**PROGRAM:**

from collections import deque

def bfs(start\_node, stop\_node):

open\_set = deque([start\_node])

visited = [0] \* n

parents = {start\_node: None}

while open\_set:

current\_node = open\_set.popleft()

if index\_node[current\_node] == stop\_node:

path = []

while current\_node is not None:

path.append(index\_node[current\_node])

current\_node = parents[current\_node]

path.reverse()

print('Path found:', ' -> '.join(path))

return path

visited[current\_node] = 1

for i in range(n):

if nodes[current\_node][i] == 1 and not visited[i]:

open\_set.append(i)

visited[i] = 1

parents[i] = current\_node

print('Path does not exist!')

return None

n = 20

nodes = [[0 for \_ in range(n)] for \_ in range(n)]

edges = [

('ORADEA', 'ZERIND'),

('ORADEA', 'SIBIU'),

('ZERIND', 'ARAD'),

('ARAD', 'SIBIU'),

('ARAD', 'TIMISOARA'),

('TIMISOARA', 'LUGOJ'),

('LUGOJ', 'MEHADIA'),

('MEHADIA', 'DROBETA'),

('DROBETA', 'CRAIOVA'),

('CRAIOVA', 'PITESTI'),

('CRAIOVA', 'RIMNICU\_VILCEA'),

('PITESTI', 'BUCHAREST'),

('BUCHAREST', 'GIURGIU'),

('BUCHAREST', 'URZICENI'),

('URZICENI', 'HIRSOVA'),

('HIRSOVA', 'EFORIE'),

('URZICENI', 'VASLUI'),

('VASLUI', 'IASI'),

('IASI', 'NEAMT'),

('BUCHAREST', 'FAGARAS'),

('SIBIU', 'FAGARAS'),

('SIBIU', 'RIMNICU\_VILCEA')

]

node\_index = {

'ORADEA': 0,

'ZERIND': 1,

'ARAD': 2,

'TIMISOARA': 3,

'LUGOJ': 4,

'MEHADIA': 5,

'DROBETA': 6,

'CRAIOVA': 7,

'PITESTI': 8,

'BUCHAREST': 9,

'GIURGIU': 10,

'URZICENI': 11,

'HIRSOVA': 12,

'EFORIE': 13,

'VASLUI': 14,

'IASI': 15,

'NEAMT': 16,

'FAGARAS': 17,

'SIBIU': 18,

'RIMNICU\_VILCEA': 19

}

index\_node = {v: k for k, v in node\_index.items()}

for edge in edges:

x, y = node\_index[edge[0]], node\_index[edge[1]]

