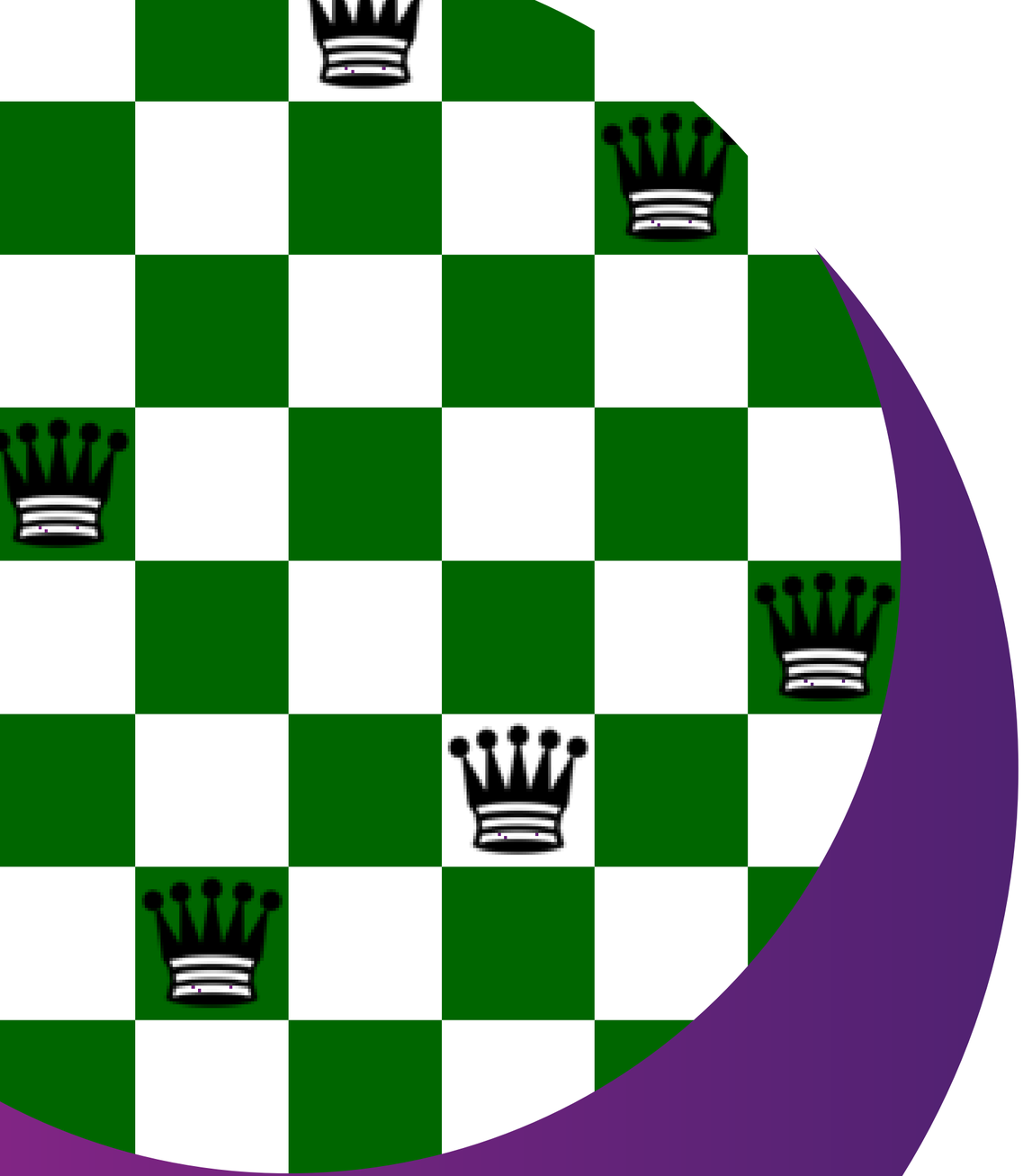
TORTAS AVL

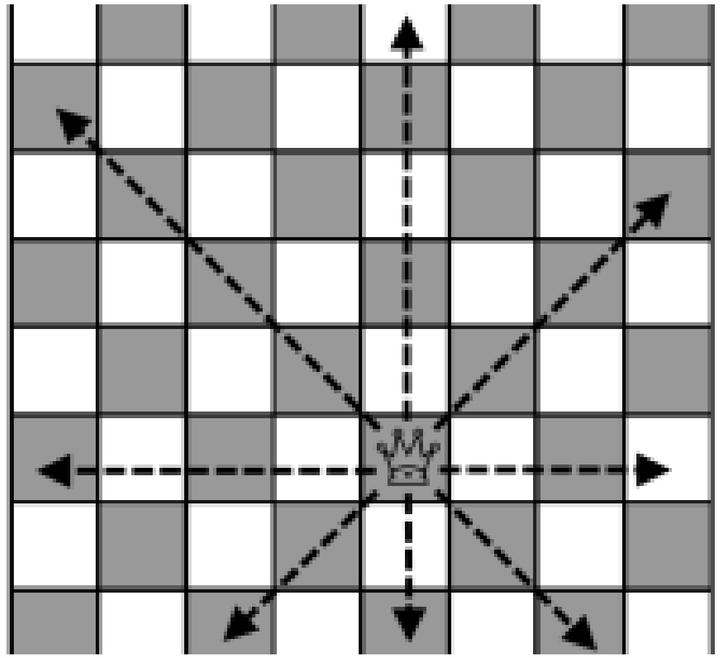
8 QUEENS PUZZLE

Carlos Eduardo López Cuevas - A01640751
Rodrigo Armando Reveles Picie - A01641220
Ricardo Sebastián González Rivas - A01646105



HOW TO PLAY?

The goal is to place eight queens on a standard 8×8 chessboard so that none threaten each other.



Because queens attack along rows, columns, and diagonals, this transcends simple gameplay mechanics, becoming a fundamental constraint-satisfaction problem (CSP).

A BRIEF HISTORY OF THE 8 QUEENS PUZZLE

The **Eight-queens** puzzle originated in **1848**, when the German chess player **Max Bezzel** proposed the original problem of placing eight queens on an 8×8 chessboard without attacking each other.

In **1850**, **Franz Nauck** published solutions and generalized the puzzle to the N-queens problem, allowing boards of any size.

IIn **1972**, **Edsger Dijkstra** helped popularize the N-queens puzzle by using it as an example of structured programming and backtracking. Since then, it has become a classic teaching tool for recursion, algorithm analysis, and data structures in computer science.

RULES

Unique Square: Each of the eight queens must occupy its own unique square on the board

