المحاضرة السابعة
A* Algorithm

# A* Algorithm

```
def a_star(graph, start, goal, heuristic):
    open_set = [start]
    came_from = {}
     g_score = {node: float('inf') for node in graph}
    g_score[start] = 0
    f_score = {node: float('inf') for node in graph}
    f_score[start] = heuristic[start]
   while open_set:
       current = min(open_set, key=lambda node:
f_score[node])
```

```
        if current == goal:

            path = []
            while current in came_from:
                path.append(current)
                current = came_from[current]
            path.append(start)
            path.reverse()
            return path
          open_set.remove(current)
```

# A* Algorithm

```
  for neighbor, cost in graph[current].items():
        tentative_g = g_score[current] + cost
        if tentative_g < g_score[neighbor]:
            came_from[neighbor] = current
            g_score[neighbor] = tentative_g
            f_score[neighbor] = tentative_g +
heuristic[neighbor]
            if neighbor not in open_set:
                open_set.append(neighbor)
  return None
```

```
graph = {
    'A': {'B': 1, 'C': 4},
    'B': {'A': 1, 'C': 2, 'D': 5},
    'C': {'A': 4, 'B': 2, 'D': 1},
    'D': {'B': 5, 'C': 1, 'E': 3},
    'E': {'D': 3}
}

heuristic = {
    'A': 7,
    'B': 6,
    'C': 2,
    'D': 1,
    'E': 0
}

start_node = 'A'
goal_node = 'E'
```

```
# implementation

path = a_star(graph,
start_node, goal_node,
heuristic)
if path:
   Print (path)
else:
   print(“no path”)
```