Daily Coding Problem #38

Problem

This problem was asked by Microsoft.

You have an N by N board. Write a function that, given N, returns the number of possible arrangements of the board where N queens can be placed on the board without threatening each other, i.e. no two queens share the same row, column, or diagonal.

Solution

If we were to attempt to solve this problem using brute force, we would quickly find out that it would be prohibitively expensive. Consider a typical 8 by 8 board: we have 64 spots to place 8 queens, so that's 64 choose 8 possible placements. In general, that's factorial in runtime!

This problem is ripe for solving with backtracking. In backtracking, we can visualize the search space like a tree, and we would explore it depth-first. Each node would be a possible configuration. If the configuration contains eight queens and is valid, then we're done and we can add it to our count. Otherwise, we can try to place another queen somewhere on the board and search from there. If we encounter an invalid board, then we can just prune the entire subtree from our search -- there's no point in exploring a board that we know won't work.

Notice we can pare down the search space by ensuring we only place queens in distinct rows, since we know that two queens can never occupy the same row.

Now we can just represent the board as a one-dimensional array of max size N, where each value represents which column the queen is on. For example, one solution for N = 4 would just be [1, 3, 0, 2].

```
def n_queens(n, board=[]):
    if n == len(board):
        return 1

    count = 0
    for col in range(n):
        board.append(col)
        if is_valid(board):
            count += n_queens(n, board)
        board.pop()
    return count

def is_valid(board):
    current_queen_row, current_queen_col = len(board) - 1, board[-1]
    # Iterate over all already-placed queens and check if any of them can attack
    # each other.
    for row, col in enumerate(board[:-1]):
        diff = abs(current_queen_col - col)
        if diff == 0 or diff == current_queen_row - row:
            return False
    return True
```

If you're interested in optimizing this problem even further, check out this paper that uses constant space by representing all columns and diagonals simply with integers! However, this depends on n being smaller than the number of bits in your integer.