**Assignment 6: Implement Basic Search Strategies – 8-Queens Problem**

**Problem Statement**

The goal of this assignment is to solve the 8-queens problem using basic search strategies, specifically through the application of backtracking. The challenge is to place 8 queens on an 8x8 chessboard in such a way that no two queens can attack each other.

**Objectives:**

* Learn to apply search strategies to constraint problems.
* Solve the 8-queens problem using a backtracking approach.

**Theory**

**What is the 8-Queens Problem?**  
The 8-queens problem is a classic constraint satisfaction problem where the objective is to place eight queens on a chessboard such that no two queens threaten each other. Queens can attack other pieces in the same row, column, or diagonal, so the placement must ensure that no two queens share these.

**Methodology**

1. **Start with an Empty Board:**
   * Begin with an empty 8x8 chessboard where no queens are placed.
2. **Place Queens One by One:**
   * Attempt to place a queen in each column of the current row, checking whether that position is safe, i.e., it does not threaten any previously placed queens.
3. **Use Backtracking:**
   * If placing a queen leads to a conflict, backtrack by removing the last placed queen and trying the next available position.
   * Repeat the process until all 8 queens are successfully placed or all possibilities are exhausted.
4. **Continue Until a Valid Configuration is Found:**
   * The algorithm explores all configurations of queen placements until one is found where no queens threaten each other.

**Working Principle / Algorithm**

The backtracking algorithm for solving the 8-queens problem:

1. **Initialize the Chessboard:**
   * Create an 8x8 array initialized to zero, representing an empty board.
2. **Define a Recursive Function to Place Queens:**
   * The function takes the current row as a parameter.
   * If all queens are placed (i.e., row equals 8), return true to indicate a solution is found.
3. **For Each Column in the Current Row:**
   * Check if placing a queen in the current column is safe by ensuring no other queen threatens it.
   * If valid, place the queen (mark the board with 1) and recursively attempt to place the next queen in the next row.
4. **Backtrack if Necessary:**
   * If placing a queen leads to no solution, remove the queen (reset the board position to 0) and try the next column.
5. **Return the Result:**
   * Once a valid configuration is found, print or return the board configuration.

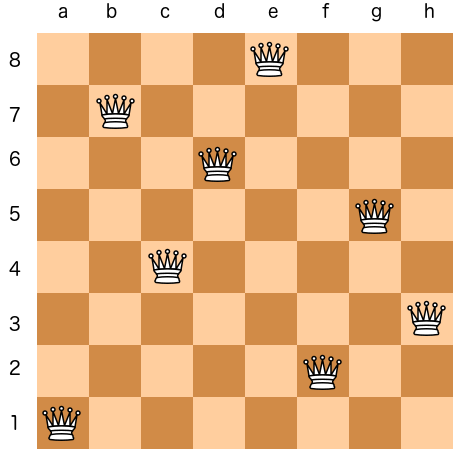
**Advantages**

* **Simplicity:**  
  The backtracking algorithm is simple and effective for constraint problems like the 8-queens problem.
* **Effectiveness:**  
  It efficiently explores possible configurations to find solutions, even though it may involve many backtracking steps.

**Disadvantages / Limitations**

* **Exponential Growth:**  
  The search space grows exponentially with the increase in board size (e.g., the 16-queens problem is significantly more complex).
* **Performance:**  
  The time complexity can be high if a large number of backtracking steps are required.

**Diagram**



**Conclusion**

Backtracking is an effective search strategy for solving constraint satisfaction problems like the 8-queens problem. It explores all possible configurations while ensuring that constraints are met, making it a valuable approach for solving similar problems. Despite its simplicity, it can become computationally expensive for larger problems due to the exponential growth of the search space.