N-Queens

SOFE 2715U: Data Structures
Home Activity

Alden O'Cain (100558599)
Reyhan Kogukoglu (100779413)

Ontario Tech University
Faculty of Engineering and Applied Science
Shahryar Rahnamayan, PhD, P.Eng

March 25th, 2022

Alden O'Cain - 100558599
Reyhan Kogukoglu - 100779413

Table of Contents

Background:

First posed in the 19th century, the n-queens puzzle is the problem of placing n queens on an n x n chessboard such that no two queens attack each other; thus, a solution requires that no two queens share the same row, column, or diagonal. For 8-Queens, there are 92 valid solutions.

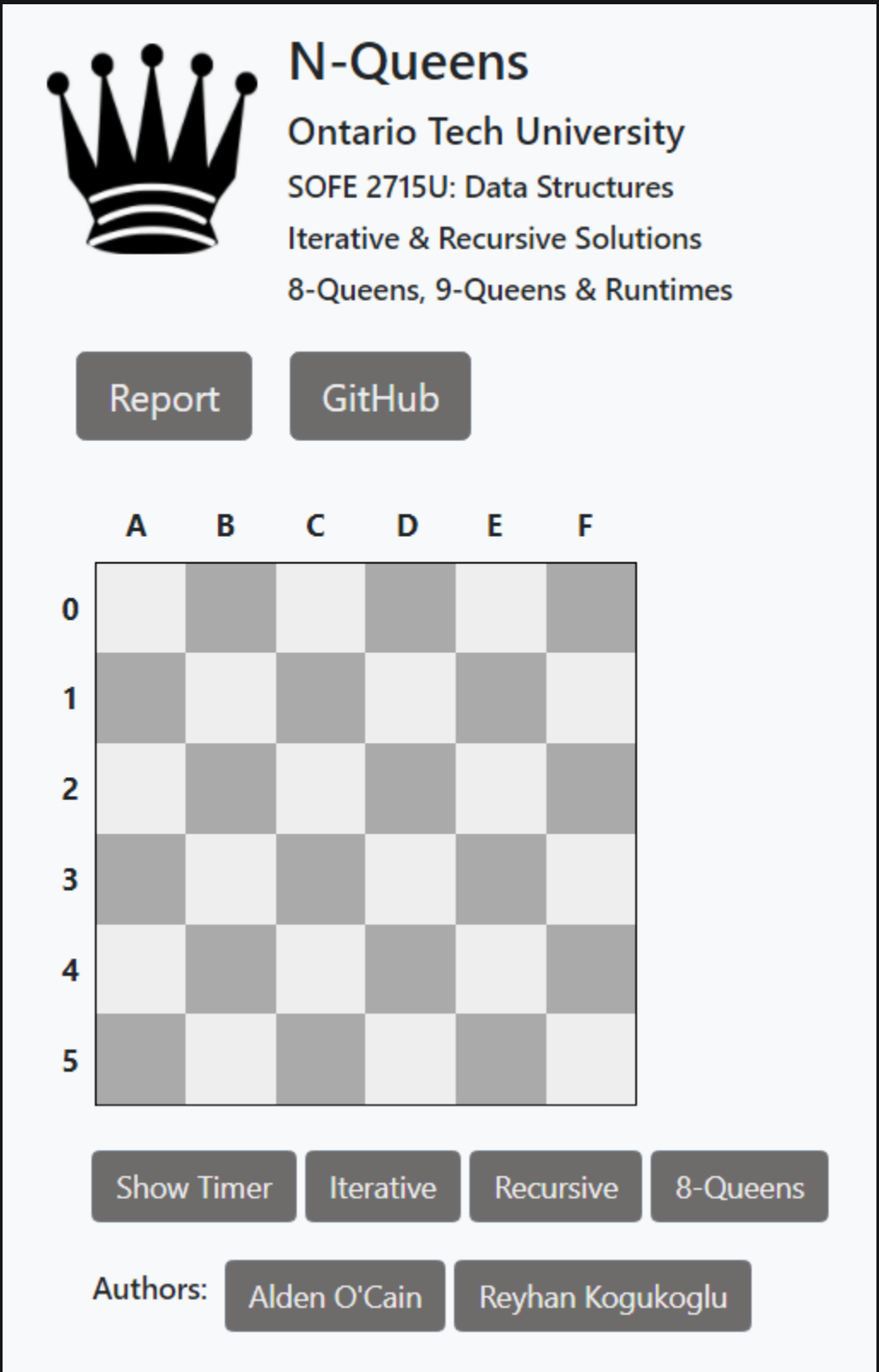
Problem Statement:

Solve N-Queen problem using:

1) an iterative method, and
2) recursive method for N=8 and N=9.

Alden O'Cain - 100558599
Reyhan Kogukoglu - 100779413

Webpage / GUI:



Alden O'Cain - 100558599
Reyhan Kogukoglu - 100779413

Iterative Pseudocode:

```
function iterative_solution(){
    // start the timer
    // get the board size based off what is displaying on the screen
    // initialize a board with n queens along the main diagonal

    // find the permutations of the column indices
    // permute the boards based off the found permutations that represent the
columns

    // check each possible board for validity (see pseudocode in check_board)
    // if valid save the board into a multi dimensional array of solution boards
    // if this was the first valid board stop the timer to save the time for
first solution

    // update the HTML for the last attempted board (or first solution)
    // calculate the runtime
    // unlock the timer module in the user interface
    // report the findings (below) to the browser console
    // (# of solutions), method (iterative), board_size and total runtime
}
```

Alden O'Cain - 100558599
Reyhan Kogukoglu - 100779413

Recursive Pseudocode:

```
function recursive_solution(){
    // create board of size board_size

    for (k = 0; k < board_size; k++){
      boardRow = []
      for (l = 0; l < board_size; l++){
      // Place 0's in each row cell
      }
          // Push row onto board
          }
      nQueen(board, 0)


    function nQueen(board, row){
        if (row is equal to board size){
            // Push board into valid solutions array
        Return
    }

    for (column = 0; column < board size; column++){
            if (isSafe(board, row, column){
            // Set Board[row][column] to 1;
            // Make a recursive call nQueen(board, row + 1);
            // set board[row][column] to 0;
            }
    }

    function isSafe(board, row, column){
        // check if the columns have any threats
        // return false if there are threats

        // check right diagonal for threats
        // return false if there are threats

        // check left diagonal for threats
        // return false if there are threats

        // return true if safe
    }
}
```

Alden O'Cain - 100558599
Reyhan Kogukoglu - 100779413

Check Board Pseudocode:

```
function checkBoard(){
    // checks the full board based off rows, columns and diagonals

    // rowValidity = checkRows(board);
    // colValidity = checkCols(board);
    // ldiagValidity = checkLeftDiags(board);
    // rdiagValidity = checkRightDiags(board);

    // if all are valid, then board is valid
        // return true
    // if not
        // return false
}

function checkBoardDiags(){
    // checks the full board (diagonals only)

    // ldiagValidity = checkLeftDiags(board);
    // rdiagValidity = checkRightDiags(board);

    // if both are valid, then board is valid
    // return true
    // if not
    // return false
}

function checkRows(){
    // board_size = board.length;
    // for y 0 - board_size
        // numQueens in row = 0
        // for x 0 - board_size
            // if cell is 1
                // numQueens += 1
            // if numQueens > 1
                // return false
    // else return true
}

function checkCols(){
    // board_size = board.length;
    // for x 0 - board_size
        // numQueens in column = 0
        // for y 0 - board_size
            // if cell is 1
                // numQueens += 1
            // if numQueens > 1
                // return false
    // else return true
}
```