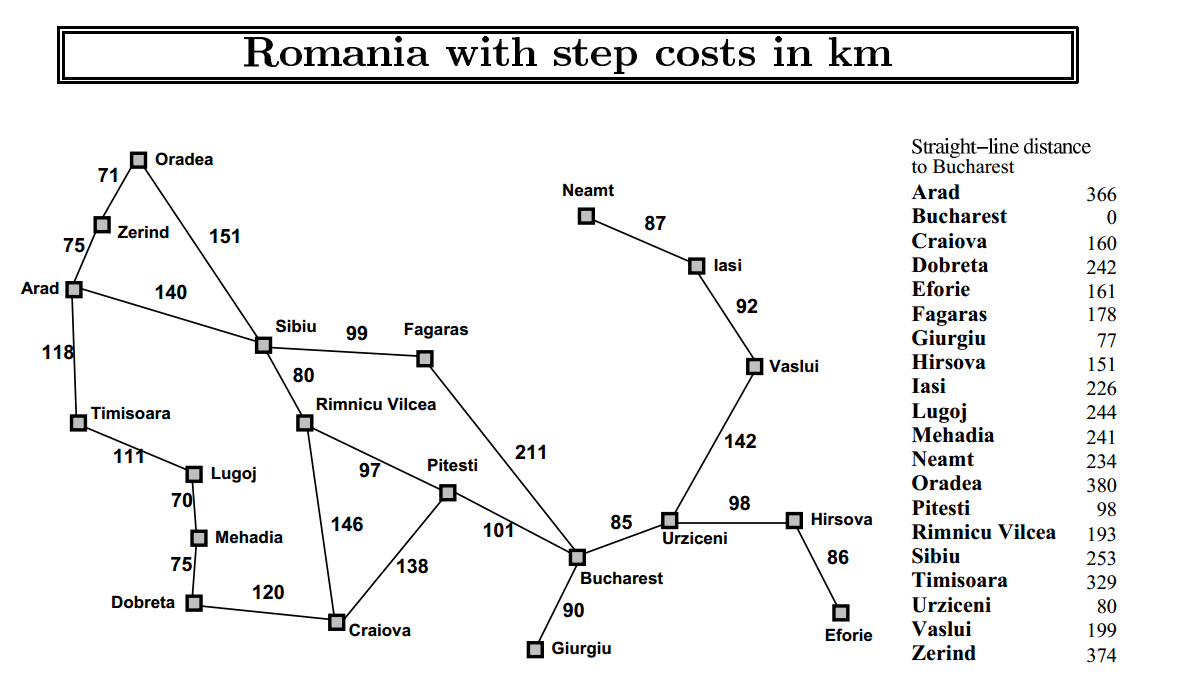
nodes[x][y] = nodes[y][x] = 1

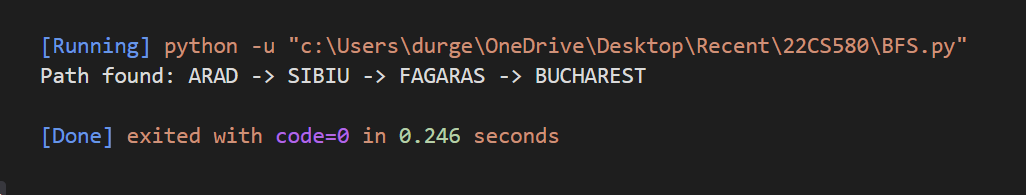
start\_node = node\_index['ARAD']

stop\_node = 'BUCHAREST'

bfs(start\_node, stop\_node)

**OUTPUT:**





**PROGRAM:**

def dfs(x, stop\_node, path):

path.append(index\_node[x])

if index\_node[x] == stop\_node:

print('Path found:', ' -> '.join(path))

return True

visited[x] = 1

for i in range(n):

if nodes[x][i] == 1 and not visited[i]:

if dfs(i, stop\_node, path):

return True

path.pop()

return False

n = 20

nodes = [[0 for \_ in range(n)] for \_ in range(n)]

edges = [

('ORADEA', 'ZERIND'),

('ORADEA', 'SIBIU'),

('ZERIND', 'ARAD'),

('ARAD', 'SIBIU'),

('ARAD', 'TIMISOARA'),

('TIMISOARA', 'LUGOJ'),

('LUGOJ', 'MEHADIA'),

('MEHADIA', 'DROBETA'),

('DROBETA', 'CRAIOVA'),

('CRAIOVA', 'PITESTI'),

('CRAIOVA', 'RIMNICU\_VILCEA'),

('PITESTI', 'BUCHAREST'),

('BUCHAREST', 'GIURGIU'),

('BUCHAREST', 'URZICENI'),

('URZICENI', 'HIRSOVA'),

('HIRSOVA', 'EFORIE'),

('URZICENI', 'VASLUI'),

('VASLUI', 'IASI'),

('IASI', 'NEAMT'),

('BUCHAREST', 'FAGARAS'),

('SIBIU', 'FAGARAS'),

('SIBIU', 'RIMNICU\_VILCEA')

]

node\_index = {

'ORADEA': 0,

'ZERIND': 1,

'ARAD': 2,

'TIMISOARA': 3,

'LUGOJ': 4,

'MEHADIA': 5,

'DROBETA': 6,

'CRAIOVA': 7,

'PITESTI': 8,

'BUCHAREST': 9,

'GIURGIU': 10,

'URZICENI': 11,

'HIRSOVA': 12,

'EFORIE': 13,

'VASLUI': 14,

'IASI': 15,

'NEAMT': 16,

'FAGARAS': 17,

'SIBIU': 18,

'RIMNICU\_VILCEA': 19

}

index\_node = {v: k for k, v in node\_index.items()}

for edge in edges:

x, y = node\_index[edge[0]], node\_index[edge[1]]

nodes[x][y] = nodes[y][x] = 1

visited = [0] \* n

start\_node = 'ARAD'

stop\_node = 'BUCHAREST'

path = []

if not dfs(node\_index[start\_node], stop\_node, path):

print('Path does not exist!')

**OUTPUT:**

