The Eight Queens Puzzle asks you to discover all possible ways that eight queens could be placed on a chessboard so that no queen could “capture” any other on the next move.

In class, we discussed a “dumb” algorithm for solving this problem. We saw that we could represent each board configuration where there is one queen in each column by an 8 digit base 8 integer. The algorithm goes as follows:

0. Create 8 nested for loops, each with an index going from 0 – 7.
1. Let q be a vector whose components are the values of the indices.
2. check for queens occupying the same “row” by seeing if there are any duplicate digits in q, and
3. check to see if any of the queens represented by the configuration in q lie on the same diagonal.

If the answer to both questions 2 and 3 is “no”, we have a good configuration, so print it.

If the answer to either question 2 or question 3 is “yes”, then this number does not represent a valid configuration, so we skip it.

4. If the for loops have not yet completed, they will generate a new set of index values, then go to 1.
5. Otherwise, you’re done.

I coded it using two “helper functions”: 

![One solution to the eight queens puzzle](image)
def duplicates(q):
    if list q contains duplicate digits:
        return True
    else:
        return False

def diagonal_threat(q):
    if there is a diagonal “threat” (2 queens occupying the same diagonal):
        return True
    else:
        return False

Assignment:

Write a program to generate and print all the solutions to the Eight Queens Puzzle. Print each solution on its own line. Here are the first 35 (out of 92) solutions that my program printed.