American International University-Bangladesh

Faculty Science & Technology

Mid Term Assignment

COURSE INSTRACTOR: DR. M M MANJURUL ISLAM

NAME	TAMIMUL ALAM
STUDENT ID	20-42215-1
SUBJECT	ARTIFICIAL INTELLIGENCE AND EXPERT SYSTEM
SECTION	A
DEPARTMENT	BSc CSE

SUBMISSION DATE:10/28/2021

Summarized report

Breadth first search: BFS is a graph traversal algorithm that requires you to start at a specific node (source or starting node) and traverse the graph layer by layer, thus examining the neighbor nodes (nodes which are directly connected to source node). Then proceed to the neighbor nodes on the following level.

Path Cost report:

Possible Paths:

```
['Arad', 'Sibiu', 'Fagaras', 'Bucharest'] 140

['Arad', 'Sibiu', 'Rimnicu Vilcea', 'Pitesti', 'Bucharest'] 140

['Arad', 'Sibiu', 'Rimnicu Vilcea', 'Pitesti', 'Bucharest'] 239

['Arad', 'Sibiu', 'Rimnicu Vilcea', 'Pitesti', 'Bucharest'] 450
```

Shortest Path:

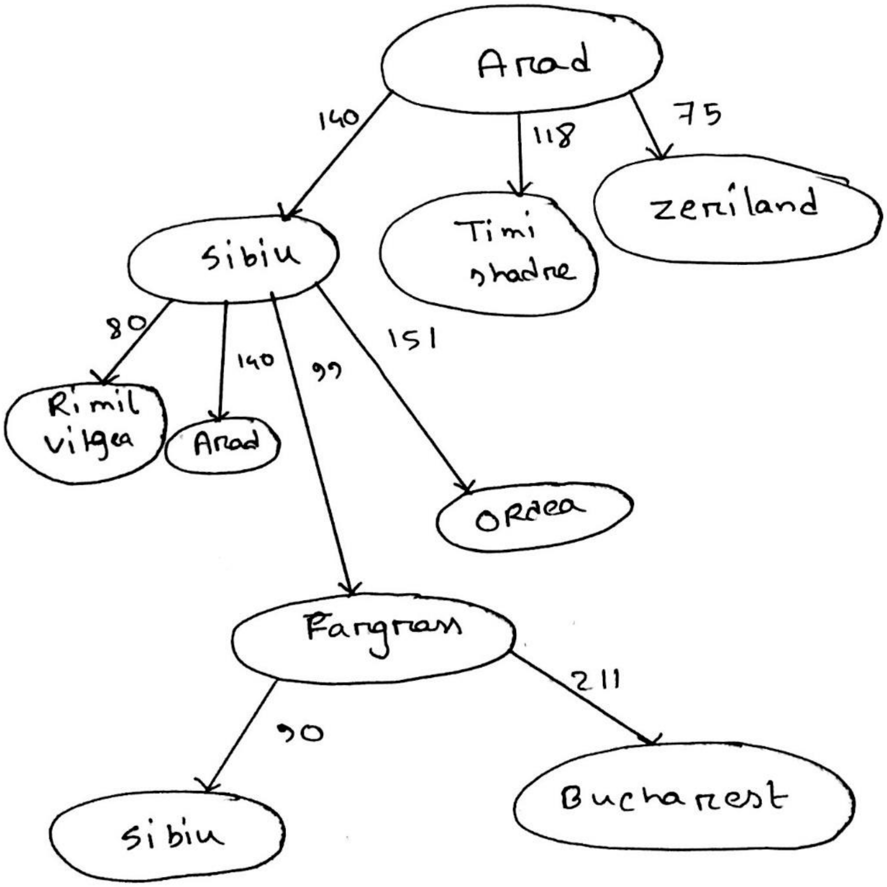
```
['Arad', 'Sibiu', 'Fagaras', 'Bucharest'] 140
```

(P T O)

Name: Tamimul Alam

ID: 20-42215-1

BFS:



Shortest path : Arad ⟶ sibiu ⟶ Fagarass ⟶ Bucharest

(Path cost 140)

Depth first search: The depth-first search (DFS) algorithm is used to traverse or explore data structures such as trees and graphs. The algorithm starts at the root node (in the case of a graph, any random node can be used as the root node) and examines each branch as far as feasible before retracing.

