206. Discuss the importance of visualizing the solutions of the N-Queens Problem to understand the placement of queens better. Use a graphical representation to show how queens are placed on the board for different values of N. Explain how visual tools can help in debugging the algorithm and gaining insights into the problem's complexity. Provide examples of visual representations for N = 4, N = 5, and N = 8, showing different valid solutions.

a. Visualization for 4-Queens:
   Input: N = 4
   Output:
   Explanation: Each 'Q' represents a queen, and '.' represents an empty space.

PROGRAM:-

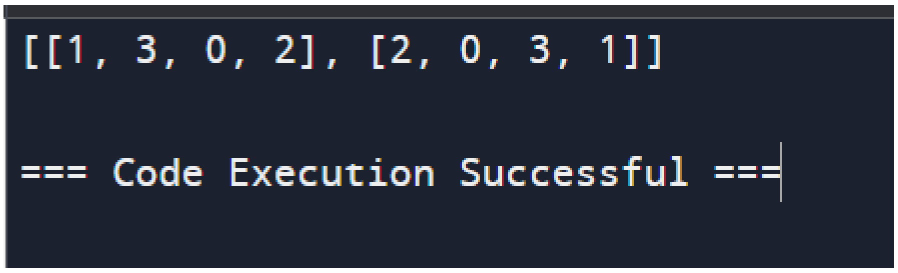
```
def solve_n_queens(N):
   def is_valid(board, row, col):
      for i in range(row):
         if board[i] == col or \
            board[i] - i == col - row or \
            board[i] + i == col + row:
             return False
      return True

   def solve(board, row):
      if row == N:
         solutions.append(board[:])
         return
      for col in range(N):
         if is_valid(board, row, col):
            board[row] = col
            solve(board, row + 1)
            board[row] = -1

   solutions = []
   solve([-1] * N, 0)
   return solutions

# Example usage
print(solve_n_queens(4))  # Find solutions for 4-Queens problem
```

OUTPUT:-



```
[[1, 3, 0, 2], [2, 0, 3, 1]]

=== Code Execution Successful ===
```

TIME COMPLEXITY:-O(n!)