Alden O'Cain - 100558599
Reyhan Kogukoglu - 100779413

Iterative Implementation (TypeScript):

```
function iterative_solution(){
    set_button_green();
    startTimer(); // saves the start time of the program
    let first_solution_found:boolean = false;
    let valid_solutions:number[][][]=[];
    // const board_size = checkHTMLBoardSize()!;

    const board_size = 6;

    // initialize the board with a main diagonal of queens
    let board:number[][] = [];
    for(let k=0; k < board_size; k++){
        let boardRow:number[] = [];
        for(let l=0; l < board_size; l++){
            boardRow.push(0);
        }
        boardRow[k] = 1;
        board.push(boardRow);
    }
    // updateBoardHTML(board); // seems to get missed even with a delay

    // setup - calculates permutations of set of N size
    const permutations = (ourPermutationsList: any[]) : any => {
        if (ourPermutationsList.length <= 2) return ourPermutationsList.length === 2 ?
[ourPermutationsList, [ourPermutationsList[1], ourPermutationsList[0]]] : ourPermutationsList;
        return ourPermutationsList.reduce(
            (acc: string | any[], item: any, i: number) =>
                acc.concat(
                    permutations([...ourPermutationsList.slice(0, i), ...ourPermutationsList.s
1)]).map((val: any) => [
                        item,
                        ...val,
                    ])
                ),
            []
        );
    };
    // saves permutations representative of columns forming main diagonal (containing queens)
    var rowPermutationList:number[][] = permutations(Array.from(Array(board_size).keys()));
    // console.log(rowPermutationList.length) // should be 362 880 for 9, 40 320 for 8
    // rowPermutationList.forEach(element => {
    //     console.log(element);
    // });

    // setup cont. - performing permutedBoard on diagonal sub arrays
    var allPossibleBoards:number[][][] = [];
    var permutedBoard:number[][];
    rowPermutationList.forEach(permutation =>{
        // loop through all permutations

        // create a new empty board
        permutedBoard = [];
        for(let k=0; k < board_size; k++){
            let permutedBoardRow:number[] = [];
            for(let l=0; l < board_size; l++){
                permutedBoardRow.push(0);
            }
            permutedBoard.push(permutedBoardRow);
        }
```

```
        // change column positions of diagonal entries
        // based off element values the permutation
        let permutationCounter:number = 0;
        permutation.forEach(element =>{
            permutedBoard[permutationCounter] = board[element];
            permutationCounter += 1;
        });
        console.log(permutedBoard);
        allPossibleBoards.push(permutedBoard);

    });

    // action: check all possible boards for validity
    allPossibleBoards.forEach(permutedBoard =>{
        board = permutedBoard;
        let size_before:number = valid_solutions.length;
        valid_solutions = checkBoardDiags(board, valid_solutions);

        // console.log(permutedBoard);

        // if this is the first solution, save the time and calculate difference
        if(valid_solutions.length != size_before && !first_solution_found){
            first_solution_found = true;
            stopTimerFirst();
        }
    });

    updateBoardHTML(board); // updates the HTML chess board on the screen
    let runtime:number = stopTimer(); // saves the total times and returns value
    unlockTimerButton("Iterative"); // unlocks the timer and sets label
    // unlockSolutions(); // unlocks the solutions menu
    console.log(valid_solutions.length + " solutions found iteratively for " + board_size + "-Queens in "
+ runtime + " milliseconds!");
}

function set_button_green(){
    // unset recursive button from being green if recursive was run first
    let recursiveBtn = document.getElementById("recursive-btn")!;
    let recursiveBtnStyles = recursiveBtn.getAttribute("class");
    if (String(recursiveBtnStyles).includes("success")){
        recursiveBtn.setAttribute("class", "mt-4 btn btn-secondary chess-btn");
    }
    // sets the button clicked to green
    let timerBtn = document.getElementById("iterative-btn")!;
    timerBtn.setAttribute("class", "mt-4 btn btn-success");
}
```

Alden O'Cain - 100558599
Reyhan Kogukoglu - 100779413

Recursive Implementation (TypeScript):

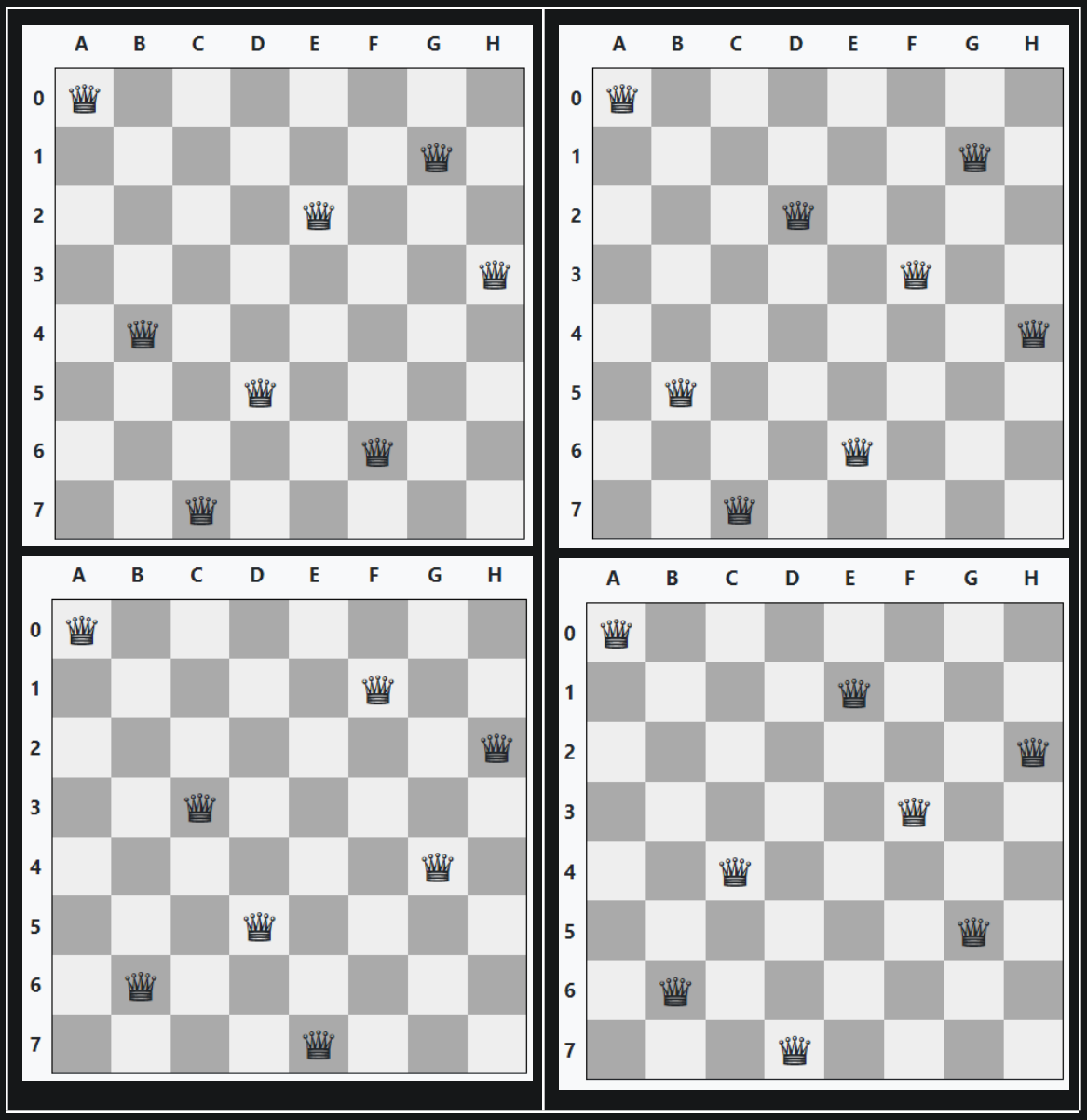
```
let valid_solutions:number[][][]=[];

function recursive_solution(){
    // unset iterative button from being green if recursive was run first
    let recursiveBtn = document.getElementById("iterative-btn")!;
    let recursiveBtnStyles = recursiveBtn.getAttribute("class");
    if (String(recursiveBtnStyles).includes("success")){
        recursiveBtn.setAttribute("class", "mt-4 btn btn-secondary chess-btn");
    }
    // sets the button clicked to green
    let timerBtn = document.getElementById("recursive-btn")!;
    timerBtn.setAttribute("class", "mt-4 btn btn-success");

    // so basically for loop checks each column of the row
    // and the Queen(b, r+1) iterate to next row anad then
    // it use recursive to chahck the column of that row
    const board_size = checkHTMLBoardSize()!;
    // initialize the board empty
    let board:number[][] = [];
    for(let k=0; k < board_size; k++){
        let boardRow:number[] = [];
        for(let l=0; l < board_size; l++){
            boardRow.push(0);
        }
        board.push(boardRow);
    }
    // recursive_nqueen(board, 0, board_size, valid_solutions);

    // solveNQUtil(board,0);

    nQueen(board, 0);

    let runtime:number = 0;
    console.log(valid_solutions.length + " solutions found recursively for " + board_size + "-Queens in "
+ runtime + " milliseconds!");
}

function nQueen(board:number[][], row:number){
    const board_size = board.length;
    if(row == board_size){
        valid_solutions.push(board);
        return
    }

    for(let column=0; column < board_size; column++){
        // @ts-ignore
        if(isSafeRec(board, row, column)){
            board[row][column] = 1;
            nQueen(board, row + 1);
            board[row][column] = 0;
        }
    }
}
```

```
function isSafeRec(board:number[][], row:number, column:number){
    const board_size = board.length;
    for(let i=0; i < row; i++){
        if(board[i][column] == 1){
            return false;
        }
    }

    let i:number = row;
    let j:number = column;
    while(i >= 0 && j >= 0){
        if(board[i][j] == 1){
            return false;
        }
        i -= 1;
        j -= 1;
    }

    i = row;
    j = column;
    while(i >= 0 && j < board_size){
        if(board[i][j] == 1){
            return false;
        }
        i -= 1;
        j -= 1;
    }
    return true;
}
```