Path Cost report:

Possible paths:

```
['Arad', 'Timisoara', 'Lugoj', 'Mehadia', 'Drobeta', 'Craiova', 'Pitesti', 'Bucharest'] 118

['Arad', 'Timisoara', 'Lugoj', 'Mehadia', 'Drobeta', 'Craiova', 'Pitesti', 'Rimnicu Vilcea', 'Sibiu', 'Fagaras', 'Bucharest'] 118

['Arad', 'Timisoara', 'Lugoj', 'Mehadia', 'Drobeta', 'Craiova', 'Pitesti', 'Rimnicu Vilcea', 'Sibiu', 'Fagaras', 'Bucharest'] 229

['Arad', 'Timisoara', 'Lugoj', 'Mehadia', 'Drobeta', 'Craiova', 'Pitesti', 'Rimnicu Vilcea', 'Sibiu', 'Fagaras', 'Bucharest'] 299

['Arad', 'Timisoara', 'Lugoj', 'Mehadia', 'Drobeta', 'Craiova', 'Pitesti', 'Rimnicu Vilcea', 'Sibiu', 'Fagaras', 'Bucharest'] 374

['Arad', 'Timisoara', 'Lugoj', 'Mehadia', 'Drobeta', 'Craiova', 'Pitesti', 'Rimnicu Vilcea', 'Sibiu', 'Fagaras', 'Bucharest'] 494

['Arad', 'Timisoara', 'Lugoj', 'Mehadia', 'Drobeta', 'Craiova', 'Pitesti', 'Rimnicu Vilcea', 'Sibiu', 'Fagaras', 'Bucharest'] 632

['Arad', 'Timisoara', 'Lugoj', 'Mehadia', 'Drobeta', 'Craiova', 'Pitesti', 'Rimnicu Vilcea', 'Sibiu', 'Fagaras', 'Bucharest'] 733
```

Shortest path:

```
['Arad', 'Timisoara', 'Lugoj', 'Mehadia', 'Drobeta', 'Craiova', 'Pitesti', 'Bucharest'] 118
```

Greedy best first search: Expand the parent's first successor using a greedy method. Following the creation of a successor

i) If the successor's heuristic outperforms the parent's, the successor is moved to the front of the queue (with the parent reinserted exactly behind it) and the loop is restarted.

ii) If not, the successor is added to the queue (in a location determined by its heuristic value). The method will assess the parent's remaining successors if any.

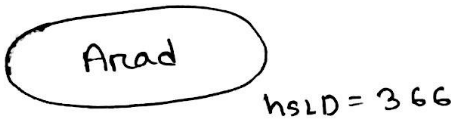
(P T O)