No Attacks: No two queens may share the same row, column, or diagonal line.

Exact Count: The solution is only valid if exactly eight queens are placed simultaneously.

Static Challenge: There are no "turns" or opponent; the challenge lies entirely in constructing a valid static configuration.

4 x 4 Chess Board

Finding a valid arrangement is computationally intensive due to the exponential growth of possibilities.

4.4 Billion Placements

The number of ways to place 8 queens on 64 squares is astronomical, making brute force impractical.

Algorithmic Necessity: Strategies like Backtracking, Heuristics (Hill Climbing), and Genetic Algorithms are essential to navigate the search space efficiently.

DATA STRUCTURES FOR EFFICIENCY

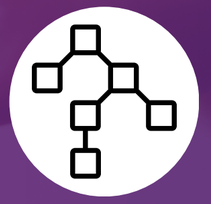
Vectors and arrays

Used to track occupied lines.

col[c], diag1[r+c] and **diag2[r-c+N-1]** allow for O(1) conflict checks, avoiding full board scans.

Stacks

Essential for simulating recursion iteratively. The stack stores the state of the board at each step, allowing the algorithm to "backtrack" to a previous valid state.



Search Trees

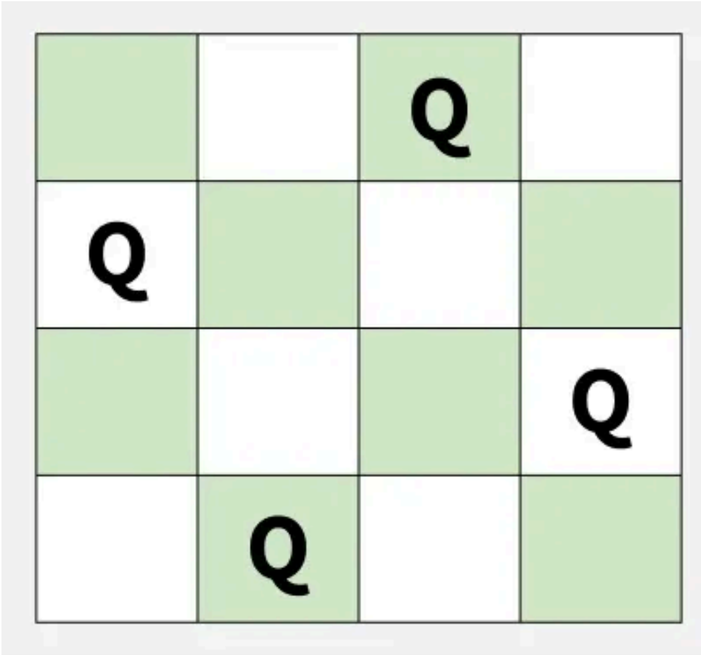
While not always explicitly stored, the solution space forms an N-ary tree. Algorithms traverse this tree, pruning branches that violate constraints.

