Problem Statement:
Multi-agent N-queens problem: Create a chessboard kind of square area with N x N squares. Then create N turtles and place them randomly in the board. Now each turtle has to occupy a position such that there are no other turtles in the same row, column or diagonals containing this turtle. Clearly specify how you would program each turtle to take a decision. At each step, a turtle can move one position along the row, column or diagonals.

Approach:

1. The queens are placed randomly on the grid and it is arranged as 1 queen per column by ensure column command, since there can be more than one queen in one row.
2. Set priorities on every queen (the lower the id of the queen, the higher is her priority), the colors of the queens are assigned as blue and when a queen hasn't moved for 50 ticks then we make the position permanent assuming that this is her permanent position.
3. Let the other queens find their right position but if the rest of blue queens don't find the correct position for num^3. times, we reset the whole board and repeat this step until the output is achieved.

Input: N - Queens
Output: Possible solutions

Constraints: No two queens can place along same diagonal, along same row or along same column

Pseudocode for N-Queen Problem

```
->check-diag
  check if there is another queen on the diagonals

->check-row
  check if there is another queen on the same row

->ask turtles [if (ticks - moves > 50) [set color green]]
  if the turtle has not moved for 50 ticks, it has found its final position and is changed to
green tick

->ask turtles [if (moves > num * num * num) [set moves 0 set flg 1]]
  if the last few queens are unsuccessful in finding their appropriate positions on the chess
board set flg to 1

->if (flg = 1)
[
setup
ensure-unique-column
set flg 0
]
    //call setup to randomly reorganize the queens. This is done in line of backtracking..

->if (not any? turtles with [color = blue] ) [stop]
  if all the turtles have found their positions, stop the program
```