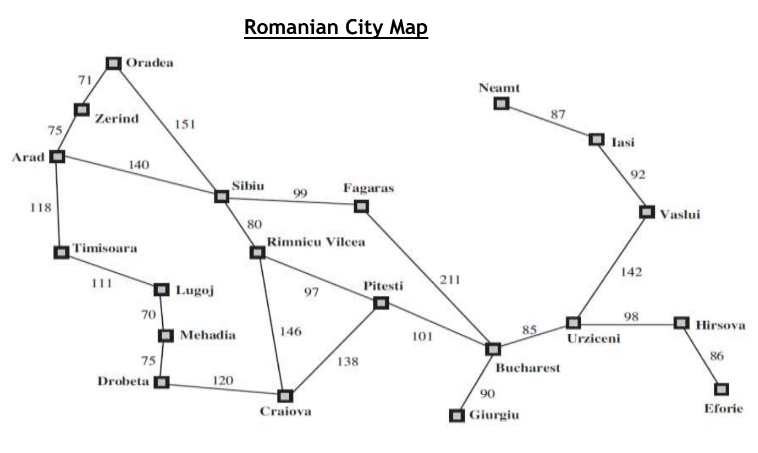
Practical No. 5

Uniform Cost Search

**Q.1]** Write a Python code to implement the Uniform Cost Search algorithm.

**Graph:**



**Code:**

from queue import PriorityQueue

graphy = {

'Arad': {'Sibiu': 140, 'Zerind': 75, 'Timisoara': 118},

'Zerind': {'Arad': 75, 'Oradea': 71},

'Oradea': {'Zerind': 71, 'Sibiu': 151},

'Sibiu': {'Arad': 140, 'Oradea': 151, 'Fagaras': 99, 'Rimnicu': 80},

'Timisoara': {'Arad': 118, 'Lugoj': 111},

'Lugoj': {'Timisoara': 111, 'Mehadia': 70},

'Mehadia': {'Lugoj': 70, 'Drobeta': 75},

'Drobeta': {'Mehadia': 75, 'Craiova': 120},

'Craiova': {'Drobeta': 120, 'Rimnicu': 146, 'Pitesti': 138},

'Rimnicu': {'Sibiu': 80, 'Craiova': 146, 'Pitesti': 97},

'Fagaras': {'Sibiu': 99, 'Bucharest': 211},

'Pitesti': {'Rimnicu': 97, 'Craiova': 138, 'Bucharest': 101},

'Bucharest': {'Fagaras': 211, 'Pitesti': 101, 'Giurgiu': 90, 'Urziceni': 85},

'Giurgiu': {'Bucharest': 90},

'Urziceni': {'Bucharest': 85, 'Vaslui': 142, 'Hirsova': 98},

'Hirsova': {'Urziceni': 98, 'Eforie': 86},

'Eforie': {'Hirsova': 86},

'Vaslui': {'Iasi': 92, 'Urziceni': 142},

'Iasi': {'Vaslui': 92, 'Neamt': 87},

'Neamt': {'Iasi': 87}

}

**def** doUSC(st, ed):

priory, visisted = PriorityQueue(), {}

priory.put((0, st, "{0} -> ".format(st)))

visisted[st] = True

while priory.qsize() > 0:

(curCost, curNode, path) = priory.get()

if (curNode == ed):

return (curCost, path)

else:

for eachChild in graphy[curNode].keys():

pathCost = curCost + graphy[curNode][eachChild]

if eachChild not in visisted or visisted[eachChild] >= pathCost:

visisted[eachChild] = pathCost

priory.put((pathCost, eachChild, path + "{0} -> ".format(eachChild)))

src = "Arad"

des = "Bucharest"

if input("want cutom input ('y'/'n'): ").lower() == "y":

src = input("Enter the source city: ")

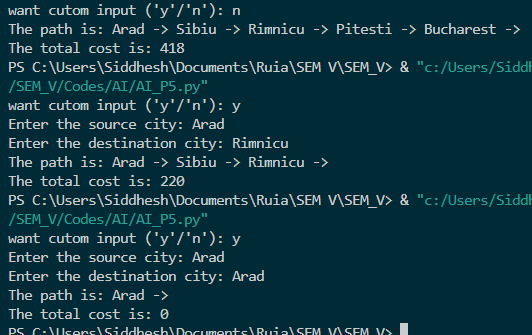
des = input("Enter the destination city: ")

(total, path) = doUSC(src, des)

print("The path is:",path)

print("The total cost is:",total)

**Output:**

****