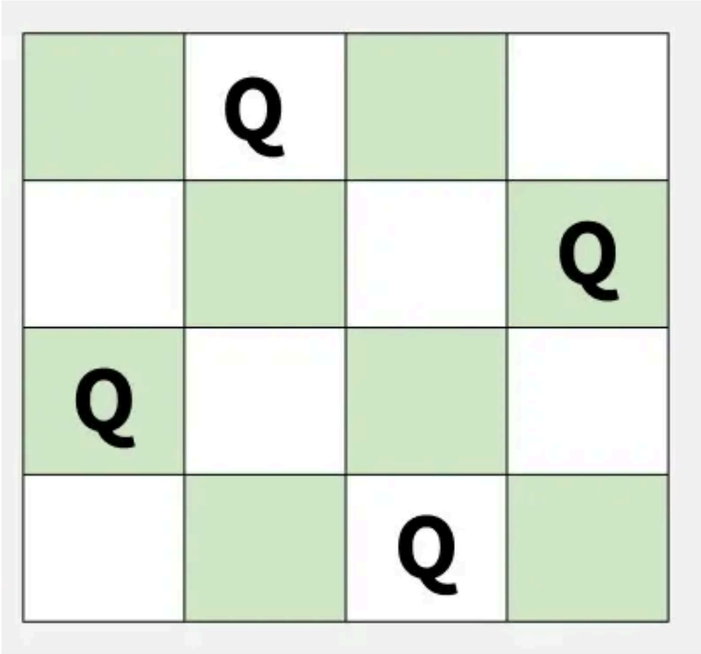
THE BACKTRACKING ALGORITHM

Backtracking is the standard approach. It places a queen, validates the move, and recursively attempts to place a queen in the next row.

- Step 1: Place queen in current column.
- Step 2: Check safety using O(1) arrays.
- Step 3: If safe, move to next row (Recurse).
- Step 4: If no position is safe, backtrack to previous row and try next column.

This effectively prunes vast sections of the search tree that contain no solutions.





Performance Integration

Different optimization techniques can be used to solve the N-Queens problem, and the choice of data structures has a huge impact on how fast a solution can be found.

1. NAIVE BOARD SCAN

This approach checks the entire board every time a queen is placed to see if there are conflicts. It's the slowest method because it repeatedly scans many squares.

2. ARRAY CONSTRAINTS

This approach uses arrays such as col, diag1, and diag2 to quickly determine whether a position is safe. Each check is done in constant time, which makes the algorithm much faster.