Greedy best first search

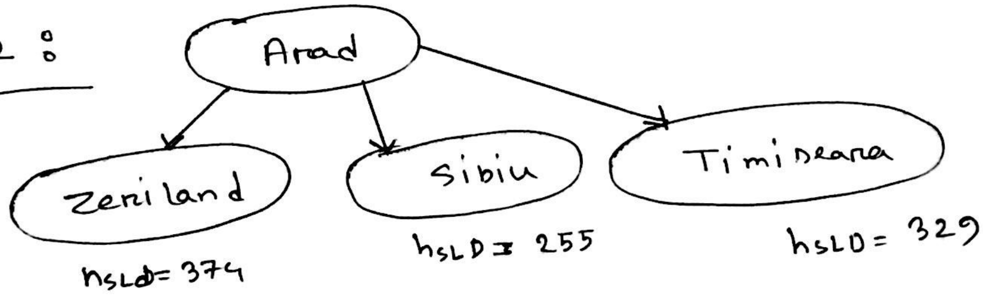
Goal is Bucharest →

Start node is Arad

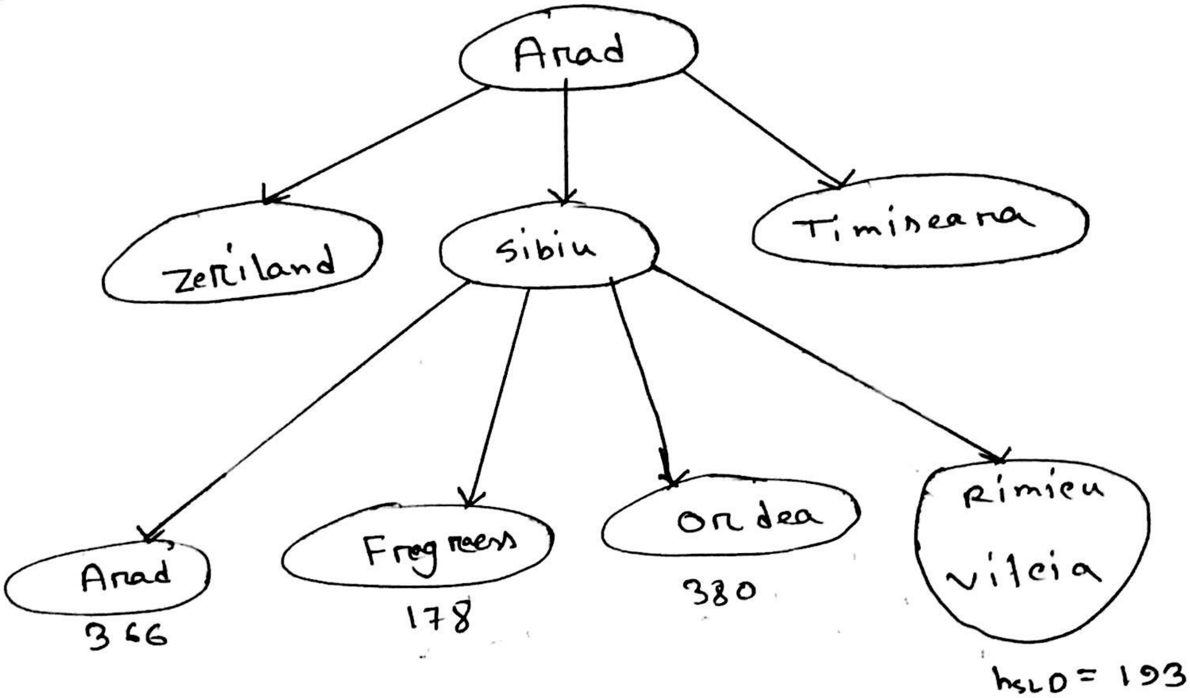
Step 1:



Step 2:

