LAB-7  
TASK 1  
A\* (A-Star) Algorithm

# Definition of A\* Algorithm:

A\* (A-Star) Algorithm is a popular and powerful pathfinding and graph traversal algorithm. It is widely used for finding the shortest path between a start node and a goal node, especially in fields like artificial intelligence for games, robotics, and route navigation systems. A\* efficiently finds the least-cost path by combining features of Dijkstra's Algorithm and Greedy Best-First Search.

# Usage of A\* Algorithm:

- Game Development: Finding the shortest path for characters or units.

- Robotics: Navigation of robots in an environment.

- GPS Systems: Finding the fastest route between two locations.

- Artificial Intelligence: Decision-making systems.

A\* uses two main costs:

- g(n): Actual cost from the start node to the current node.

- h(n): Estimated cost from the current node to the goal (heuristic).

- f(n) = g(n) + h(n): Total estimated cost.

The node with the lowest f(n) value is selected first.

# Explanation of the Code:

1. Imports:

- We import heapq to use a priority queue, which allows us to always pick the node with the smallest f value efficiently.

2. Node Class:

- Represents a position in the grid.  
 - Keeps track of cost values (g, h, f).  
 - \_\_eq\_\_ is used to compare nodes based on their positions.

3. Heuristic Function:

- Calculates the Manhattan distance between two points.  
 - It estimates how far the current node is from the goal.

4. A\* Algorithm Function:

- open\_list stores nodes to be evaluated.  
 - closed\_list stores nodes that have been evaluated.  
 - start\_node and goal\_node are initialized.

5. Main Loop:

- Continuously pops the node with the lowest f from open\_list.  
 - Checks if the current node is the goal.  
 - Otherwise, adds the current node's position to closed\_list.  
 - Generates neighbors (up, down, left, right).  
 - Checks if neighbors are walkable and not already evaluated.  
 - Calculates costs for the neighbor.  
 - Pushes valid neighbors into open\_list.

6. Path Reconstruction:

- Once the goal is found, it backtracks using the parent pointers.  
 - Reverses the path to give it from start to goal.

7. Example Usage:

- A simple 5x5 grid is created.  
 - Start and goal positions are defined.  
 - The path is found and printed.

## OUTPUT:

# Conclusion:

The A\* algorithm is an efficient and elegant method for solving shortest path problems. By balancing actual cost and estimated future cost, it can find the best route quickly even in complex environments.