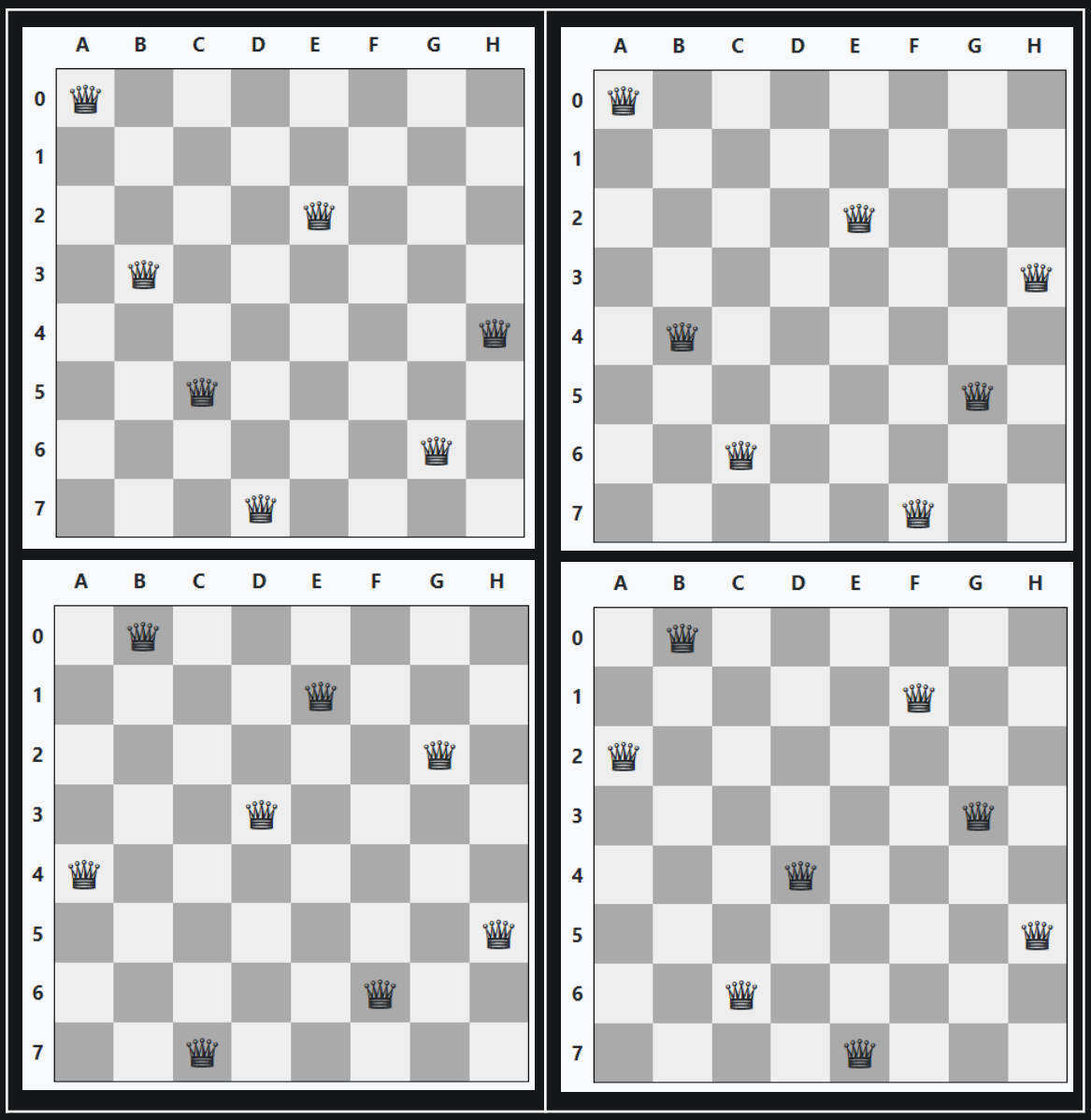
Alden O'Cain - 100558599
Reyhan Kogukoglu - 100779413

All Solutions:

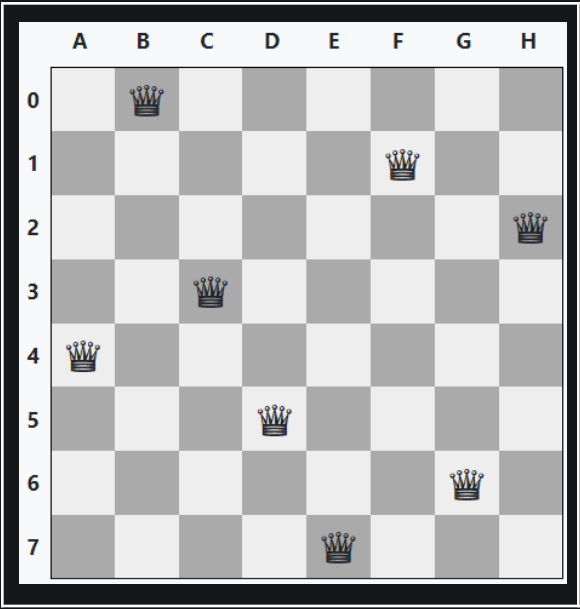
8-Queens Solutions:

