Assignment 3

1. Modify **AStarMaze** to compare the behaviors of the **Greedy Best-First** and **A\*** search algorithms.

**A\* Greedy**

|  |  |
| --- | --- |
| new\_g = current\_cell.g + 1  self.cells[new\_pos[0]][new\_pos[1]].f = new\_g + self.cells[new\_pos[0]][new\_pos[1]].h  self.cells[new\_pos[0]][new\_pos[1]].parent = current\_cell | new\_g = current\_cell.g |
| optimal, as the cost function g increases by 1 with each move. | not optimal. To make it behave as shown, I removed the “+1” increment from the code so that g(n)=0. Consequently, f(n)=h(n). |

1. Repeat the above experiment but this time:

* Use the Euclidean Distance heuristic.
* The agent is allowed to make diagonal moves (i.e., NE, NW, SE, SW) in addition to the usual N, S, E, and W moves.
* The moves are made randomly and not in any specific order.

**A\* Greedy**

|  |  |
| --- | --- |
| Euclidian Alg.  def heuristic(self, pos):  dist = (abs(pos[0] - self.goal\_pos[0]) \*\* 2 + abs(pos[1] - self.goal\_pos[1]) \*\* 2) \*\* 0.5  return round(dist)  BF Alg.  for dx, dy in [(0, 1), (0, -1), (1, 0), (-1, 0), (1, 1), (1, -1), (-1, 1), (-1, -1)]:  new\_pos = (current\_pos[0] + dx, current\_pos[1] + dy)  if 0 <= new\_pos[0] < self.rows and 0 <= new\_pos[1] < self.cols and not self.cells[new\_pos[0]][  new\_pos[1]].is\_wall:  #### The cost of moving to a new position is 1 unit  new\_g = current\_cell.g + 1 | Euclidian Alg.  def heuristic(self, pos):  dist = (abs(pos[0] - self.goal\_pos[0]) \*\* 2 + abs(pos[1] - self.goal\_pos[1]) \*\* 2) \*\* 0.5  return round(dist)  BF Alg.  #### Agent goes E, W, N, S, NE, NW, SE, SW whenever possible  for dx, dy in [(0, 1), (0, -1), (1, 0), (-1, 0), (1, 1), (1, -1), (-1, 1), (-1, -1)]:  new\_pos = (current\_pos[0] + dx, current\_pos[1] + dy)  if 0 <= new\_pos[0] < self.rows and 0 <= new\_pos[1] < self.cols and not self.cells[new\_pos[0]][  new\_pos[1]].is\_wall:  #### The cost of moving to a new position is 1 unit  new\_g = current\_cell.g |
| evaluates both g and h. When h values leveled off, it prioritized minimizing g. | considers only h, resulting in a less efficient and less direct path. |

1. The evaluation function in **AstarMaze** is defined as **f(n) = g(n) + h(n)**. A weighted version of the function can be defined as:

