Sure! Let's go through the results you've summarized and analyze the values for each heuristic method (Manhattan, Chebyshev, and Euclidean).

**1. Goal Position: (1, 1)**

The goal position (1, 1) is the target location that your **A\* search algorithm** aims to reach. In all three heuristics, the goal is the same, which means you're solving the same problem, but with different heuristic methods guiding the search process.

**2. Path Length: 189 (same for all heuristics)**

* **Path Length** refers to the number of **steps** or **cells** that the agent has to travel to reach the goal from the starting point. In your case, regardless of the heuristic method (Manhattan, Chebyshev, or Euclidean), the number of steps from the start to the goal remains **189**.
* **Reason**: This is because the actual **shortest path** from the start to the goal is the same, no matter the heuristic used. The **path length** represents the number of grid cells the agent needs to traverse, which doesn't change across the different heuristics.

**3. Search Length:**

The **Search Length** represents the total number of **cells** that the **A\* search algorithm** explores during the search process before it finds the path to the goal. This includes all the cells considered by the algorithm, even if they aren't part of the final path.

* **Manhattan Heuristic**: 4265
* **Chebyshev Heuristic**: 5462
* **Euclidean Heuristic**: 5099

**Why the differences in search length?**

* **Manhattan**: The Manhattan heuristic calculates the cost of moving from the current cell to the goal by summing the absolute differences in the x and y coordinates. It often results in **more steps** being evaluated, as it doesn’t account for diagonal movement.
* **Chebyshev**: The Chebyshev heuristic calculates the maximum of the absolute differences in the x and y coordinates. This heuristic is **more flexible** than Manhattan because it allows for **diagonal movement** (i.e., it treats diagonal moves as having the same cost as orthogonal moves). This could result in a slightly **larger search space** because the algorithm may explore more cells to find the shortest path.
* **Euclidean**: The Euclidean heuristic calculates the straight-line distance (the **Euclidean distance**) between the current cell and the goal, which gives the **closest approximation** to the actual cost. Since it's based on the direct distance, the Euclidean heuristic tends to **explore fewer cells** than Manhattan and Chebyshev heuristics.

**4. Execution Time:**

The **Execution Time** represents the total amount of time the **A\* search algorithm** took to find the path from the start to the goal. This can be affected by the complexity of the heuristic function, the number of cells explored, and other factors.

* **Manhattan Heuristic**: 0.0089 seconds
* **Chebyshev Heuristic**: 0.0161 seconds
* **Euclidean Heuristic**: 0.0107 seconds

**Why the differences in execution time?**

* **Manhattan Heuristic**: This heuristic is computationally simpler because it only involves adding the absolute differences of the x and y coordinates. It is a **faster heuristic**, leading to the shortest execution time among the three.
* **Chebyshev Heuristic**: The Chebyshev heuristic is slightly **more complex** because it calculates the maximum of the absolute differences in both coordinates, and it also involves considering the possibility of diagonal moves. As a result, it leads to a **slightly longer execution time**.
* **Euclidean Heuristic**: The Euclidean heuristic requires calculating the **square root** of the sum of squares of differences between the coordinates. This is **computationally more expensive** than Manhattan but often results in fewer cells being explored, balancing the search length and execution time. This is why its execution time is slightly **longer than Manhattan** but **shorter than Chebyshev**.

**Summary:**

* **Goal Position**: In all cases, it's the same (1, 1)—so no differences here.
* **Path Length**: Remains constant at **189** steps in all cases because the optimal path length doesn't depend on the heuristic, but only on the maze layout.
* **Search Length**: The number of explored cells differs:
  + **Manhattan** explores **fewer cells** (4265).
  + **Chebyshev** explores the **most cells** (5462), possibly due to its consideration of diagonal moves.
  + **Euclidean** explores **fewer cells** than Chebyshev but **more than Manhattan** (5099).
* **Execution Time**:
  + **Manhattan** is the **fastest** (0.0089 seconds).
  + **Euclidean** comes next (0.0107 seconds).
  + **Chebyshev** takes the longest (0.0161 seconds) due to its more complex heuristic calculations.

**Conclusion:**

* **Manhattan** is the most **efficient** in terms of execution time and number of explored cells, but it may not always give the most optimal or efficient path in terms of actual distance traveled (because it doesn't consider diagonal movement).
* **Euclidean** and **Chebyshev** provide **better paths** in some scenarios (especially when diagonal movement is allowed) but require more computational resources (in terms of cells explored and execution time)

**Manhattan**

**A maze with blue and white lines

Description automatically generated**

**Chebyshev**

A maze with blue lines

Description automatically generated  
  
**Euclidean**

A maze with blue and white lines

Description automatically generated