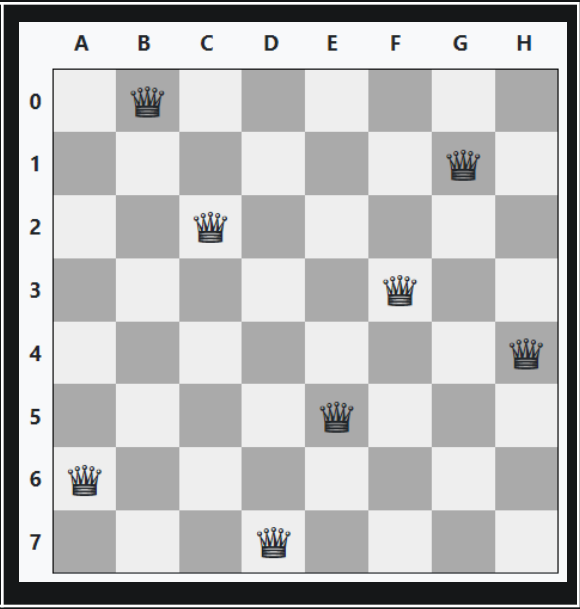


Alden O'Cain - 100558599
Reyhan Kogukoglu - 100779413



Alden O’Cain - 100558599
Reyhan Kogukoglu - 100779413



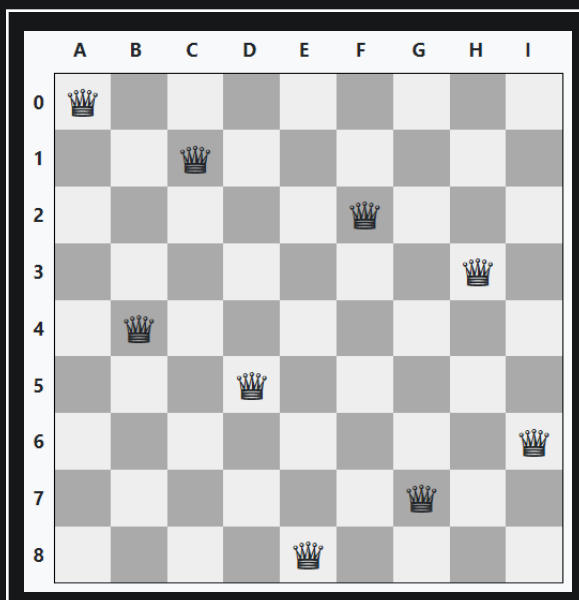


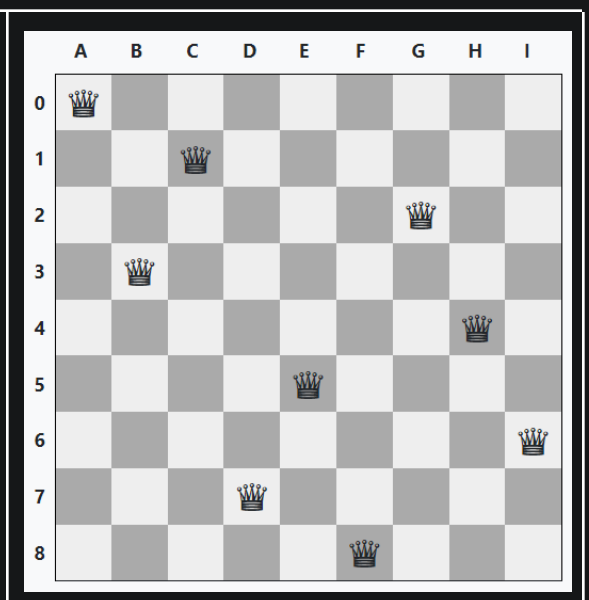
(0,0), (1,4), (2,7), (3,5), (4,2), (5,6), (6,1), (7,3)
(0,0), (1,5), (2,7), (3,2), (4,6), (5,3), (6,1), (7,4)
(0,0), (1,6), (2,3), (3,5), (4,7), (5,1), (6,4), (7,2)
(0,0), (1,6), (2,4), (3,7), (4,1), (5,3), (6,5), (7,2)
(0,1), (1,3), (2,5), (3,7), (4,2), (5,0), (6,6), (7,4)
(0,1), (1,4), (2,6), (3,0), (4,2), (5,7), (6,5), (7,3)
(0,1), (1,4), (2,6), (3,3), (4,0), (5,7), (6,5), (7,2)
(0,1), (1,5), (2,0), (3,6), (4,3), (5,7), (6,2), (7,4)
(0,1), (1,5), (2,7), (3,2), (4,0), (5,3), (6,6), (7,4)
(0,1), (1,6), (2,2), (3,5), (4,7), (5,4), (6,0), (7,3)
(0,1), (1,6), (2,4), (3,7), (4,0), (5,3), (6,5), (7,2)
(0,1), (1,7), (2,5), (3,0), (4,2), (5,4), (6,6), (7,3)
(0,2), (1,0), (2,6), (3,4), (4,7), (5,1), (6,3), (7,5)
(0,2), (1,6), (2,1), (3,7), (4,5), (5,3), (6,0), (7,4)
(0,2), (1,6), (2,1), (3,7), (4,4), (5,0), (6,3), (7,5)
(0,2), (1,5), (2,1), (3,4), (4,7), (5,0), (6,6), (7,3)
(0,2), (1,5), (2,1), (3,6), (4,0), (5,3), (6,7), (7,4)
(0,2), (1,5), (2,1), (3,6), (4,4), (5,0), (6,7), (7,3)
(0,2), (1,5), (2,3), (3,0), (4,7), (5,4), (6,6), (7,1)
(0,2), (1,5), (2,3), (3,1), (4,7), (5,4), (6,6), (7,0)
(0,2), (1,5), (2,7), (3,0), (4,3), (5,6), (6,4), (7,1)
(0,2), (1,5), (2,7), (3,0), (4,4), (5,6), (6,1), (7,3)
(0,2), (1,5), (2,7), (3,1), (4,3), (5,0), (6,6), (7,4)
(0,2), (1,4), (2,1), (3,7), (4,0), (5,6), (6,3), (7,5)
(0,2), (1,4), (2,1), (3,7), (4,5), (5,3), (6,6), (7,0)
(0,2), (1,4), (2,6), (3,0), (4,3), (5,1), (6,7), (7,5)
(0,2), (1,4), (2,7), (3,3), (4,0), (5,6), (6,1), (7,5)
(0,2), (1,7), (2,3), (3,6), (4,0), (5,5), (6,1), (7,4)
(0,3), (1,0), (2,4), (3,7), (4,5), (5,2), (6,6), (7,1)
(0,3), (1,0), (2,4), (3,7), (4,1), (5,6), (6,2), (7,5)
(0,3), (1,1), (2,4), (3,7), (4,5), (5,0), (6,2), (7,6)
(0,3), (1,1), (2,6), (3,2), (4,5), (5,7), (6,0), (7,4)
(0,3), (1,1), (2,6), (3,2), (4,5), (5,7), (6,4), (7,0)
(0,3), (1,1), (2,6), (3,4), (4,0), (5,7), (6,5), (7,2)
(0,3), (1,1), (2,7), (3,4), (4,6), (5,0), (6,2), (7,5)

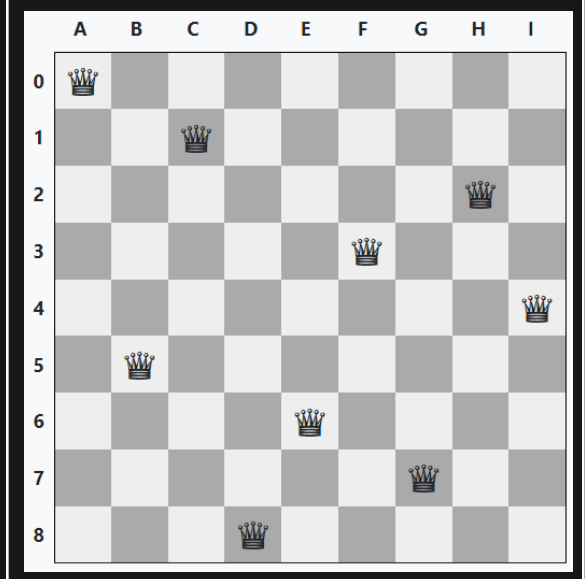
(0,4), (1,0), (2,3), (3,5), (4,7), (5,1), (6,6), (7,2)
(0,4), (1,0), (2,7), (3,3), (4,1), (5,6), (6,2), (7,5)
(0,4), (1,0), (2,7), (3,5), (4,2), (5,6), (6,1), (7,3)
(0,4), (1,1), (2,3), (3,6), (4,2), (5,7), (6,5), (7,0)
(0,4), (1,1), (2,3), (3,5), (4,7), (5,2), (6,0), (7,6)
(0,4), (1,1), (2,5), (3,0), (4,6), (5,3), (6,7), (7,2)
(0,4), (1,1), (2,7), (3,0), (4,3), (5,6), (6,2), (7,5)
(0,4), (1,2), (2,0), (3,5), (4,7), (5,1), (6,3), (7,6)
(0,4), (1,2), (2,0), (3,6), (4,1), (5,7), (6,5), (7,3)