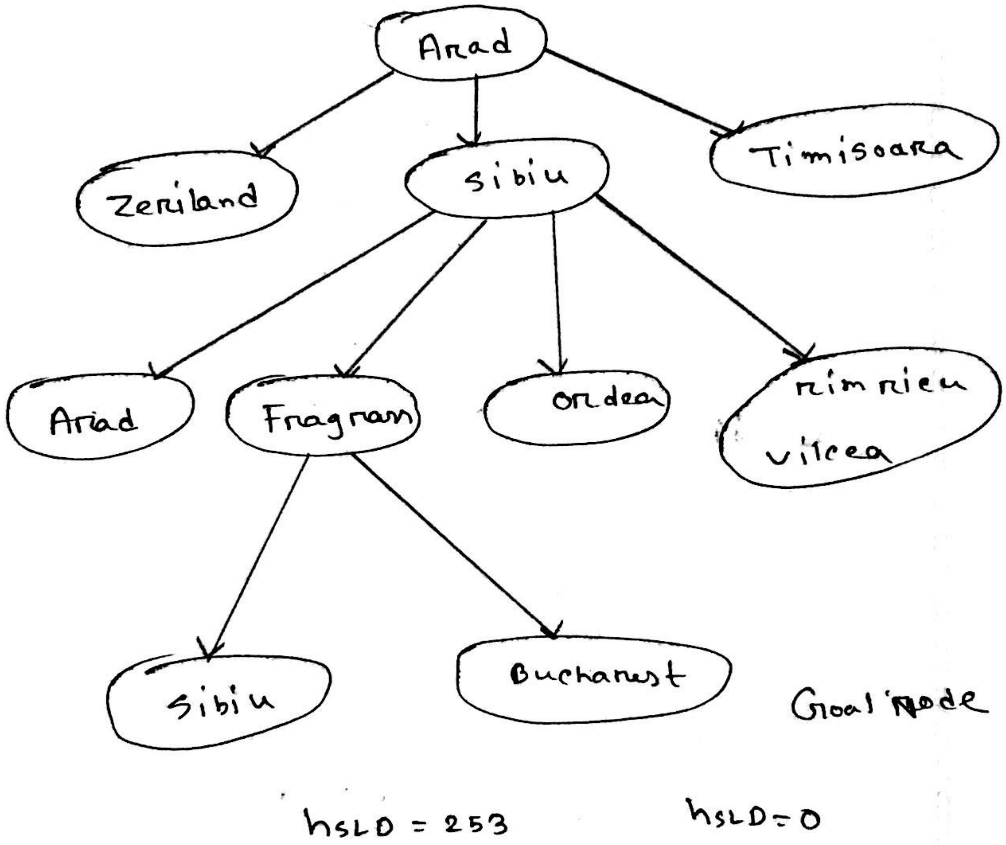


Step 3:

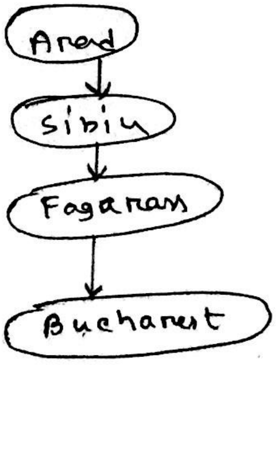


Expand with minimum h_{SLD}

Step 4 :



