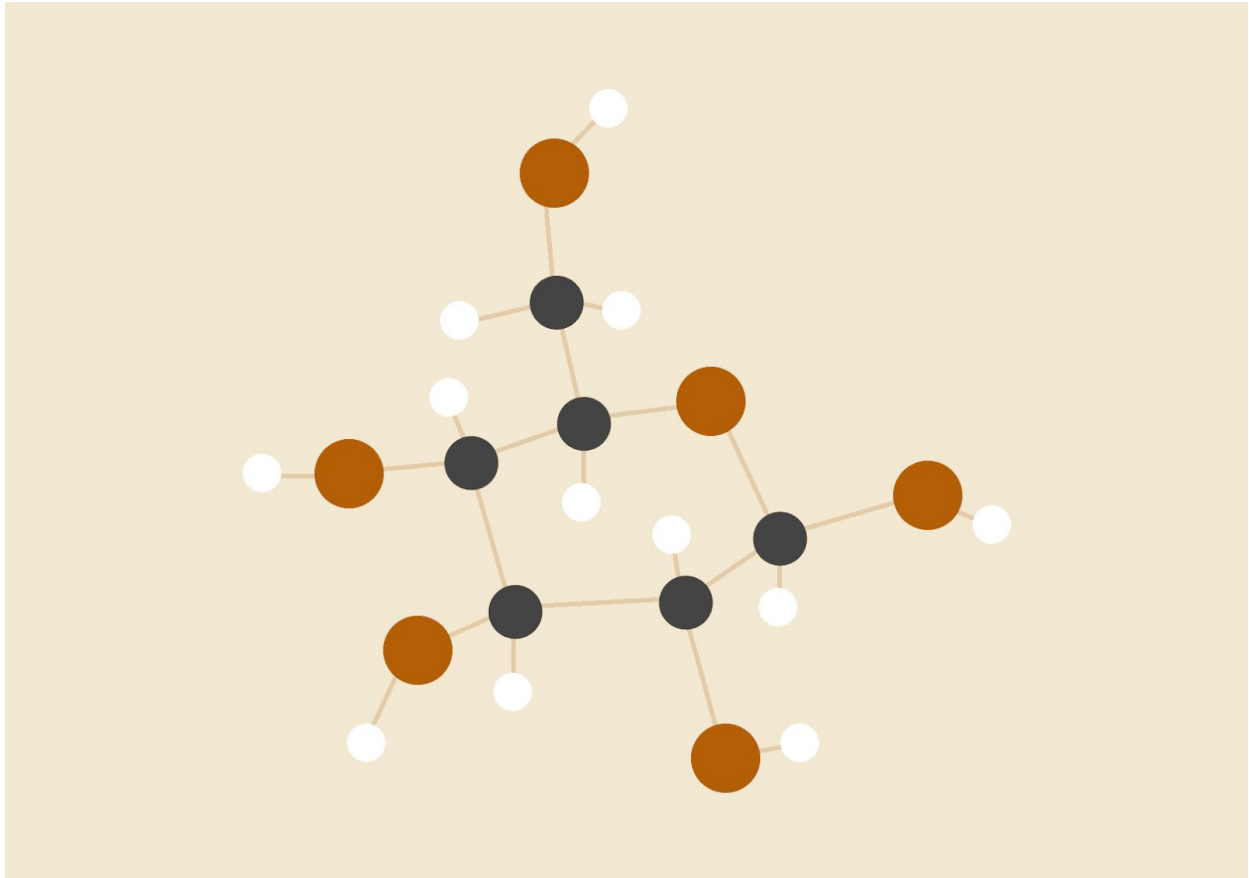# ASSIGNMENT 9

Data Structures Laboratory

Rishi Chordia

18118052

BTech CSE

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 $\rightarrow$ (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) $\rightarrow$ $\cdots$ $\rightarrow$ (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.
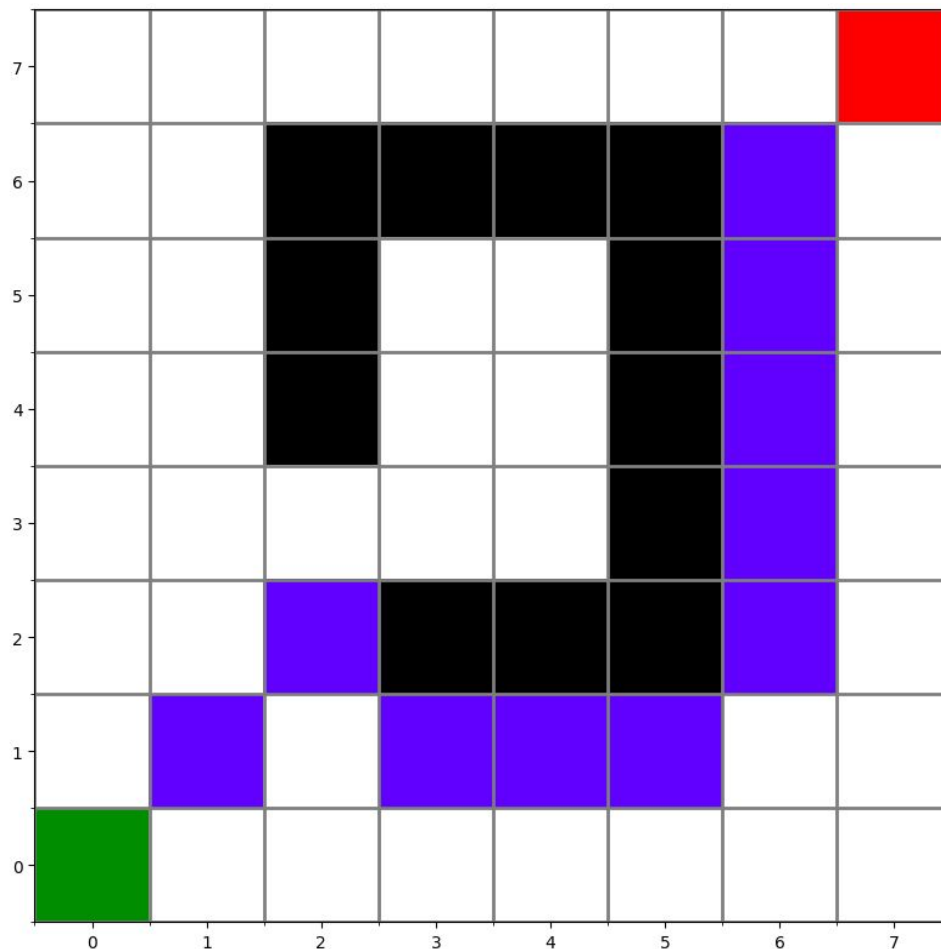
Algorithms and Implementation:

- A* search algorithm was used.
- Fibonacci heap,List and 2-D arrays were also used.

Snapshots and Computation Time

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)

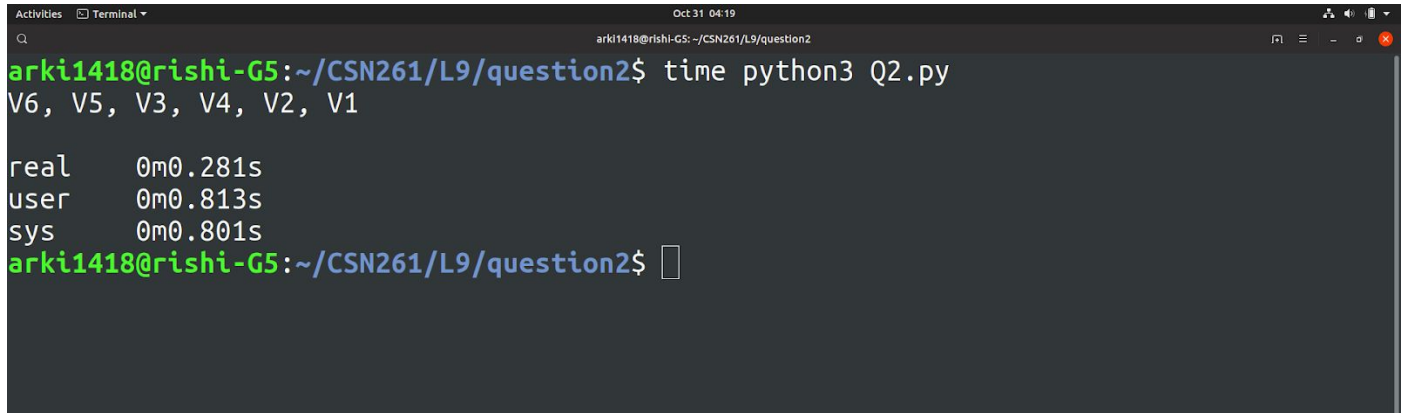Note: Use network package in python for this problem

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.

Algorithms and Implementation:

- **Depth First Search algorithm has been used**
- **Topological Sort is done using modified DFS**
- **Stacks have been implemented using Lists**
- **Graphs and Sets data structures have also been used**

Snapshots and Computation Time

```
Activities    Terminal ▾                              Oct 31 04:19
                              arki1418@rishi-G5: ~/CSN261/L9/question2
arki1418@rishi-G5:~/CSN261/L9/question2$ time python3 Q2.py
V6, V5, V3, V4, V2, V1

real    0m0.281s
user    0m0.813s
sys     0m0.801s
arki1418@rishi-G5:~/CSN261/L9/question2$
```