(0,4), (1,2), (2,7), (3,3), (4,6), (5,0), (6,5), (7,1)
(0,4), (1,6), (2,1), (3,3), (4,7), (5,0), (6,2), (7,5)
(0,4), (1,6), (2,1), (3,5), (4,2), (5,0), (6,7), (7,3)
(0,4), (1,6), (2,1), (3,5), (4,2), (5,0), (6,3), (7,7)
(0,4), (1,6), (2,0), (3,2), (4,7), (5,5), (6,3), (7,1)
(0,4), (1,6), (2,0), (3,3), (4,1), (5,7), (6,5), (7,2)
(0,4), (1,6), (2,3), (3,0), (4,2), (5,7), (6,5), (7,1)
(0,4), (1,7), (2,3), (3,0), (4,2), (5,5), (6,1), (7,6)
(0,4), (1,7), (2,3), (3,0), (4,6), (5,1), (6,5), (7,2)
(0,5), (1,0), (2,4), (3,1), (4,7), (5,2), (6,6), (7,3)
(0,5), (1,1), (2,6), (3,0), (4,2), (5,4), (6,7), (7,3)
(0,5), (1,1), (2,6), (3,0), (4,3), (5,7), (6,4), (7,2)
(0,5), (1,2), (2,0), (3,7), (4,3), (5,1), (6,6), (7,4)
(0,5), (1,2), (2,0), (3,7), (4,4), (5,1), (6,3), (7,6)
(0,5), (1,2), (2,0), (3,6), (4,4), (5,7), (6,1), (7,3)
(0,5), (1,2), (2,6), (3,3), (4,0), (5,7), (6,1), (7,4)
(0,5), (1,2), (2,6), (3,1), (4,3), (5,7), (6,0), (7,4)
(0,5), (1,2), (2,6), (3,1), (4,7), (5,4), (6,0), (7,3)
(0,5), (1,2), (2,4), (3,7), (4,0), (5,3), (6,1), (7,6)
(0,5), (1,2), (2,4), (3,6), (4,0), (5,3), (6,1), (7,7)
(0,5), (1,3), (2,0), (3,4), (4,7), (5,1), (6,6), (7,2)
(0,5), (1,3), (2,1), (3,7), (4,4), (5,6), (6,0), (7,2)
(0,5), (1,3), (2,6), (3,0), (4,2), (5,4), (6,1), (7,7)
(0,5), (1,3), (2,6), (3,0), (4,7), (5,1), (6,4), (7,2)
(0,5), (1,7), (2,1), (3,3), (4,0), (5,6), (6,4), (7,2)
(0,6), (1,0), (2,2), (3,7), (4,5), (5,3), (6,1), (7,4)

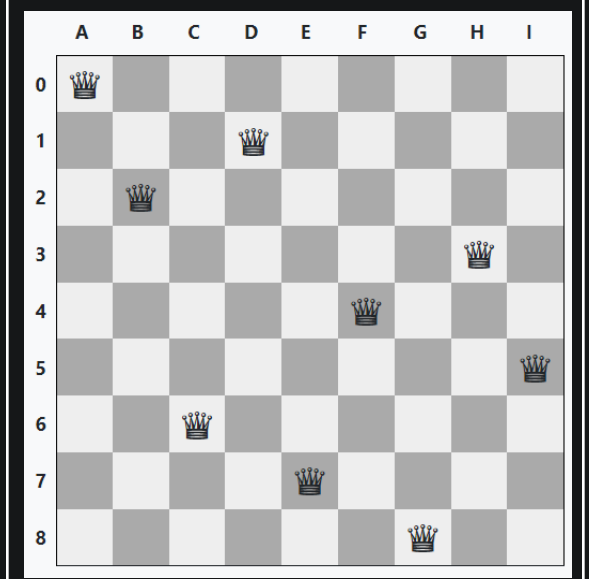
(0,3), (1,1), (2,7), (3,5), (4,0), (5,2), (6,4), (7,6)	(0,6), (1,1), (2,3), (3,0), (4,7), (5,4), (6,2), (7,5)
(0,3), (1,5), (2,0), (3,4), (4,1), (5,7), (6,2), (7,6)	(0,6), (1,1), (2,5), (3,2), (4,0), (5,3), (6,7), (7,4)
(0,3), (1,5), (2,7), (3,1), (4,6), (5,0), (6,2), (7,4)	(0,6), (1,2), (2,0), (3,5), (4,7), (5,4), (6,1), (7,3)
(0,3), (1,5), (2,7), (3,2), (4,0), (5,6), (6,4), (7,1)	(0,6), (1,2), (2,7), (3,1), (4,4), (5,0), (6,5), (7,3)
(0,3), (1,6), (2,2), (3,7), (4,1), (5,4), (6,0), (7,5)	(0,6), (1,3), (2,1), (3,4), (4,7), (5,0), (6,2), (7,5)
(0,3), (1,6), (2,0), (3,7), (4,4), (5,1), (6,5), (7,2)	(0,6), (1,3), (2,1), (3,7), (4,5), (5,0), (6,2), (7,4)
(0,3), (1,6), (2,4), (3,1), (4,5), (5,0), (6,2), (7,7)	(0,6), (1,4), (2,2), (3,0), (4,5), (5,7), (6,1), (7,3)
(0,3), (1,6), (2,4), (3,2), (4,0), (5,5), (6,7), (7,1)	(0,7), (1,2), (2,0), (3,5), (4,1), (5,4), (6,6), (7,3)
(0,3), (1,7), (2,0), (3,2), (4,5), (5,1), (6,6), (7,4)	(0,7), (1,3), (2,0), (3,2), (4,5), (5,1), (6,6), (7,4)
(0,3), (1,7), (2,0), (3,4), (4,6), (5,1), (6,5), (7,2)	(0,7), (1,1), (2,3), (3,0), (4,6), (5,4), (6,2), (7,5)
(0,3), (1,7), (2,4), (3,2), (4,0), (5,6), (6,1), (7,5)	(0,7), (1,1), (2,4), (3,2), (4,0), (5,6), (6,3), (7,5)

