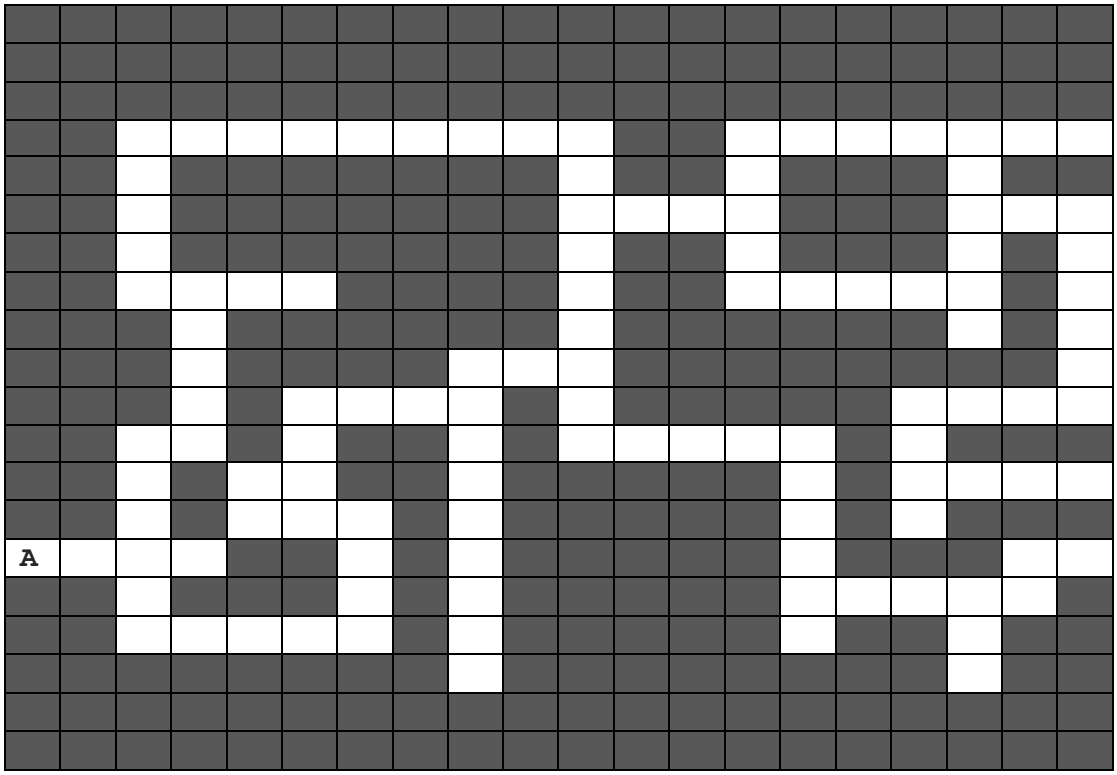
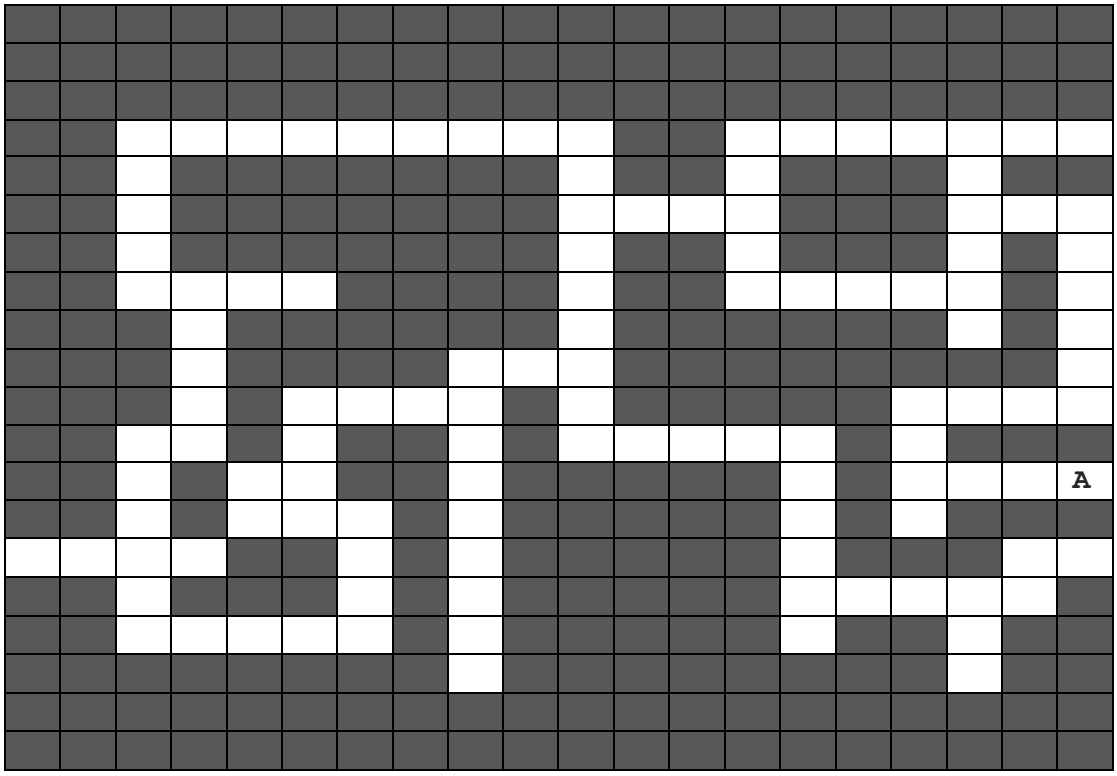
Consider you have a Maze that needs to be solved. The goal of an agent is to complete the Maze in fixed number of moves. Your implementation should be modular so that the sensors, actuators and environment characteristics (size etc.) can be easily changed. You need to implement a graph-based search agent with visited state check. You can use the provided agents.py file which contains the abstract implementation of agent and environment. You are required to formulate this problem by defining the four parameters:

1. Initial State
2. Successor
3. Goal Test
4. Path cost



Initial State



Final State

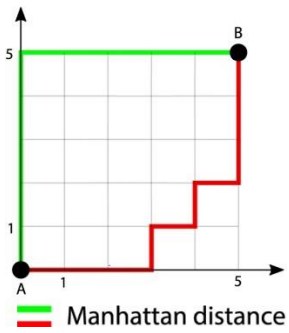
Note: Moves can be made only to an empty block represented by 1 while the filled block can be considered as 0. Sequence of steps should be in given order: Up, Left, Right and then Down.

Problem Description:

- Available actions are Left, Right, Up, Down.

  Left, Right, Up, down have the same meaning as discussed in the class

- The termination criterion is that the Maze is completed by the agent.
- You need to solve this Maze using Greedy Best First Search and A* algorithm. You can use Manhattan distance as the heuristic.



Manhattan distance is the tiling distance between two points.

Environment announces when this criterion is met and stops the execution.

- Agent knows the size of Maze (grid 20 x 20), the content of the cell they land in and the location of the landing cell (coordinates).
- The agent should be initially placed at (15,1) block as described in the Maze grid.
- The performance of an agent is calculated after the termination criteria is met. The performance measure of an agent is the number of steps used to complete the maze. Maze is completed when agent reaches (13,20) block.
- The environment is deterministic and fully observable.
- The perception is given by the environment and includes, cell coordinates and if the current piece in the cell is empty or blocked.