

stack = list()

stack.append(fromNode)

visited.add(fromNode)

print("Visited: ")

while (len(stack) > 0):

currentNode = stack[0]

stack.pop(0)

""" if currentNode in visited:

continue """

""" visited.add(currentNode) """

print(currentNode,end= "-> ")

for friendNode in self.graph[currentNode]:

if friendNode not in visited:

visited.add(friendNode)

stack.append(friendNode)

graph = Graph()

graph.addEdge("Oradea", "Zerind")

graph.addEdge("Oradea", "Sibiu")

graph.addEdge("Zerind", "Arad")

graph.addEdge("Arad", "Sibiu")

graph.addEdge("Arad", "Timisoara")

graph.addEdge("Timisoara", "Lugoj")

graph.addEdge("Lugoj", "Mehadia")

graph.addEdge("Mehadia", "Drobeta")

graph.addEdge("Drobeta", "Craiova")

graph.addEdge("Craiova", "Rimnicu Vilcea")

graph.addEdge("Craiova", "Pitesti")

graph.addEdge("Rimnicu Vilcea", "Sibiu")

graph.addEdge("Rimnicu Vilcea", "Pitesti")

graph.addEdge("Sibiu", "Fagaras")

graph.addEdge("Fagaras", "Bucharest")

graph.addEdge("Pitesti", "Bucharest")

graph.addEdge("Bucharest", "Giurgiu")

graph.addEdge("Bucharest", "Urziceni")

graph.addEdge("Urziceni", "Hirsova")

graph.addEdge("Hirsova", "Eforie")

graph.addEdge("Urziceni", "Vaslui")

graph.addEdge("Vaslui", "Iasi")

graph.addEdge("Iasi", "Neamt")

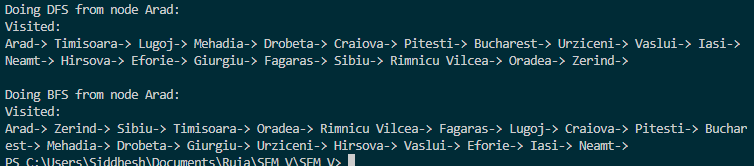
print("Doing DFS from node Arad:")

graph.doDFS('Arad')

print("Doing BFS from node Arad:")

graph.doBFS('Arad')

**Output:**

****