f(n) = . g(n) + . h(n) where

1. Explain how different values of alpha and beta affect the A\* algorithm's behavior. Tabulate your results:

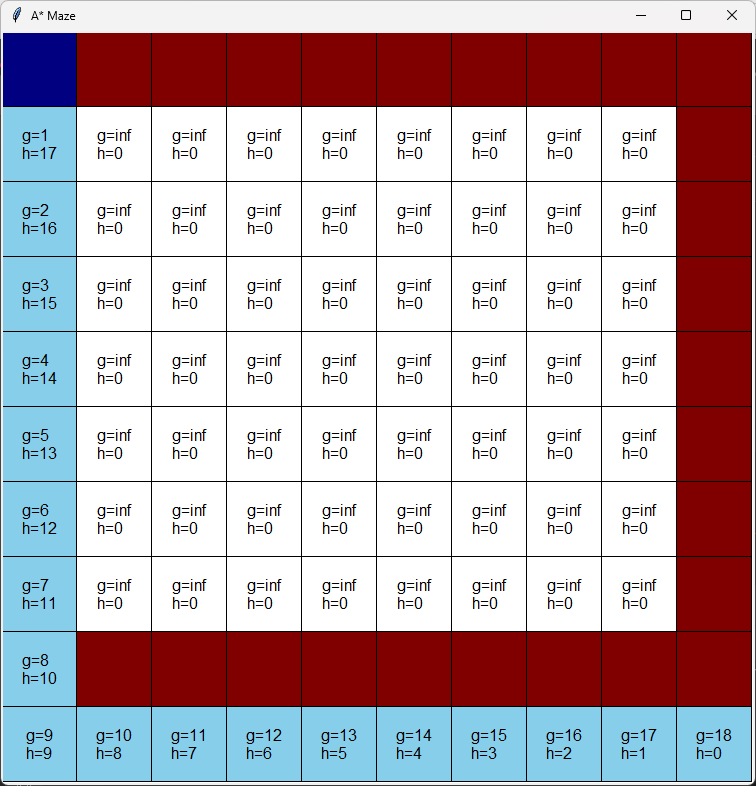
***a b Maze behavior***

| 1 | 1 |  | A\*  Optimal, shortest path  Balances cost and the heuristic |
| --- | --- | --- | --- |
| 0 | 1 |  | Greedy Best  First  Long, inefficient total path  Only follows heuristic |
| 0 | .5 |  | Heuristic-limited  Slightly greedy  Ignores g, influence of h reduced by half compared to greedy best first  more cautious than GBF |
| 1 | .5 |  | Cost-dominant  Prefers safe known routes over shortcuts  More cautious than A\*, but less goal oriented |

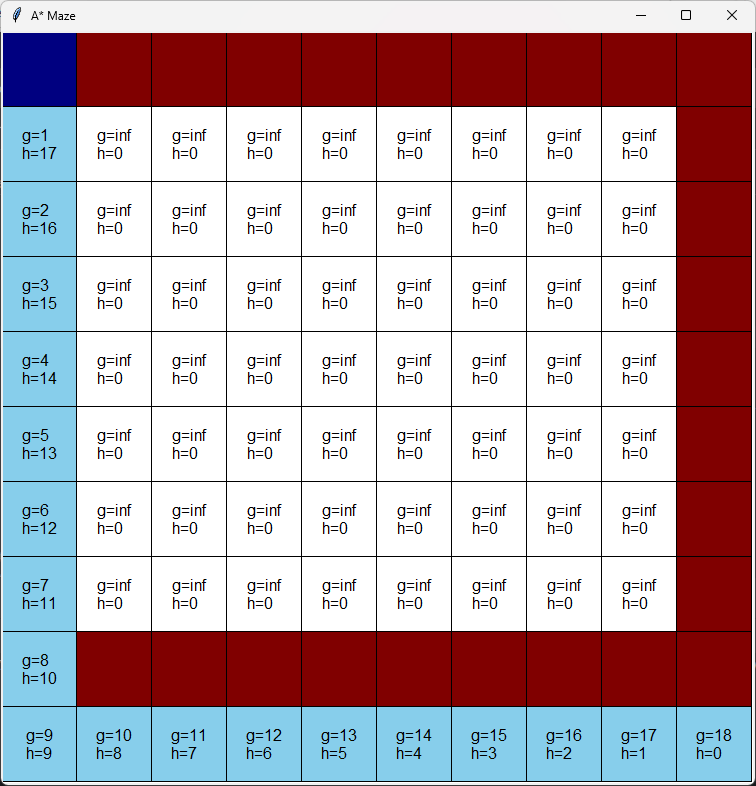
2. Beta can be considered the algorithm's bias towards states that are closer to goal. Run the algorithm for various values of the bias to determine what changes, if any, are observed in the optimum path. Include screenshots of the path for each specific value of b along with your explanation

(a=1 for all paths below)