9-Queens Solutions:

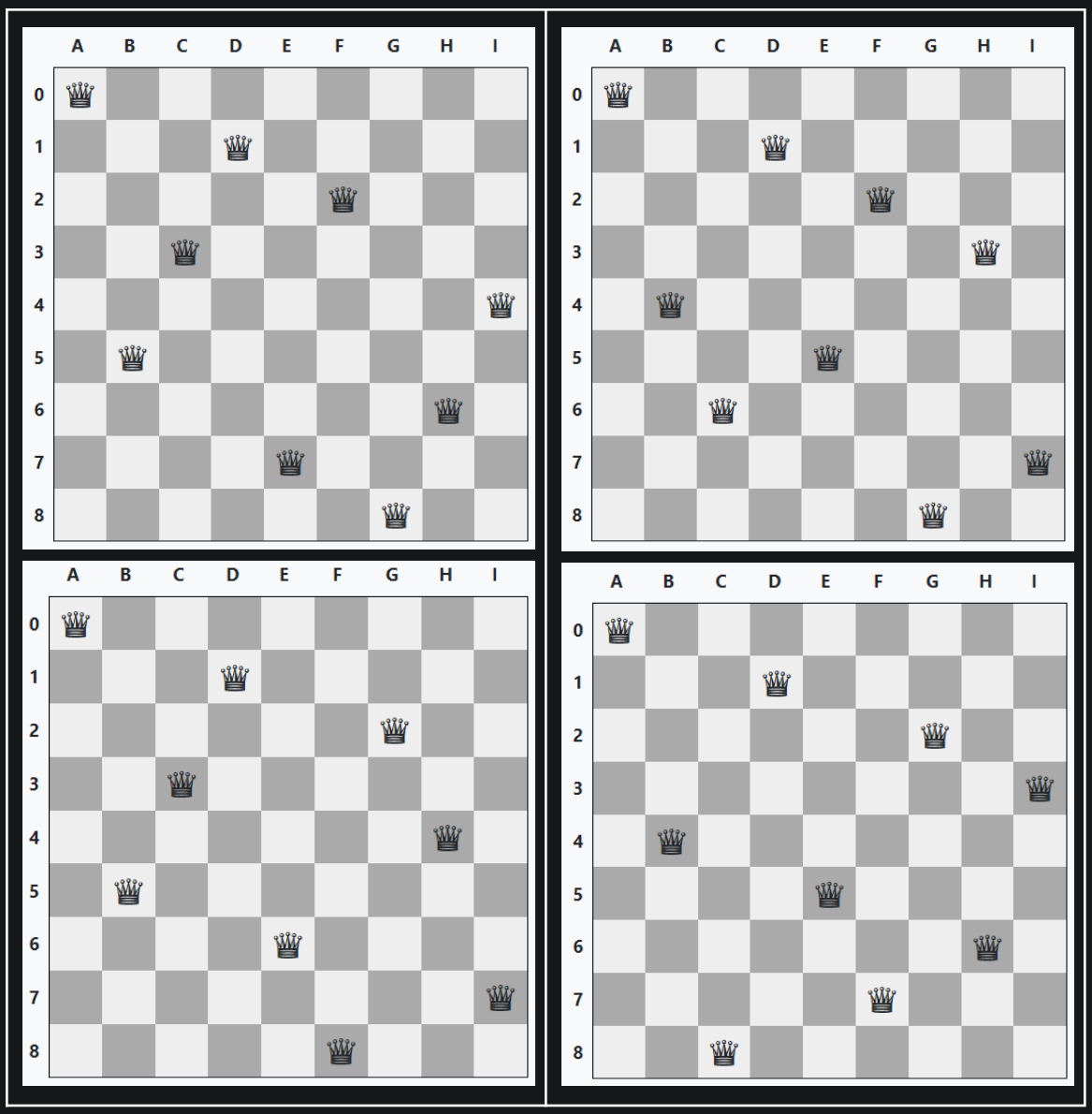




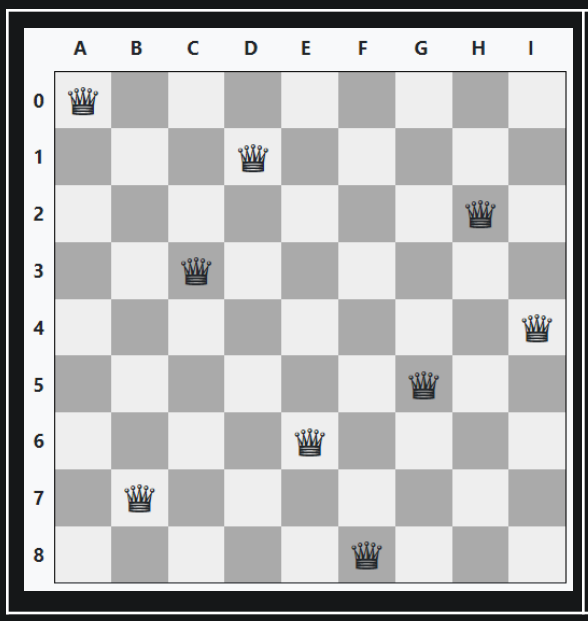


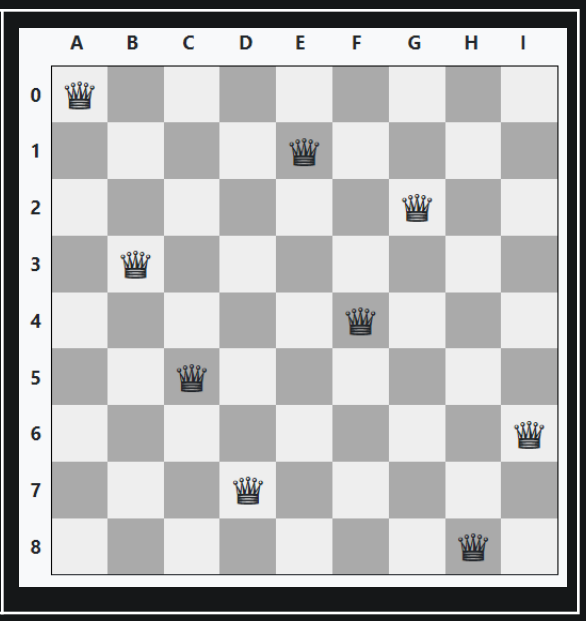


Alden O'Cain - 100558599
Reyhan Kogukoglu - 100779413



Alden O'Cain - 100558599
Reyhan Kogukoglu - 100779413





(0,0), (1,2), (2,5), (3,7), (4,1), (5,3), (6,8), (7,6), (8,4)
(0,0), (1,2), (2,6), (3,1), (4,7), (5,4), (6,8), (7,3), (8,5)
(0,0), (1,2), (2,7), (3,5), (4,8), (5,1), (6,4), (7,6), (8,3)
(0,0), (1,3), (2,1), (3,7), (4,5), (5,8), (6,2), (7,4), (8,6)
(0,0), (1,3), (2,5), (3,2), (4,8), (5,1), (6,7), (7,4), (8,6)
(0,0), (1,3), (2,5), (3,7), (4,1), (5,4), (6,2), (7,8), (8,6)
(0,0), (1,3), (2,6), (3,2), (4,7), (5,1), (6,4), (7,8), (8,5)
(0,0), (1,3), (2,6), (3,8), (4,1), (5,4), (6,7), (7,5), (8,2)
(0,0), (1,3), (2,7), (3,2), (4,8), (5,6), (6,4), (7,1), (8,5)
(0,0), (1,4), (2,6), (3,1), (4,5), (5,2), (6,8), (7,3), (8,7)
(0,0), (1,4), (2,6), (3,8), (4,2), (5,7), (6,1), (7,3), (8,5)
(0,0), (1,4), (2,6), (3,8), (4,3), (5,1), (6,7), (7,5), (8,2)
(0,0), (1,4), (2,1), (3,5), (4,8), (5,2), (6,7), (7,3), (8,6)
(0,0), (1,4), (2,8), (3,1), (4,5), (5,7), (6,2), (7,6), (8,3)
(0,0), (1,4), (2,8), (3,5), (4,3), (5,1), (6,7), (7,2), (8,6)
(0,0), (1,5), (2,1), (3,8), (4,6), (5,3), (6,7), (7,2), (8,4)
(0,0), (1,5), (2,7), (3,2), (4,6), (5,3), (6,1), (7,8), (8,4)
(0,0), (1,5), (2,7), (3,4), (4,1), (5,3), (6,8), (7,6), (8,2)
(0,0), (1,5), (2,3), (3,1), (4,6), (5,8), (6,2), (7,4), (8,7)
(0,0), (1,5), (2,3), (3,1), (4,7), (5,2), (6,8), (7,6), (8,4)
(0,0), (1,5), (2,8), (3,4), (4,1), (5,7), (6,2), (7,6), (8,3)
(0,0), (1,6), (2,3), (3,5), (4,8), (5,1), (6,4), (7,2), (8,7)
(0,0), (1,6), (2,3), (3,7), (4,2), (5,4), (6,8), (7,1), (8,5)
(0,0), (1,6), (2,3), (3,7), (4,2), (5,8), (6,5), (7,1), (8,4)
(0,0), (1,6), (2,4), (3,7), (4,1), (5,8), (6,2), (7,5), (8,3)
(0,0), (1,7), (2,3), (3,1), (4,6), (5,8), (6,5), (7,2), (8,4)
(0,0), (1,7), (2,4), (3,2), (4,5), (5,8), (6,1), (7,3), (8,6)
(0,0), (1,7), (2,4), (3,2), (4,8), (5,6), (6,1), (7,3), (8,5)
(0,1), (1,3), (2,0), (3,6), (4,8), (5,5), (6,2), (7,4), (8,7)
(0,1), (1,3), (2,6), (3,0), (4,2), (5,8), (6,5), (7,7), (8,4)
(0,1), (1,3), (2,7), (3,2), (4,8), (5,5), (6,0), (7,4), (8,6)
(0,1), (1,3), (2,8), (3,6), (4,2), (5,0), (6,5), (7,7), (8,4)
(0,1), (1,3), (2,8), (3,6), (4,4), (5,2), (6,0), (7,5), (8,7)
(0,1), (1,4), (2,6), (3,0), (4,2), (5,7), (6,5), (7,3), (8,8)
(0,1), (1,4), (2,6), (3,3), (4,0), (5,2), (6,8), (7,5), (8,7)

(0,4), (1,6), (2,0), (3,3), (4,1), (5,7), (6,5), (7,8), (8,2)
(0,4), (1,6), (2,0), (3,5), (4,7), (5,1), (6,3), (7,8), (8,2)
(0,4), (1,6), (2,1), (3,3), (4,7), (5,0), (6,2), (7,8), (8,5)
(0,4), (1,6), (2,1), (3,3), (4,7), (5,0), (6,8), (7,5), (8,2)
(0,4), (1,6), (2,1), (3,5), (4,2), (5,0), (6,7), (7,3), (8,8)
(0,4), (1,6), (2,1), (3,5), (4,7), (5,0), (6,3), (7,8), (8,2)
(0,4), (1,6), (2,3), (3,0), (4,2), (5,5), (6,8), (7,1), (8,7)
(0,4), (1,6), (2,3), (3,0), (4,2), (5,7), (6,5), (7,1), (8,8)
(0,4), (1,6), (2,3), (3,0), (4,2), (5,8), (6,5), (7,7), (8,1)
(0,4), (1,6), (2,3), (3,0), (4,7), (5,1), (6,8), (7,5), (8,2)
(0,4), (1,6), (2,8), (3,2), (4,7), (5,1), (6,3), (7,5), (8,0)
(0,4), (1,6), (2,8), (3,3), (4,1), (5,7), (6,5), (7,2), (8,0)
(0,4), (1,6), (2,8), (3,3), (4,7), (5,0), (6,2), (7,5), (8,1)
(0,4), (1,7), (2,1), (3,8), (4,2), (5,0), (6,6), (7,3), (8,5)
(0,4), (1,7), (2,1), (3,8), (4,5), (5,2), (6,0), (7,3), (8,6)
(0,4), (1,7), (2,1), (3,6), (4,2), (5,0), (6,8), (7,3), (8,5)
(0,4), (1,7), (2,1), (3,6), (4,2), (5,5), (6,8), (7,0), (8,3)
(0,4), (1,7), (2,3), (3,0), (4,2), (5,5), (6,8), (7,6), (8,1)
(0,4), (1,7), (2,3), (3,0), (4,6), (5,1), (6,5), (7,2), (8,8)
(0,4), (1,7), (2,3), (3,8), (4,6), (5,2), (6,0), (7,5), (8,1)
(0,4), (1,7), (2,0), (3,3), (4,6), (5,2), (6,5), (7,8), (8,1)
(0,4), (1,7), (2,0), (3,8), (4,3), (5,1), (6,6), (7,2), (8,5)
(0,4), (1,7), (2,5), (3,0), (4,2), (5,6), (6,8), (7,3), (8,1)
(0,4), (1,7), (2,5), (3,8), (4,2), (5,0), (6,6), (7,3), (8,1)
(0,4), (1,8), (2,1), (3,3), (4,6), (5,2), (6,7), (7,5), (8,0)
(0,4), (1,8), (2,1), (3,5), (4,7), (5,2), (6,0), (7,3), (8,6)
(0,4), (1,8), (2,3), (3,5), (4,7), (5,1), (6,6), (7,0), (8,2)
(0,5), (1,2), (2,0), (3,7), (4,3), (5,8), (6,6), (7,4), (8,1)
(0,5), (1,2), (2,0), (3,7), (4,4), (5,1), (6,8), (7,6), (8,3)
(0,5), (1,2), (2,0), (3,3), (4,6), (5,8), (6,1), (7,4), (8,7)
(0,5), (1,2), (2,6), (3,1), (4,3), (5,7), (6,0), (7,4), (8,8)
(0,5), (1,2), (2,6), (3,1), (4,3), (5,8), (6,0), (7,7), (8,4)
(0,5), (1,2), (2,6), (3,1), (4,7), (5,4), (6,0), (7,3), (8,8)
(0,5), (1,2), (2,6), (3,3), (4,0), (5,8), (6,1), (7,4), (8,7)
(0,5), (1,2), (2,8), (3,1), (4,4), (5,7), (6,0), (7,6), (8,3)