Optimum path is →



Edgcost

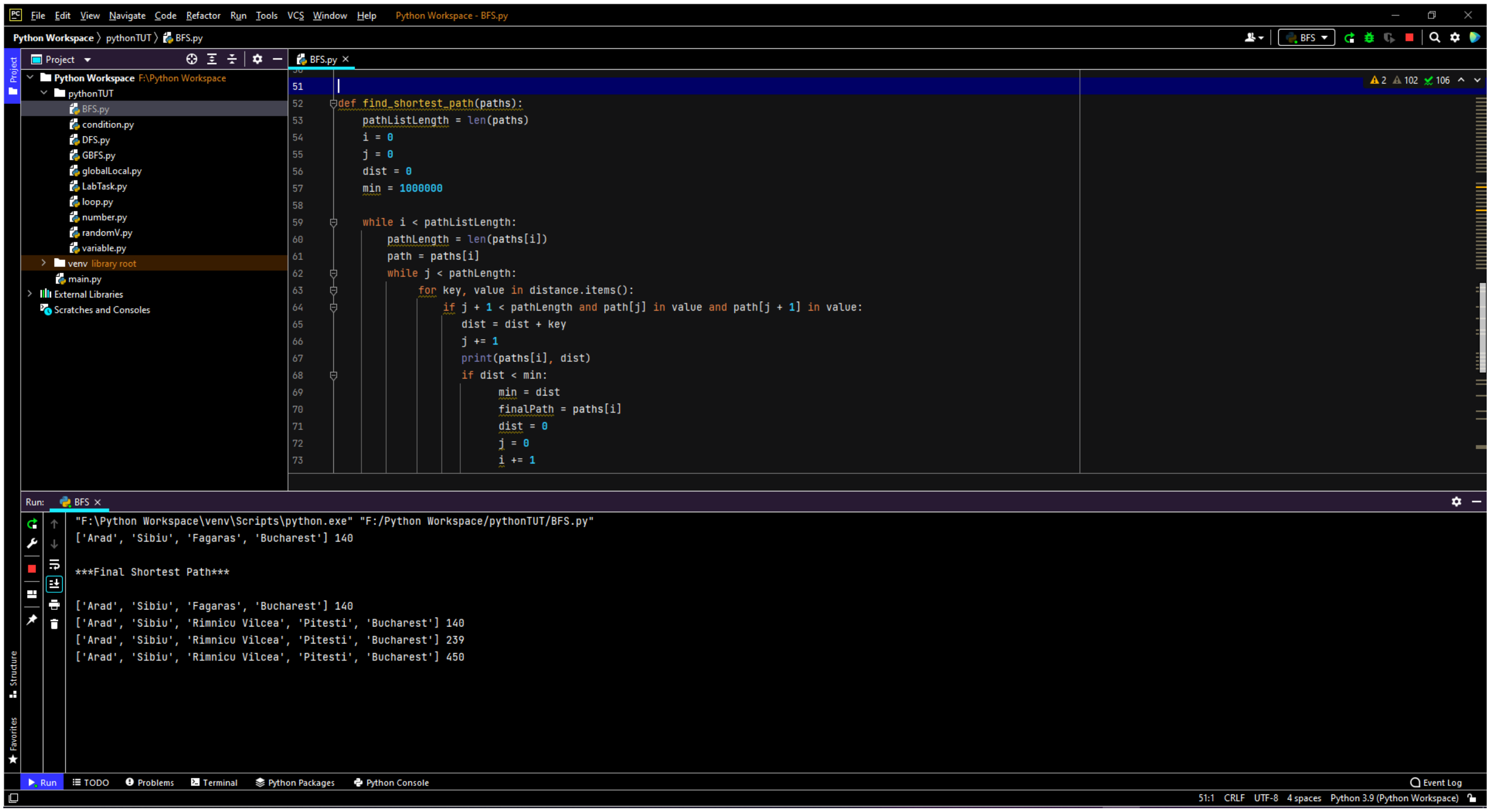
⇒ 140 + 99 + 211

= 450

GBFS

Output Screenshot

BFS

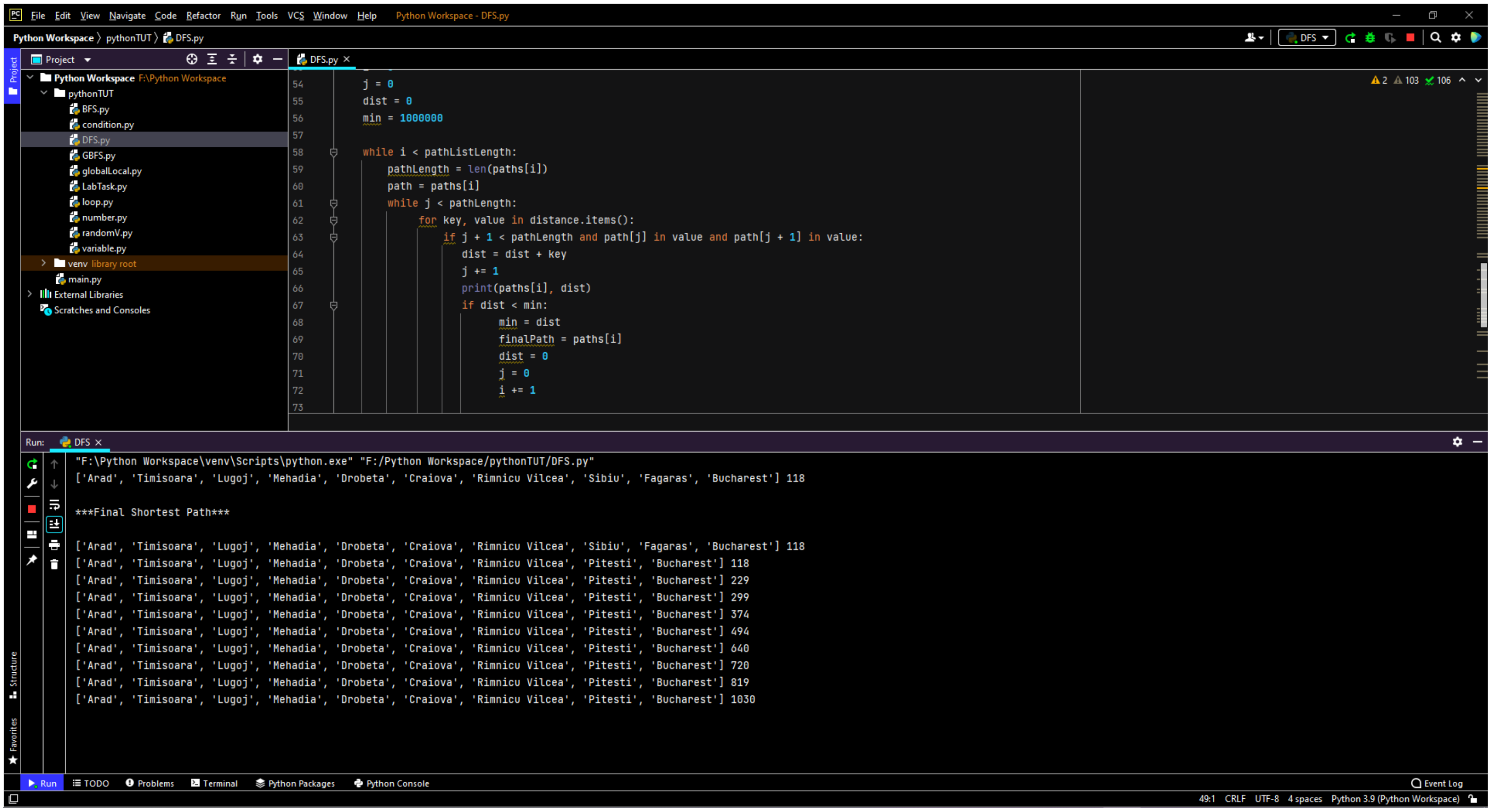


File Edit View Navigate Code Refactor Run Tools VCS Window Help
Python Workspace - BFS.py
Python Workspace 〉 pythonTUT 〉 BFS.py
Project
Python Workspace F:\Python Workspace
pythonTUT
BFS.py
condition.py
DFS.py
GBFS.py
globalLocal.py
LabTask.py
loop.py
number.py
randomV.py
variable.py
venv library root
main.py
External Libraries
Scratches and Consoles
BFS.py
Run: BFS
Run TODO Problems Terminal Python Packages Python Console
Event Log
51:1 CRLF UTF-8 4 spaces Python 3.9 (Python Workspace)

```

def find_shortest_path(paths):
    pathListLength = len(paths)
    i = 0
    j = 0
    dist = 0
    min = 1000000

    while i < pathListLength:
        pathLength = len(paths[i])
        path = paths[i]
        while j < pathLength:
             for key, value in distance.items():
                 if j + 1 < pathLength and path[j] in value and path[j + 1] in value:
                    dist = dist + key
                    j += 1
                    print(paths[i], dist)
                    if dist < min:
                          min = dist
                          finalPath = paths[i]
                          dist = 0
                          j = 0
                          i += 1
```

```
"F:\Python Workspace\venv\Scripts\python.exe" "F:/Python Workspace/pythonTUT/BFS.py"
['Arad', 'Sibiu', 'Fagaras', 'Bucharest'] 140

***Final Shortest Path***

['Arad', 'Sibiu', 'Fagaras', 'Bucharest'] 140
['Arad', 'Sibiu', 'Rimnicu Vilcea', 'Pitesti', 'Bucharest'] 140
['Arad', 'Sibiu', 'Rimnicu Vilcea', 'Pitesti', 'Bucharest'] 239
['Arad', 'Sibiu', 'Rimnicu Vilcea', 'Pitesti', 'Bucharest'] 450
```

