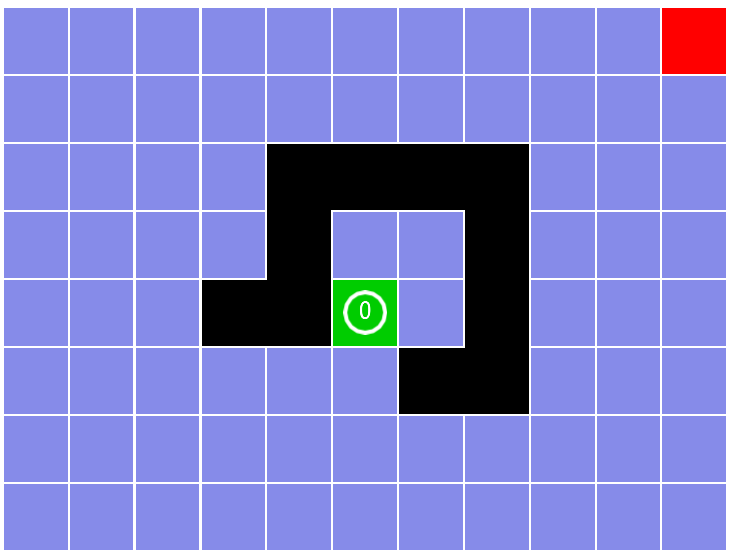
Artificial Intelligence: Search Algorithms

In this project, we aim to implement all five major search algorithms (BFS, UCS, DFS, Greedy Best-First search, and A^* search) we have learned in class.

1 Instructions

We consider a maze under a windy condition as shown in the following figure. We assume that the



wind comes from the south and the cost of one step for the agent is defined as follows: 1 for moving northward; 2 for moving eastward or westward; 3 for moving southward. We assume that the square labeled with 0 is the starting square and the top right is the goal square and all dark-shaded squares are obstacles.

For the best-first search algorithms (Greedy or A^*), we use a modified Manhattan distance used in class as the heuristic function $h(n)$ by considering the windy situation. For example, for the start node, the agent has to move at least 3 steps eastward and 3 steps northward in order to reach the goal. Therefore, we have $h(n) = 3\ 2 + 3\ 1 = 9$ at the start node.

We use a label we did in class to indicates the order of choosing the corresponding unlabeled square and adding it to the frontier. To break tier for unlabeled squares (expanding children nodes), use this order: first westward; then northward; then eastward; then southward. To break tier for labeled squares (picking one child node to expand), the smallest label is picked first.

Follow the same way as done in the class to show the search steps with labels inside circles for the following search algorithms: BFS, UCS, DFS, Greedy Best-First search, and A* search. The format of your output should look like this:

```
[]  []  []  []  []  []  []  []  25  27  28
[]  []  []  []  []  []  []  24  22  26  []
[]  []  []  []  ##  ##  ##  ##  20  23  []
[]  []  []  []  ##  01  []  ##  18  21  []
[]  []  []  ##  ##  00  02  ##  16  19  []
[]  []  []  []  04  03  ##  ##  13  17  []
[]  []  []  []  06  05  07  09  11  14  []
[]  []  []  []  []  08  10  12  15  []  []
```

2. Iterative deepening search (or iterative deepening depth-first search) is a general strategy, often used in combination with depth-first tree search, that finds the best depth limit. It does this by gradually increasing the limit – first 0, then 1, then 2, and so on – until a goal is found. This will occur when the depth limit reaches *d*, the depth of the shallowest goal node. Iterative deepening combines the benefits of depth-first and breadth-first search. Implement this algorithm for the above windy maze. Explain how it combines the benefits of depth-first and breadth-first search with the examples observed in your testing.