(0,1), (1,4), (2,6), (3,8), (4,2), (5,5), (6,3), (7,0), (8,7)	(0,5), (1,2), (2,8), (3,3), (4,0), (5,7), (6,1), (7,4), (8,6)
(0,1), (1,4), (2,6), (3,8), (4,3), (5,7), (6,0), (7,2), (8,5)	(0,5), (1,2), (2,8), (3,6), (4,0), (5,3), (6,1), (7,4), (8,7)
(0,1), (1,4), (2,7), (3,0), (4,2), (5,5), (6,8), (7,6), (8,3)	(0,5), (1,2), (2,4), (3,7), (4,0), (5,3), (6,1), (7,6), (8,8)
(0,1), (1,4), (2,7), (3,0), (4,8), (5,5), (6,2), (7,6), (8,3)	(0,5), (1,2), (2,4), (3,7), (4,0), (5,8), (6,3), (7,1), (8,6)
(0,1), (1,4), (2,7), (3,5), (4,8), (5,2), (6,0), (7,3), (8,6)	(0,5), (1,2), (2,4), (3,7), (4,0), (5,8), (6,6), (7,1), (8,3)
(0,1), (1,4), (2,7), (3,5), (4,8), (5,2), (6,0), (7,6), (8,3)	(0,5), (1,1), (2,8), (3,4), (4,2), (5,7), (6,3), (7,6), (8,0)
(0,1), (1,4), (2,8), (3,3), (4,0), (5,7), (6,5), (7,2), (8,6)	(0,5), (1,1), (2,4), (3,6), (4,8), (5,2), (6,7), (7,3), (8,0)
(0,1), (1,5), (2,0), (3,2), (4,6), (5,8), (6,3), (7,7), (8,4)	(0,5), (1,1), (2,4), (3,6), (4,8), (5,3), (6,7), (7,0), (8,2)
(0,1), (1,5), (2,0), (3,6), (4,3), (5,7), (6,2), (7,4), (8,8)	(0,5), (1,0), (2,4), (3,1), (4,8), (5,6), (6,3), (7,7), (8,2)
(0,1), (1,5), (2,0), (3,6), (4,4), (5,2), (6,8), (7,3), (8,7)	(0,5), (1,0), (2,4), (3,6), (4,8), (5,2), (6,7), (7,1), (8,3)
(0,1), (1,5), (2,0), (3,8), (4,4), (5,7), (6,3), (7,6), (8,2)	(0,5), (1,0), (2,4), (3,6), (4,8), (5,3), (6,1), (7,7), (8,2)
(0,1), (1,5), (2,2), (3,0), (4,7), (5,3), (6,8), (7,6), (8,4)	(0,5), (1,0), (2,6), (3,3), (4,7), (5,2), (6,4), (7,8), (8,1)
(0,1), (1,5), (2,8), (3,2), (4,4), (5,7), (6,3), (7,0), (8,6)	(0,5), (1,7), (2,0), (3,4), (4,8), (5,1), (6,3), (7,6), (8,2)
(0,1), (1,6), (2,4), (3,0), (4,8), (5,3), (6,5), (7,7), (8,2)	(0,5), (1,7), (2,0), (3,6), (4,3), (5,1), (6,8), (7,4), (8,2)
(0,1), (1,6), (2,4), (3,7), (4,0), (5,3), (6,5), (7,2), (8,8)	(0,5), (1,7), (2,1), (3,6), (4,0), (5,2), (6,4), (7,8), (8,3)
(0,1), (1,6), (2,8), (3,5), (4,2), (5,0), (6,3), (7,7), (8,4)	(0,5), (1,7), (2,2), (3,0), (4,8), (5,1), (6,4), (7,6), (8,3)
(0,1), (1,7), (2,0), (3,3), (4,6), (5,8), (6,5), (7,2), (8,4)	(0,5), (1,7), (2,2), (3,0), (4,8), (5,4), (6,1), (7,3), (8,6)
(0,1), (1,7), (2,4), (3,2), (4,8), (5,5), (6,3), (7,0), (8,6)	(0,5), (1,7), (2,2), (3,6), (4,8), (5,1), (6,4), (7,0), (8,3)
(0,1), (1,7), (2,5), (3,8), (4,2), (5,0), (6,3), (7,6), (8,4)	(0,5), (1,7), (2,4), (3,1), (4,8), (5,6), (6,3), (7,0), (8,2)
(0,1), (1,8), (2,4), (3,2), (4,7), (5,3), (6,6), (7,0), (8,5)	(0,5), (1,3), (2,0), (3,6), (4,8), (5,1), (6,7), (7,4), (8,2)
(0,1), (1,8), (2,5), (3,2), (4,4), (5,7), (6,0), (7,3), (8,6)	(0,5), (1,3), (2,1), (3,6), (4,8), (5,2), (6,4), (7,7), (8,0)
(0,1), (1,8), (2,5), (3,2), (4,6), (5,3), (6,0), (7,7), (8,4)	(0,5), (1,3), (2,1), (3,7), (4,4), (5,2), (6,0), (7,8), (8,6)
(0,1), (1,8), (2,5), (3,3), (4,6), (5,0), (6,2), (7,4), (8,7)	(0,5), (1,3), (2,1), (3,7), (4,4), (5,8), (6,0), (7,2), (8,6)
(0,2), (1,0), (2,3), (3,6), (4,8), (5,1), (6,4), (7,7), (8,5)	(0,5), (1,3), (2,1), (3,7), (4,2), (5,8), (6,6), (7,4), (8,0)
(0,2), (1,0), (2,5), (3,7), (4,4), (5,1), (6,3), (7,8), (8,6)	(0,5), (1,3), (2,6), (3,0), (4,2), (5,8), (6,1), (7,7), (8,4)
(0,2), (1,0), (2,6), (3,1), (4,7), (5,5), (6,3), (7,8), (8,4)	(0,5), (1,3), (2,6), (3,0), (4,7), (5,1), (6,4), (7,2), (8,8)
(0,2), (1,0), (2,6), (3,4), (4,7), (5,1), (6,3), (7,5), (8,8)	(0,5), (1,3), (2,6), (3,0), (4,7), (5,4), (6,1), (7,8), (8,2)
(0,2), (1,0), (2,7), (3,3), (4,8), (5,6), (6,4), (7,1), (8,5)	(0,5), (1,3), (2,8), (3,0), (4,2), (5,6), (6,1), (7,7), (8,4)
(0,2), (1,0), (2,8), (3,6), (4,4), (5,1), (6,7), (7,5), (8,3)	(0,5), (1,3), (2,8), (3,0), (4,4), (5,1), (6,7), (7,2), (8,6)
(0,2), (1,4), (2,1), (3,7), (4,0), (5,6), (6,3), (7,5), (8,8)	(0,5), (1,3), (2,8), (3,4), (4,7), (5,1), (6,6), (7,2), (8,0)
(0,2), (1,4), (2,1), (3,7), (4,0), (5,3), (6,6), (7,8), (8,5)	(0,5), (1,8), (2,0), (3,3), (4,6), (5,2), (6,7), (7,1), (8,4)
(0,2), (1,4), (2,6), (3,0), (4,3), (5,1), (6,7), (7,5), (8,8)	(0,5), (1,8), (2,2), (3,0), (4,7), (5,3), (6,1), (7,6), (8,4)
(0,2), (1,4), (2,7), (3,1), (4,8), (5,5), (6,0), (7,6), (8,3)	(0,5), (1,8), (2,4), (3,1), (4,7), (5,2), (6,6), (7,3), (8,0)
(0,2), (1,4), (2,7), (3,1), (4,8), (5,6), (6,0), (7,3), (8,5)	(0,5), (1,8), (2,4), (3,0), (4,7), (5,3), (6,1), (7,6), (8,2)
(0,2), (1,4), (2,8), (3,1), (4,3), (5,6), (6,0), (7,7), (8,5)	(0,5), (1,8), (2,4), (3,7), (4,0), (5,2), (6,6), (7,1), (8,3)
(0,2), (1,4), (2,8), (3,3), (4,0), (5,6), (6,1), (7,5), (8,7)	(0,5), (1,8), (2,6), (3,3), (4,0), (5,7), (6,1), (7,4), (8,2)
(0,2), (1,5), (2,1), (3,6), (4,0), (5,3), (6,7), (7,4), (8,8)	(0,6), (1,0), (2,3), (3,1), (4,7), (5,5), (6,8), (7,2), (8,4)
(0,2), (1,5), (2,1), (3,8), (4,4), (5,0), (6,7), (7,3), (8,6)	(0,6), (1,0), (2,3), (3,5), (4,8), (5,2), (6,4), (7,7), (8,1)
(0,2), (1,5), (2,7), (3,0), (4,3), (5,6), (6,4), (7,1), (8,8)	(0,6), (1,0), (2,3), (3,7), (4,4), (5,2), (6,8), (7,5), (8,1)
(0,2), (1,5), (2,7), (3,0), (4,4), (5,8), (6,1), (7,3), (8,6)	(0,6), (1,0), (2,5), (3,1), (4,4), (5,7), (6,3), (7,8), (8,2)
(0,2), (1,5), (2,7), (3,1), (4,3), (5,8), (6,6), (7,4), (8,0)	(0,6), (1,0), (2,5), (3,7), (4,1), (5,3), (6,8), (7,2), (8,4)
(0,2), (1,5), (2,7), (3,4), (4,0), (5,8), (6,6), (7,1), (8,3)	(0,6), (1,0), (2,5), (3,8), (4,1), (5,3), (6,7), (7,2), (8,4)
(0,2), (1,5), (2,7), (3,4), (4,1), (5,8), (6,6), (7,3), (8,0)	(0,6), (1,0), (2,7), (3,4), (4,1), (5,8), (6,2), (7,5), (8,3)
(0,2), (1,5), (2,8), (3,0), (4,7), (5,3), (6,1), (7,6), (8,4)	(0,6), (1,1), (2,3), (3,0), (4,7), (5,4), (6,8), (7,5), (8,2)
(0,2), (1,5), (2,8), (3,1), (4,4), (5,6), (6,3), (7,0), (8,7)	(0,6), (1,1), (2,3), (3,5), (4,0), (5,8), (6,4), (7,2), (8,7)
(0,2), (1,5), (2,8), (3,1), (4,7), (5,0), (6,3), (7,6), (8,4)	(0,6), (1,1), (2,3), (3,8), (4,0), (5,7), (6,4), (7,2), (8,5)
(0,2), (1,5), (2,8), (3,4), (4,7), (5,0), (6,3), (7,1), (8,6)	(0,6), (1,1), (2,5), (3,2), (4,0), (5,7), (6,4), (7,8), (8,3)
(0,2), (1,5), (2,8), (3,6), (4,0), (5,3), (6,1), (7,4), (8,7)	(0,6), (1,1), (2,7), (3,5), (4,0), (5,2), (6,4), (7,8), (8,3)
(0,2), (1,5), (2,8), (3,6), (4,1), (5,3), (6,7), (7,0), (8,4)	(0,6), (1,2), (2,0), (3,5), (4,7), (5,4), (6,1), (7,3), (8,8)
(0,2), (1,5), (2,8), (3,6), (4,3), (5,0), (6,7), (7,1), (8,4)	(0,6), (1,2), (2,0), (3,8), (4,4), (5,7), (6,1), (7,3), (8,5)
(0,2), (1,6), (2,1), (3,3), (4,7), (5,0), (6,4), (7,8), (8,5)	(0,6), (1,2), (2,5), (3,1), (4,4), (5,0), (6,8), (7,3), (8,7)
(0,2), (1,6), (2,1), (3,7), (4,4), (5,8), (6,0), (7,5), (8,3)	(0,6), (1,2), (2,5), (3,7), (4,0), (5,3), (6,8), (7,4), (8,1)
(0,2), (1,6), (2,1), (3,7), (4,5), (5,3), (6,0), (7,4), (8,8)	(0,6), (1,2), (2,5), (3,7), (4,0), (5,4), (6,8), (7,1), (8,3)
(0,2), (1,6), (2,3), (3,1), (4,8), (5,5), (6,0), (7,4), (8,7)	(0,6), (1,2), (2,7), (3,1), (4,3), (5,5), (6,8), (7,4), (8,0)
(0,2), (1,6), (2,3), (3,1), (4,8), (5,4), (6,0), (7,7), (8,5)	(0,6), (1,2), (2,7), (3,1), (4,4), (5,0), (6,8), (7,3), (8,5)
(0,2), (1,6), (2,3), (3,7), (4,4), (5,8), (6,0), (7,5), (8,1)	(0,6), (1,2), (2,7), (3,5), (4,1), (5,8), (6,4), (7,0), (8,3)
(0,2), (1,6), (2,8), (3,0), (4,4), (5,1), (6,7), (7,5), (8,3)	(0,6), (1,3), (2,1), (3,4), (4,7), (5,0), (6,2), (7,5), (8,8)
(0,2), (1,6), (2,8), (3,3), (4,1), (5,4), (6,7), (7,5), (8,0)	(0,6), (1,3), (2,1), (3,4), (4,8), (5,0), (6,2), (7,7), (8,5)
(0,2), (1,7), (2,1), (3,3), (4,8), (5,6), (6,4), (7,0), (8,5)	(0,6), (1,3), (2,1), (3,7), (4,5), (5,0), (6,2), (7,4), (8,8)
(0,2), (1,7), (2,3), (3,6), (4,8), (5,1), (6,4), (7,0), (8,5)	(0,6), (1,3), (2,1), (3,8), (4,4), (5,0), (6,7), (7,5), (8,2)
(0,2), (1,7), (2,5), (3,0), (4,8), (5,1), (6,4), (7,6), (8,3)	(0,6), (1,3), (2,1), (3,8), (4,5), (5,2), (6,4), (7,7), (8,0)