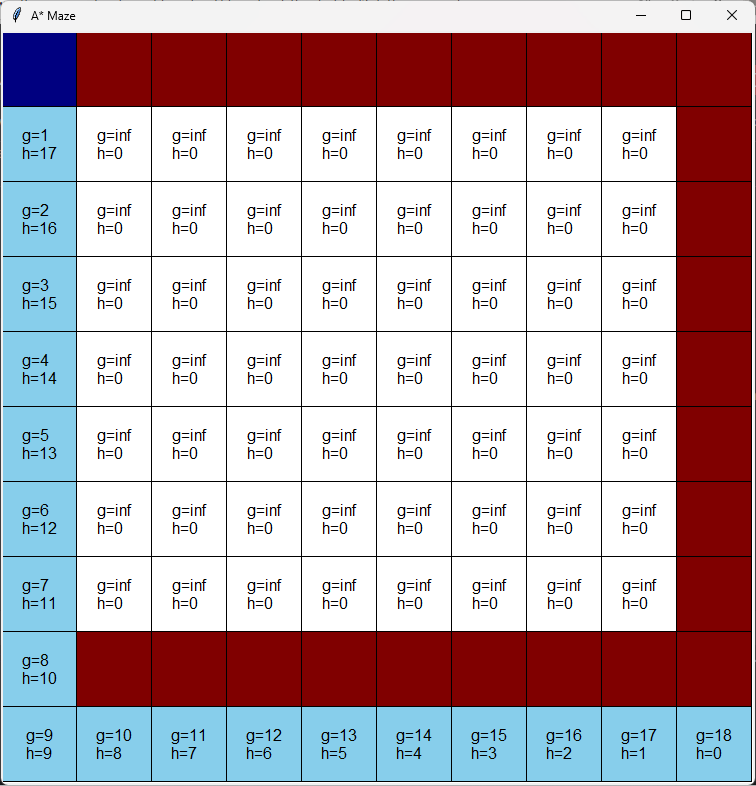
Smaller the b, the more the algorithm relies on cost and tends to be more cautious, avoiding risky heuristics. May expand many nodes before commiting

 b=0

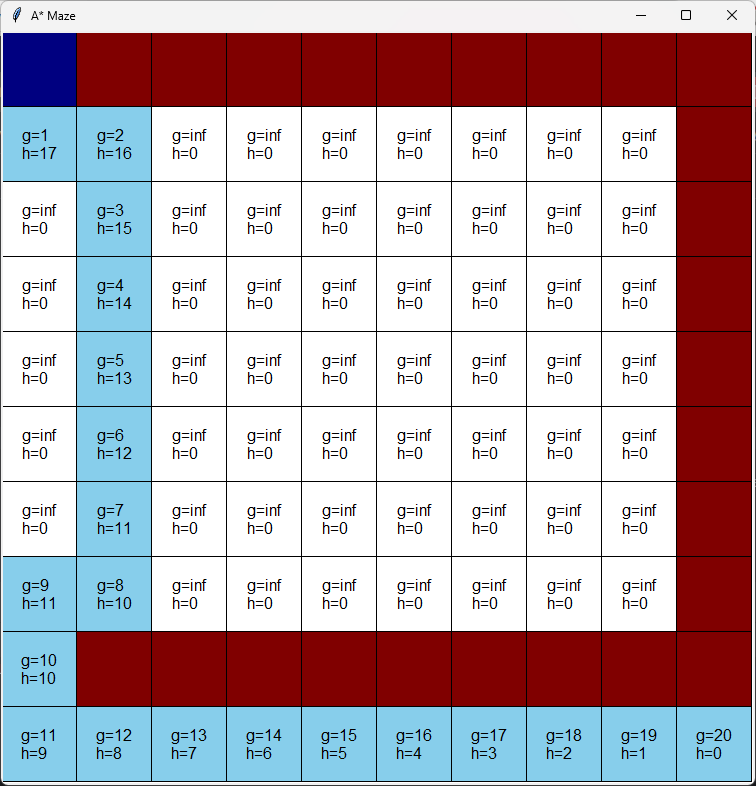
Moderate beta becomes somewhat exploratory, but prefers to be cautious. Still heuristic driven.

b=0.5

Balanced beta is optimal and balances efficiency and accuracy

b=1

Large beta becomes greedy and heads quickly towards goal regardless of heuristic. May take nonoptimal detours.

b=2