Name: Soumitri Chattopadhyay

Roll: 001911001083

Department: Information Technology

Year: UG-III (5th Semester)

Subject: Artificial Intelligence Laboratory

Assignment: 4.1 : UCS and Iterative Lengthening

Date: 25.09.2021

Assignment Details:

Please use the following graph for the assignment. [Attached in Classroom announcement]

Things to do:

1. Implement both the algorithms

2. Execute both the implementations on traveling in Romania problem.

3. Write a document based on your observations of outputs of point 2 and compare and comment on the output..

Observations and Discussion:

1. **Input file: input_Graph_83.txt**

```
20

Arad Bucharest 500

Arad              0   75    0  118    0    0    0  140    0    0    0    0
0    0    0    0    0    0    0    0

Zerind           75    0   71    0    0    0    0    0    0    0    0    0
0    0    0    0    0    0    0    0

Oradea            0   71    0    0    0    0    0  151    0    0    0    0
0    0    0    0    0    0    0    0

Timisoara       118    0    0    0  111    0    0    0    0    0    0    0
0    0    0    0    0    0    0    0

Lugoj             0    0    0  111    0   70    0    0    0    0    0    0
0    0    0    0    0    0    0    0
```

```
Mehadia           0    0    0    0   70    0   75    0    0    0    0    0
0    0    0    0    0    0    0    0

Dobreata          0    0    0    0    0   75    0    0    0    0  120    0
0    0    0    0    0    0    0    0

Shibiu          140    0  151    0    0    0    0    0   80    0    0   99
0    0    0    0    0    0    0    0

Rimnicu-Vilcea    0    0    0    0    0    0    0   80    0   97  146    0
0    0    0    0    0    0    0    0

Pitesti           0    0    0    0    0    0    0    0   97    0  138    0
101    0    0    0    0    0    0    0

Craiova           0    0    0    0    0    0  120    0  146  138    0    0
0    0    0    0    0    0    0    0

Fagaras           0    0    0    0    0    0    0   99    0    0    0    0
211    0    0    0    0    0    0    0

Bucharest         0    0    0    0    0    0    0    0    0  101    0  211
0   90   85    0    0    0    0    0

Giurgiu           0    0    0    0    0    0    0    0    0    0    0    0
90    0    0    0    0    0    0    0

Urziceni          0    0    0    0    0    0    0    0    0    0    0    0
85    0    0  142    0    0   98    0

Vaslui            0    0    0    0    0    0    0    0    0    0    0    0
0    0  142    0   92    0    0    0

Iasi              0    0    0    0    0    0    0    0    0    0    0    0
0    0    0   92    0   87    0    0

Neamt             0    0    0    0    0    0    0    0    0    0    0    0
0    0    0    0   87    0    0    0

Hirsova           0    0    0    0    0    0    0    0    0    0    0    0
0    0   98    0    0    0    0   86

Eforie            0    0    0    0    0    0    0    0    0    0    0    0
0    0    0    0    0    0   86    0
```

2. Output

[Uniform Cost Search]

The Path between Arad and Bucharest is [Arad] ---> [Shibiu] ---> [Rimnicu-Vilcea] ---> [Pitesti] ---> [Bucharest]
The Distance between source and destination is 418.0

[Iterative Lengthening Search]

Source city: Arad

```
Destination city: Bucharest
Cost of optimal path: 500.0

For cost limit: 0.0
No Path Found
For cost limit: 0.0
No Path Found
For cost limit: 75.0
No Path Found
For cost limit: 118.0
No Path Found
For cost limit: 140.0
No Path Found
For cost limit: 146.0
No Path Found
For cost limit: 220.0
No Path Found
For cost limit: 229.0
No Path Found
For cost limit: 239.0
No Path Found
For cost limit: 299.0
No Path Found
For cost limit: 317.0
No Path Found
For cost limit: 366.0
No Path Found
For cost limit: 374.0
No Path Found
For cost limit: 418.0

Path Taken:
[Arad] -> [Shibiu] -> [Rimnicu-Vilcea] -> [Pitesti] ->
[Bucharest]
```

3. Discussion and Comparison

Uniform Cost Search	**Iterative Lengthening Search**

In UCS, all nodes are put into priority queue and then searched for. Thus, a lot of memory is used.	In ILS, the repeatedly with therefore for each iteration it has to explore already explored graph, which leads to greater execution time of the algorithm.