Indian Institute of Technology Roorkee
Department of Computer Science and Engineering

CSN-261: Data Structures Laboratory
(Autumn 2019-2020)
Lab Assignment-9 (L9)

Name :- Harshit Verma **Enroll no:- 18114029**
Branch :- CSE **Sub Batch :- O2**

Problem Statement -1:

Solve rat in a maze problem using A* algorithm in Python 3. Given a maze along with the source and destination, find the shortest path that the rat must traverse to reach the destination.

A Maze is given as N*M binary matrix of blocks where source block and destination block are given as the starting point and the destination point for a rat respectively. A rat starts from source and has to reach the destination. The rat can move only in any directions including diagonally. In a maze matrix, 0 means the block is a dead end or a barrier and 1 means the block can be used in the path from source to destination.

Data structure that must be used: Fibonacci heap (https://pypi.org/project/fibheap/)

Ask for an option for approximation heuristic that shall be used for distance calculation:

1. Manhattan Distance
2. Diagonal Distance
3. Euclidean Distance

Input: A maze in 2D python list (i.e., list of list) in a Pickle format file, “p1.pkl”

Source → (x_s, y_s)

Destination → (x_d , y_d)

Output: Number steps in the found shortest distance from source to destination along with

the followed path in format given as:

$(x_s, y_s) \rightarrow (x_1 , y_1) \rightarrow \ldots \rightarrow (x_k , y_k) \rightarrow (x_d , y_d)$,

where, x implies row number and y implies column number with starting index as 0, (x_s, y_s) is the source, (x_d , y_d) is the destination and (x_k , y_k) are all intermediate blocks.

Also, visualize the given maze in a 2D matrix (heatmap) where a block which is a barrier is shown in black color, blocks that can be used in the path in white color and the blocks which are present in the found path are shown in blue color. Also, highlight source block in green color and destination block in red color.

DATA STRUCTURES USED:

- Fibonacci Heap
- List
- 2-D arrays

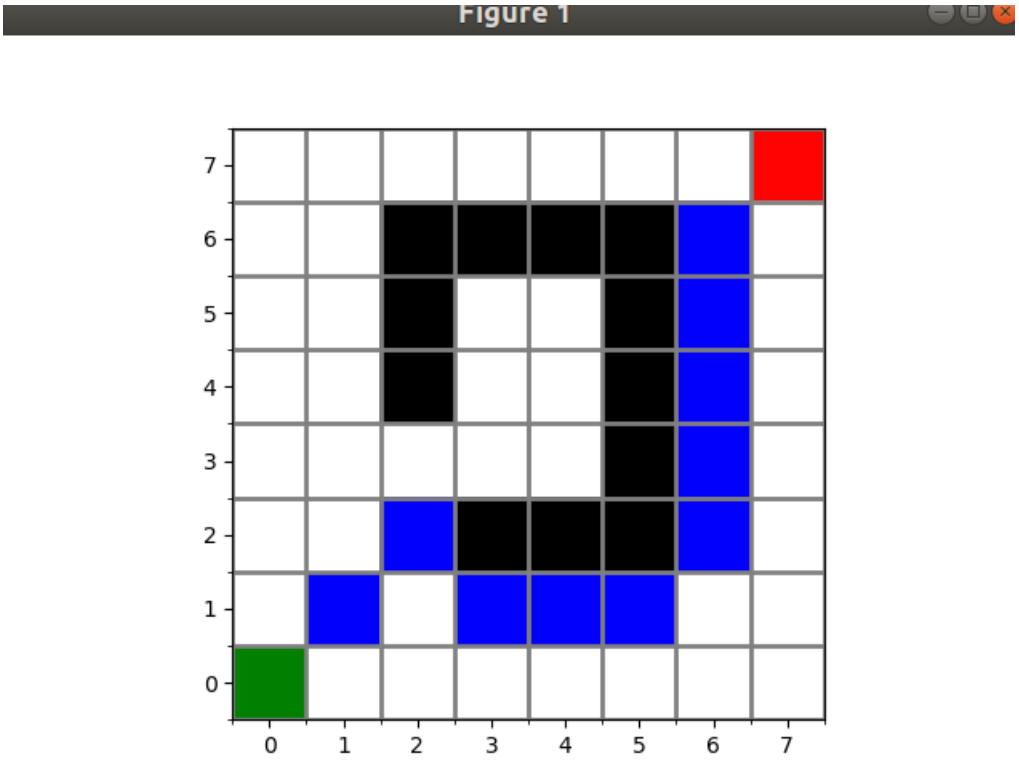
ALGORITHMS USED:

- A* search algorithm

```
thefox@thebunker:~/Desktop/CSN261_Assign/csn261_assign9/ques1
$ python3 ques1.py L9_P1_input.pkl

Please Choose A Heuristic For The A* Algorithm:
        1. Manhattan Distance
        2. Diagonal Distance
        3. Euclidean Distance

3
Enter source coordinates separated by a space : 0 0
Enter destination coordinates separated by a space : 7 7
Steps : 11
(0,0)->(1,1)->(2,2)->(3,1)->(4,1)->(5,1)->(6,2)->(6,3)->(6,4)->(6,5)->(6
,6)->(7,7)
```



Problem Statement -2:

Given a Directed Graph identify if the graph is a DAG (Directed Acyclic Graph) or not? If yes, then print the Topological sorting for given DAG. Implement this problem in Python 3.
Data structures that must be used: Set, list, Stack (list based implementation in python)
Note: Use network package in python for this problem
Input: A gpickle format networkx directed graph input file named “p2.gpickle”. Each node has a unique label.
Output: Topological sorting for given input graph, if the graph is a DAG else print “the given input is a not a DAG”. While printing the Topological sort print the label of the node.

DATA STRUCTURES USED:

- List
- Stack
- Graphs

ALGORITHMS USED:

- Depth First Search
- Topological Sort

```
thefox@thebunker:~/Desktop/CSN261_Assign/csn261_assign9/ques2
$ python3 ques2.py L9_P2_input.gpickle
Topological Sorting :
4,5,2,3,1,0
```