COMSATS UNVERISTY ISLAMABAD

Artificial Intelligence

Lab 3

Submitted by:

Hasaan Ahmad SP22-BSE-017

Submitted to:

Sir Waqas Ali

Lab 1:

```
#SP22-BSE-017 HASAAN AHMAD

class Node:
    def __init__(self, name, neighbors=None):
        self.name = name
        self.neighbors = neighbors if neighbors else []
        self.visited = False


graph = {
    'Arad': Node('Arad', [('Zerind', 75), ('Sibiu', 140), ('Timisoara',
118)]),
    'Bucharest': Node('Bucharest', [('Giurgiu', 85), ('Pitesti', 211),
('Urziceni', 98)]),
    'Craiova': Node('Craiova', [('Drobeta', 120), ('Rimnicu Vilcea', 146),
('Pitesti', 138)]),
    'Drobeta': Node('Drobeta', [('Mehadia', 80)]),
    'Eforie': Node('Eforie'),
    'Fagaras': Node('Fagaras', [('Sibiu', 99), ('Bucharest', 211)]),
    'Giurgiu': Node('Giurgiu', [('Bucharest', 90)]),
    'Hirsova': Node('Hirsova', [('Urziceni', 98)]),
    'Iasi': Node('Iasi', [('Neamt', 87)]),
    'Lugoj': Node('Lugoj', [('Mehadia', 70)]),
    'Mehadia': Node('Mehadia', [('Lugoj', 75), ('Drobeta', 151)]),
    'Neamt': Node('Neamt', [('Iasi', 92)]),
    'Oradea': Node('Oradea', [('Zerind', 140)]),
    'Pitesti': Node('Pitesti', [('Rimnicu Vilcea', 97), ('Craiova', 138),
('Bucharest', 101)]),
    'Rimnicu Vilcea': Node('Rimnicu Vilcea', [('Sibiu', 80), ('Pitesti',
97), ('Craiova', 146)]),
    'Sibiu': Node('Sibiu', [('Fagaras', 99), ('Rimnicu Vilcea', 80),
('Arad', 140), ('Oradea', 151)]),
    'Timisoara': Node('Timisoara', [('Arad', 118)]),
    'Urziceni': Node('Urziceni', [('Hirsova', 86), ('Bucharest', 98),
('Vaslui', 142)]),
    'Vaslui': Node('Vaslui', [('Urziceni', 98), ('Iasi', 92)]),
    'Zerind': Node('Zerind', [('Oradea', 71), ('Arad', 75)])
    }

def BFS(graph, initialstate, goalstate):
```

```
    frontier = [initialstate]
    explored = []

    while frontier:
        currentNode = frontier.pop(0)
        explored.append(currentNode)

        if currentNode == goalstate:
            return actionSequence(graph, initialstate, goalstate)

        for child in graph[currentNode].neighbors:
            if child[0] not in frontier and child[0] not in explored:
                graph[child[0]].parent = currentNode
                frontier.append(child[0])

def actionSequence(graph, initialstate, goalstate):
    solution = [goalstate]
    currentParent = graph[goalstate].parent

    while currentParent != initialstate:
        solution.append(currentParent)
        currentParent = graph[currentParent].parent

    solution.append(initialstate)
    solution.reverse()
    return solution

initialstate = 'Arad'
goalstate = 'Bucharest'

solution = BFS(graph, initialstate, goalstate)
print(solution)
```

Output:

```
PS D:\Study Things\5th semester\AI\LAB> &
3/Activity1.py"
['Arad', 'Sibiu', 'Fagaras', 'Bucharest']
```

Lab 2:

```
#SP22-BSE-017 HASAAN AHMAD

import queue
maze = [
    ["#", "#", "#", "#", "#", "o", "#", "#"],
    ["#", "o", "o", "o", "o", "o", "o", "#"],
    ["#", "o", "#", "#", "#", "#", "o", "#"],
    ["#", "o", "#", "o", "#", "o", "o", "#"],
    ["#", "o", "#", "o", "#", "#", "o", "#"],
    ["#", "o", "#", "o", "#", "o", "o", "#"],
    ["#", "o", "o", "o", "o", "o", "o", "#"],
    ["#", "#", "#", "#", "#", "#", "#", "#"],
]

start = (3, 3)  # Starting position
end = (0, 5)    # Destination position

queue = queue.Queue()
queue.put([start])
visited = set()
visited.add(start)

directions = [(0, 1), (0, -1), (-1, 0), (1, 0)]

while not queue.empty():
    current_path = queue.get()
    current_position = current_path[-1]

    if current_position == end:
        print("Destination reached! Path:", current_path)
        break

    for direction in directions:
        new_position = (current_position[0] + direction[0],
current_position[1] + direction[1])

        if (
            0 <= new_position[0] < len(maze) and
            0 <= new_position[1] < len(maze[0]) and
            maze[new_position[0]][new_position[1]] != "#" and
```

```
            new_position not in visited
        ):
            new_path = list(current_path)
            new_path.append(new_position)
            queue.put(new_path)
            visited.add(new_position)
```

Output:

```
3/Activity2.py"
Destination reached! Path: [(3, 3), (4, 3), (5, 3), (6, 3), (6, 4), (6, 5), (6, 6), (5, 6), (4, 6), (3, 6), (2, 6), (1, 6), (1, 5), (0, 5)]
PS D:\Study Things\5th semester\AI\LAB>
```