**NAME**

**JAHNZAIB ZAFAR**

**SECTION**

**BS ARTFICIAL INTELLIGENCE– 3B**

**TASK (7)**

**A\*Algorithum**

## without Priority Queue

* This version of the A\* algorithm uses a simple set-based approach to find the shortest path
* between two nodes in a given graph. It maintains an open set (nodes to be visited) and a
* closed set (nodes already visited). The algorithm picks the node with the lowest total cost (g + h)
* and explores its neighbors until it reaches the goal node.  
    
  If the user provides an invalid start or goal node that doesn’t exist in the graph, the program displays a clear error message like:  
  "Invalid start or goal node! Please enter correct node names."
* Key Functions:  
  - get\_nodes(v): Returns all neighbors of a given node.  
  - h(n): Returns the heuristic (estimated distance to goal).  
  - astar(start\_node, end\_node): Finds the shortest path using A\* logic.

## Priority Queue

* In this version, we improved the A\* algorithm by using the Priority Queue (min-heap)   
  from Python’s built-in ‘heapq’ module. This makes the algorithm more efficient and faster   
  because it always selects the node with the smallest total cost automatically.
* Changes made:  
  - Imported 'heapq' module for priority queue handling.  
  - Replaced manual open\_set selection with heap-based selection.  
  - Logic and functions remain the same as in the first version.