DEPARTMENT OF COMPUTER & SOFTWARE ENGINEERING

COLLEGE OF E&ME, NUST, RAWALPINDI

AI & Decision Support Systems

Lab Mid

Student Name: Nawab Aarij Imam

Degree/ Syndicate: 43 CE - A

Task1:
Code:

```
def a_star(start, goal, graph, heuristics):
    open_list = [(start, 0, heuristics[start])]
    visited = set()
    cost_so_far = {start: 0}
    cameFrom = {start: None}

    while open_list:
        current_node, current_g, current_f = open_list.pop(0)
        if current_node == goal:
            return reconstruct_path(cameFrom, current_node)

        visited.add(current_node)
        for neighbor, edge_cost in graph[current_node].items():
            if neighbor in visited:
                continue
            tentative_cost = cost_so_far[current_node] + edge_cost
            if neighbor not in cost_so_far or tentative_cost < cost_so_far[neighbor]:
                cameFrom[neighbor] = current_node
                cost_so_far[neighbor] = tentative_cost
                new_cost = tentative_cost + heuristics[neighbor]
                open_list.append((neighbor, tentative_cost, new_cost))
                open_list = sorted(open_list, key=lambda x: x[2])

    return None



def reconstruct_path(cameFrom, current):
    total_path = [current]
    while current in cameFrom and cameFrom[current] is not None:
        current = cameFrom[current]
        total_path.append(current)
```

```
    total_path.reverse()
    return total_path



def main():
    graph = {
    'A': {'B': 5, 'C': 5},
    'B': {'A': 5, 'C': 4, 'D': 3},
    'C': {'A': 5, 'B': 4, 'D': 7, 'E': 7, 'H': 8},
    'D': {'B': 3, 'C': 7, 'H': 11,'K':16, 'L': 13, 'M': 14},
    'E': {'C': 7, 'F': 4, 'H': 5},
    'F': {'E': 4, 'G': 9},
    'G': {'F': 9, 'N': 12},
    'H': {'C': 8, 'D': 11, 'E': 5, 'I': 3},
    'I': {'H': 3, 'J': 4},
    'J': {'I': 4,'N': 3, 'P': 8},
    'K': {'N': 7, 'L': 5, 'N': 7, 'D': 16},
    'L': {'D': 13, 'K': 5, 'M': 9, 'O': 4},
    'M': {'D': 14, 'L': 9, 'O': 5},
    'N': {'J': 3, 'K': 7, 'P': 7,'G': 12},
    'O': {'L': 4, 'M': 5},
    'P': {'K': 4, 'N': 7, 'J': 8}
    }

    heuristics = {
        'A': 16,
        'B': 17,
        'C': 13,
        'D': 16,
        'E': 16,
        'F': 20,
        'G': 17,
        'H': 11,
        'I': 10,
        'J': 8,
```

```
    'K': 4,
    'L': 7,
    'M': 10,
    'N': 7,
    'O': 5,
    'P': 0
  }

  start = 'A'
  goal = 'P'

  path = a_star(start, goal, graph, heuristics)
  print(path)



if __name__ == "__main__":
  main()
```

Output:

```
❯ python task.py
['A', 'C', 'H', 'I', 'J', 'P']

~/L/Cl/OneDrive-N/NUST MATERIAL/7th Semester/AI/Lab/Mid
❯
```