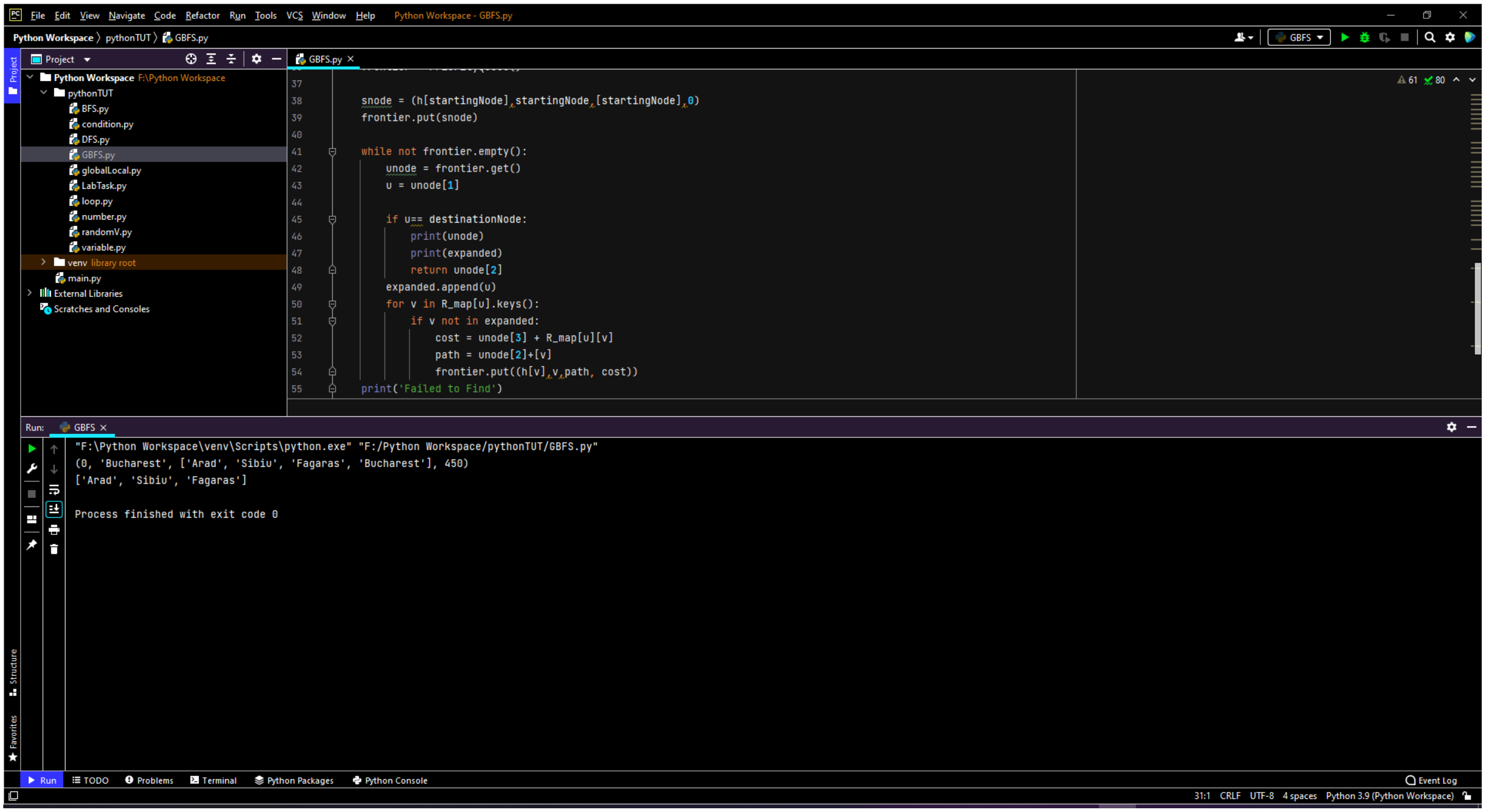
DFS


File Edit View Navigate Code Refactor Run Tools VCS Window Help
Python Workspace - DFS.py
Python Workspace 〉 pythonTUT 〉 DFS.py
Project
Python Workspace F:\Python Workspace
pythonTUT
BFS.py
condition.py
DFS.py
GBFS.py
globalLocal.py
LabTask.py
loop.py
number.py
randomV.py
variable.py
venv library root
main.py
External Libraries
Scratches and Consoles
DFS.py
Run: DFS
Run TODO Problems Terminal Python Packages Python Console
Event Log
49:1 CRLF UTF-8 4 spaces Python 3.9 (Python Workspace)

```python
    j = 0
    dist = 0
    min = 1000000

    while i < pathListLength:
        pathLength = len(paths[i])
        path = paths[i]
        while j < pathLength:
             for key, value in distance.items():
                 if j + 1 < pathLength and path[j] in value and path[j + 1] in value:
                    dist = dist + key
                    j += 1
                    print(paths[i], dist)
                    if dist < min:
                          min = dist
                          finalPath = paths[i]
                          dist = 0
                          j = 0
                          i += 1
```

```
"F:\Python Workspace\venv\Scripts\python.exe" "F:/Python Workspace/pythonTUT/DFS.py"
['Arad', 'Timisoara', 'Lugoj', 'Mehadia', 'Drobeta', 'Craiova', 'Rimnicu Vilcea', 'Sibiu', 'Fagaras', 'Bucharest'] 118

***Final Shortest Path***

['Arad', 'Timisoara', 'Lugoj', 'Mehadia', 'Drobeta', 'Craiova', 'Rimnicu Vilcea', 'Sibiu', 'Fagaras', 'Bucharest'] 118
['Arad', 'Timisoara', 'Lugoj', 'Mehadia', 'Drobeta', 'Craiova', 'Rimnicu Vilcea', 'Pitesti', 'Bucharest'] 118
