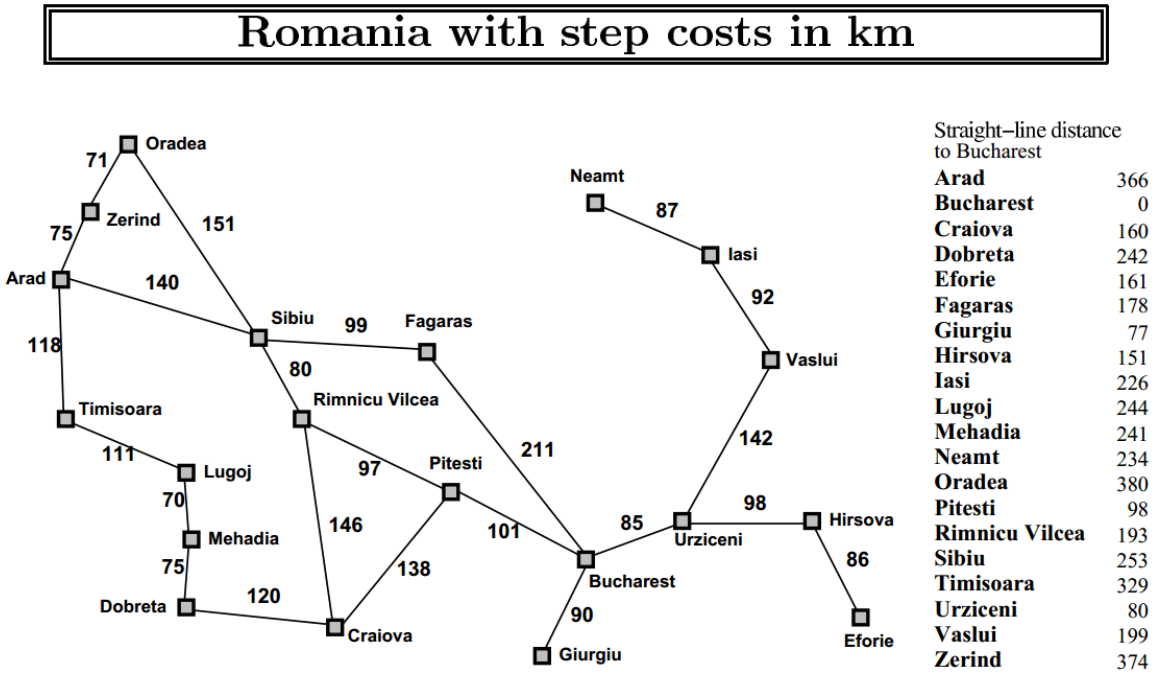
PROGRAM:

```
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:




```
[Running] python -u "c:\Users\durge\OneDrive\Desktop\Recent\22CS580\BFS.py"
Path found: ARAD -> SIBIU -> FAGARAS -> BUCHAREST

[Done] exited with code=0 in 0.246 seconds
```

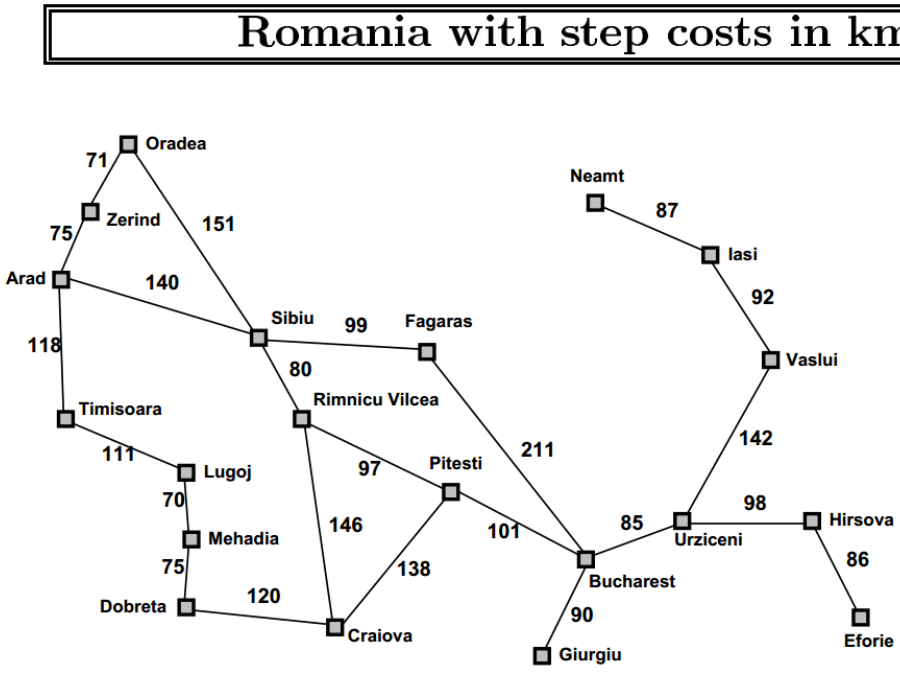
PROGRAM:

```
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:



Straight-line distance to Bucharest	
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	178
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	98
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

```
[Running] python -u "c:\Users\durge\OneDrive\Desktop\Recent\22CS580\DFS.py"
Path found: ARAD -> ZERIND -> ORADEA -> SIBIU -> FAGARAS -> BUCHAREST

[Done] exited with code=0 in 0.242 seconds
```