A – Development Environment

I developed and tested the NQueens problem implementation in Python 3.11 on a Windows operating system. The code was written using Visual Studio Code as the integrated development environment (IDE), taking advantage of its features for Python development. The machine used for development is equipped with an Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz, providing a powerful processing capability with a base clock speed of 2.60 GHz and multiple cores.

B – Problem Formulation

State Specification: Explain how you represented the state in your NQueens problem. Mention the variables and their meanings.

Initial State: Describe how you set up the initial state for your NQueens problem. Did you generate it randomly or let the user input it?

Possible Actions: Outline the possible actions in your NQueens problem. For example, "Move Queen i to row j."

Transition Model: Explain how the transition model works in your problem. How does the state change when an action is applied?

Goal Test: Describe the conditions under which you consider a state as a goal state. In this case, when the number of attacking pairs is 0.

Path Cost: Explain how you defined the cost for each action. In this case, it seems each move has a cost of 1.

C – Results

Run your implementation with different parameters and inputs. For each run, include:

Algorithm Used: Specify which search algorithm you used (BFS, UCS, DFS, DLS, IDS, Greedy, A*).

Parameters: Include the values of parameters such as N, initial state, and depth limit for DLS.

Output Results: Display the output results, including the resulting state, path taken, total cost, viewer statistics, and the time taken for the algorithm.

```
**************************************************
breadth_first
Resulting path:
[(None, '2323'), (('Move Queen 4 to row 4', 4, 4), '2324'), (('Move Queen 4 to row 4', 4, 4), '2324'), (('Move Queen 4 to row 3', 4, 3), '2323'), (('Move Queen 3 to row 1', 3, 1), '2313'), (('Move
Resulting state: 2413
Total costs: 5
Viewer stats:
2413
Correct solution?: True
**************************************************
**************************************************
uniform_cost
Resulting path:
[(None, '2323'), (('Move Queen 4 to row 4', 4, 4), '2324'), (('Move Queen 4 to row 4', 4, 4), '2324'), (('Move Queen 4 to row 3', 4, 3), '2323'), (('Move Queen 3 to row 1', 3, 1), '2313'), (('Move
Resulting state: 2413
Total costs: 5
Viewer stats:
2413
Correct solution?: True
**************************************************
**************************************************
depth_first
Resulting path:
[(None, '2323'), (('Move Queen 4 to row 4', 4, 4), '2324'), (('Move Queen 4 to row 4', 4, 4), '2324'), (('Move Queen 4 to row 3', 4, 3), '2323'), (('Move Queen 3 to row 1', 3, 1), '2313'), (('Move
Resulting state: 2413
Total costs: 5
Viewer stats:
2413
Correct solution?: True
**************************************************
limited_depth_first
Resulting path:
[(None, '2323'), (('Move Queen 4 to row 4', 4, 4), '2324'), (('Move Queen 4 to row 4', 4, 4), '2324'), (('Move Queen 4 to row 3', 4, 3), '2323'), (('Move Queen 3 to row 1', 3, 1), '2313'), (('Move
Resulting state: 2413
Total costs: 5
Viewer stats:
2413
Correct solution?: True
**************************************************
**************************************************
iterative_limited_depth_first
Resulting path:
[(None, '2323'), (('Move Queen 4 to row 4', 4, 4), '2324'), (('Move Queen 4 to row 4', 4, 4), '2324'), (('Move Queen 4 to row 3', 4, 3), '2323'), (('Move Queen 3 to row 1', 3, 1), '2313'), (('Move
Resulting state: 2413
Total costs: 5
Viewer stats:
2413
Correct solution?: True
**************************************************
**************************************************
greedy
Resulting path:
[(None, '2323'), (('Move Queen 4 to row 4', 4, 4), '2324'), (('Move Queen 4 to row 4', 4, 4), '2324'), (('Move Queen 4 to row 3', 4, 3), '2323'), (('Move Queen 3 to row 1', 3, 1), '2313'), (('Move
Resulting state: 2413
Total costs: 5
Viewer stats:
2413
Correct solution?: True
**************************************************
**************************************************
astar
Resulting path:
[(None, '2323'), (('Move Queen 4 to row 4', 4, 4), '2324'), (('Move Queen 4 to row 4', 4, 4), '2324'), (('Move Queen 4 to row 3', 4, 3), '2323'), (('Move Queen 3 to row 1', 3, 1), '2313'), (('Move
Resulting state: 2413
Total costs: 5
Viewer stats:
2413
Correct solution?: True
**************************************************
**************************************************
```

D – Discussion

Discuss the results of different search algorithms:

Completeness: Discuss whether each algorithm was able to find a solution (if one exists).

Optimality: Evaluate the optimality of the solutions found by each algorithm.

Time and Space Complexity: Compare the time and space complexity of the algorithms. Discuss which algorithms performed well in terms of runtime and memory usage.

Graph Search vs. Tree Search: Compare the graph search and tree search versions of the algorithms.

Effect of Depth Limit in DLS: Discuss the impact of the depth limit in the depth-limited search algorithm.