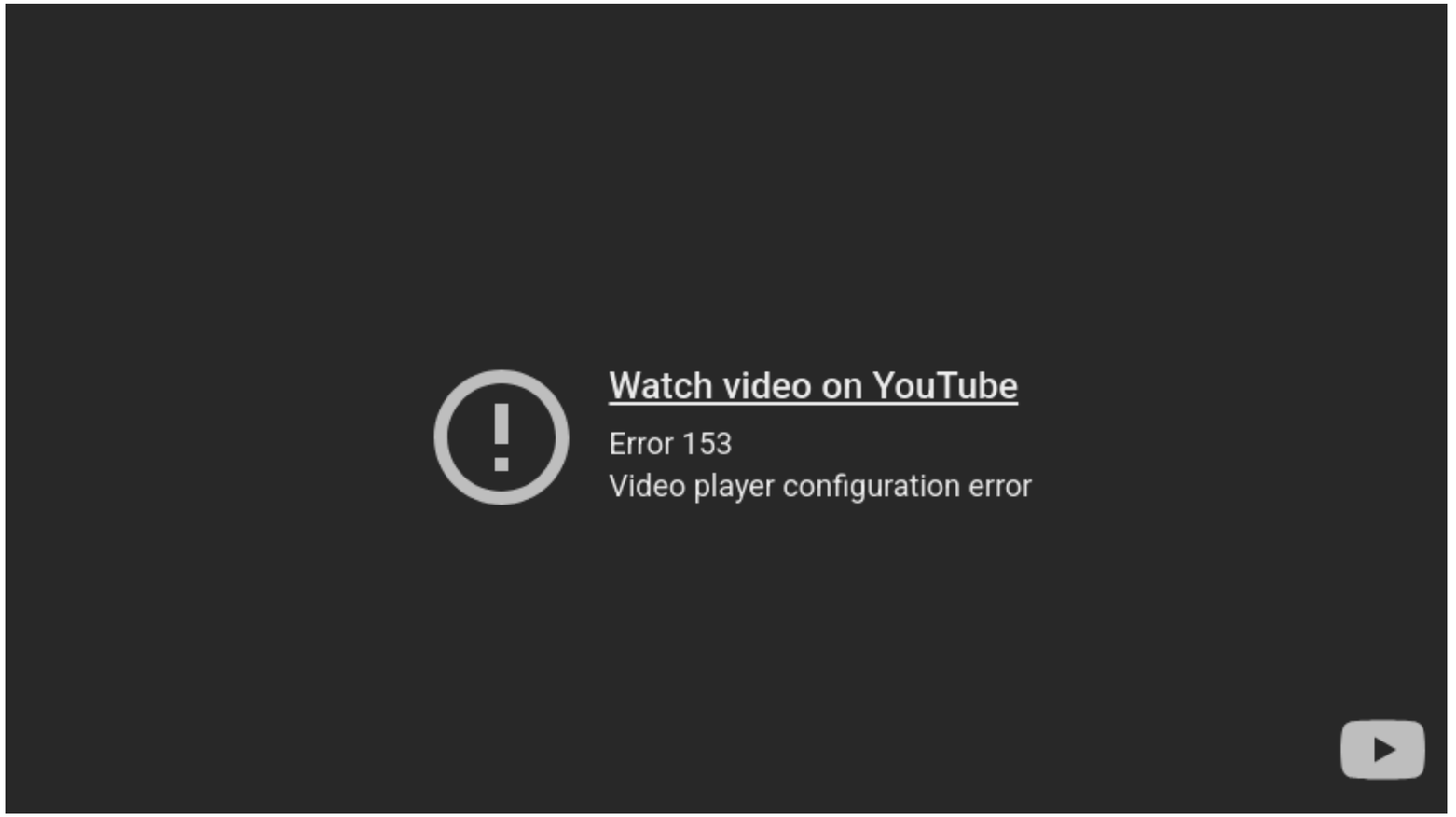
3. BITWISE OPTIMIZATION

This is the fastest method. Columns and diagonals are represented using integers, and conflicts are checked using bitwise operations. Since these operations work directly at the hardware level, they provide the highest performance. BUT we are not using this technique in our implementation due to its complexity.

USER EXPERIENCE

Our program uses a simple text-based interface. After each move, the board is printed clearly so the user can track queen placements. The system gives instant feedback on invalid positions, ranges, and conflicts, making the interaction straightforward and easy to follow within a console environment.

VIDEO


Watch video on YouTube
Error 153
Video player configuration error

CONCLUSIONS

Carlos:
Through this project, I realized that the Eight Queens puzzle is far more complex than it looks. Researching it showed me how much thought and mathematics go into solving it efficiently. This experience helped me understand the importance of choosing the right data structures and algorithms, and how techniques like backtracking can greatly improve performance and problem-solving logic.

Ricardo:
Through the Eight Queens Problem I learned that there are multipe approaches to a data structure problem. Certain structures can perform more efficiently, depending on how they're used. Implementing this classic chess problem also taught how data structures and algorithms go hand in hand to solve complex problems. A good algorithm, combined with an appropriate structure can easily elevate the performance of a program.

Rodrigo:
During this practice, I learned that even a puzzle like the 8-queens can be approached algorithmically, and I saw the versatility of data structures and how they enable efficient problem-solving with significant impact. More importantly, I realized how choosing the right representation—whether using full-board scans or constraint-tracking arrays—completely changes performance. This exercise made clear that optimization isn't optional; it's a mindset that separates a brute-force solution from a scalable one.

REFERENCES

S, Santhosh G, Joshi, P., Barui, A., & Panigrahi, Prasanta K. (2023). A Quantum Approach to solve N-Queens Problem. ArXiv.org. https://arxiv.org/abs/2312.16312

Aprende Estrategia del Rompecabezas de las Ocho Reinas - ... (2025). Eightqueenspuzzle.com. https://eightqueenspuzzle.com/learn/history?lang=es

GeeksforGeeks. (2024, January 15). Backtracking Algorithm. GeeksforGeeks. https://www.geeksforgeeks.org/dsa/backtracking-algorithms/

Bruce Boatner. (2025). Youtube.com. https://www.youtube.com/watch?v=Vr6SvKpYdxs