['Arad', 'Timisoara', 'Lugoj', 'Mehadia', 'Drobeta', 'Craiova', 'Rimnicu Vilcea', 'Pitesti', 'Bucharest'] 229
['Arad', 'Timisoara', 'Lugoj', 'Mehadia', 'Drobeta', 'Craiova', 'Rimnicu Vilcea', 'Pitesti', 'Bucharest'] 299
['Arad', 'Timisoara', 'Lugoj', 'Mehadia', 'Drobeta', 'Craiova', 'Rimnicu Vilcea', 'Pitesti', 'Bucharest'] 374
['Arad', 'Timisoara', 'Lugoj', 'Mehadia', 'Drobeta', 'Craiova', 'Rimnicu Vilcea', 'Pitesti', 'Bucharest'] 494
['Arad', 'Timisoara', 'Lugoj', 'Mehadia', 'Drobeta', 'Craiova', 'Rimnicu Vilcea', 'Pitesti', 'Bucharest'] 640
['Arad', 'Timisoara', 'Lugoj', 'Mehadia', 'Drobeta', 'Craiova', 'Rimnicu Vilcea', 'Pitesti', 'Bucharest'] 720
['Arad', 'Timisoara', 'Lugoj', 'Mehadia', 'Drobeta', 'Craiova', 'Rimnicu Vilcea', 'Pitesti', 'Bucharest'] 819
['Arad', 'Timisoara', 'Lugoj', 'Mehadia', 'Drobeta', 'Craiova', 'Rimnicu Vilcea', 'Pitesti', 'Bucharest'] 1030
```

GBFS



```python

    snode = (h[startingNode],startingNode,[startingNode],0)
    frontier.put(snode)

    while not frontier.empty():
        unode = frontier.get()
        u = unode[1]

        if u== destinationNode:
            print(unode)
            print(expanded)
            return unode[2]
        expanded.append(u)
        for v in R_map[u].keys():
            if v not in expanded:
                cost = unode[3] + R_map[u][v]
                path = unode[2]+[v]
                frontier.put((h[v],v,path, cost))
    print('Failed to Find')
```

```
"F:\Python Workspace\venv\Scripts\python.exe" "F:/Python Workspace/pythonTUT/GBFS.py"
(0, 'Bucharest', ['Arad', 'Sibiu', 'Fagaras', 'Bucharest'], 450)
['Arad', 'Sibiu', 'Fagaras']

Process finished with exit code 0
```