(0,2), (1,7), (2,5), (3,8), (4,1), (5,4), (6,0), (7,3), (8,6)	(0,6), (1,3), (2,0), (3,2), (4,5), (5,8), (6,1), (7,7), (8,4)
(0,2), (1,7), (2,5), (3,3), (4,8), (5,0), (6,4), (7,6), (8,1)	(0,6), (1,3), (2,0), (3,2), (4,7), (5,5), (6,1), (7,8), (8,4)
(0,2), (1,8), (2,1), (3,4), (4,7), (5,0), (6,6), (7,3), (8,5)	(0,6), (1,3), (2,0), (3,2), (4,8), (5,5), (6,7), (7,4), (8,1)
(0,2), (1,8), (2,3), (3,0), (4,7), (5,5), (6,1), (7,6), (8,4)	(0,6), (1,3), (2,0), (3,4), (4,1), (5,8), (6,5), (7,7), (8,2)
(0,2), (1,8), (2,3), (3,1), (4,7), (5,5), (6,0), (7,6), (8,4)	(0,6), (1,3), (2,0), (3,7), (4,1), (5,8), (6,5), (7,2), (8,4)
(0,2), (1,8), (2,3), (3,7), (4,4), (5,1), (6,5), (7,0), (8,6)	(0,6), (1,3), (2,0), (3,7), (4,4), (5,2), (6,5), (7,8), (8,1)
(0,2), (1,8), (2,5), (3,1), (4,4), (5,6), (6,0), (7,3), (8,7)	(0,6), (1,3), (2,0), (3,8), (4,1), (5,5), (6,7), (7,2), (8,4)
(0,2), (1,8), (2,5), (3,3), (4,0), (5,6), (6,4), (7,1), (8,7)	(0,6), (1,3), (2,7), (3,0), (4,4), (5,8), (6,1), (7,5), (8,2)
(0,2), (1,8), (2,5), (3,7), (4,1), (5,3), (6,0), (7,6), (8,4)	(0,6), (1,3), (2,7), (3,2), (4,8), (5,5), (6,1), (7,4), (8,0)
(0,3), (1,1), (2,4), (3,7), (4,0), (5,2), (6,5), (7,8), (8,6)	(0,6), (1,4), (2,0), (3,5), (4,8), (5,2), (6,7), (7,3), (8,1)
(0,3), (1,1), (2,7), (3,2), (4,8), (5,6), (6,4), (7,0), (8,5)	(0,6), (1,4), (2,0), (3,7), (4,5), (5,2), (6,8), (7,1), (8,3)
(0,3), (1,1), (2,6), (3,2), (4,0), (5,7), (6,4), (7,8), (8,5)	(0,6), (1,4), (2,1), (3,7), (4,0), (5,2), (6,8), (7,5), (8,3)
(0,3), (1,1), (2,6), (3,8), (4,0), (5,4), (6,7), (7,5), (8,2)	(0,6), (1,4), (2,1), (3,7), (4,0), (5,3), (6,8), (7,2), (8,5)
(0,3), (1,1), (2,6), (3,8), (4,0), (5,7), (6,4), (7,2), (8,5)	(0,6), (1,4), (2,2), (3,8), (4,5), (5,7), (6,1), (7,3), (8,0)
(0,3), (1,1), (2,8), (3,2), (4,5), (5,7), (6,0), (7,4), (8,6)	(0,6), (1,4), (2,7), (3,1), (4,8), (5,2), (6,5), (7,3), (8,0)
(0,3), (1,1), (2,8), (3,4), (4,0), (5,7), (6,5), (7,2), (8,6)	(0,6), (1,4), (2,7), (3,1), (4,8), (5,5), (6,2), (7,0), (8,3)
(0,3), (1,0), (2,2), (3,5), (4,8), (5,1), (6,7), (7,4), (8,6)	(0,6), (1,8), (2,1), (3,5), (4,0), (5,2), (6,4), (7,7), (8,3)
(0,3), (1,0), (2,4), (3,1), (4,8), (5,6), (6,2), (7,7), (8,5)	(0,6), (1,8), (2,0), (3,2), (4,4), (5,7), (6,1), (7,3), (8,5)
(0,3), (1,0), (2,4), (3,7), (4,1), (5,6), (6,2), (7,5), (8,8)	(0,6), (1,8), (2,3), (3,1), (4,4), (5,7), (6,5), (7,0), (8,2)
(0,3), (1,0), (2,4), (3,8), (4,1), (5,5), (6,7), (7,2), (8,6)	(0,6), (1,8), (2,2), (3,4), (4,1), (5,7), (6,5), (7,3), (8,0)
(0,3), (1,0), (2,6), (3,8), (4,1), (5,5), (6,7), (7,2), (8,4)	(0,6), (1,8), (2,2), (3,7), (4,1), (5,3), (6,5), (7,0), (8,4)
(0,3), (1,0), (2,8), (3,5), (4,2), (5,6), (6,1), (7,7), (8,4)	(0,6), (1,8), (2,5), (3,2), (4,0), (5,7), (6,4), (7,1), (8,3)
(0,3), (1,5), (2,0), (3,4), (4,1), (5,7), (6,2), (7,6), (8,8)	(0,7), (1,0), (2,3), (3,5), (4,2), (5,8), (6,6), (7,4), (8,1)
(0,3), (1,5), (2,0), (3,8), (4,6), (5,2), (6,7), (7,1), (8,4)	(0,7), (1,0), (2,3), (3,6), (4,2), (5,5), (6,8), (7,1), (8,4)
(0,3), (1,5), (2,0), (3,8), (4,4), (5,7), (6,1), (7,6), (8,2)	(0,7), (1,0), (2,3), (3,6), (4,4), (5,1), (6,8), (7,5), (8,2)
(0,3), (1,5), (2,2), (3,8), (4,1), (5,4), (6,7), (7,0), (8,6)	(0,7), (1,0), (2,4), (3,6), (4,1), (5,5), (6,2), (7,8), (8,3)
(0,3), (1,5), (2,2), (3,8), (4,1), (5,7), (6,4), (7,6), (8,0)	(0,7), (1,1), (2,4), (3,6), (4,0), (5,3), (6,5), (7,8), (8,2)
(0,3), (1,5), (2,2), (3,8), (4,6), (5,0), (6,7), (7,1), (8,4)	(0,7), (1,1), (2,3), (3,0), (4,6), (5,8), (6,5), (7,2), (8,4)
(0,3), (1,5), (2,7), (3,1), (4,4), (5,0), (6,8), (7,6), (8,2)	(0,7), (1,1), (2,8), (3,5), (4,2), (5,0), (6,3), (7,6), (8,4)
(0,3), (1,5), (2,7), (3,1), (4,4), (5,6), (6,8), (7,0), (8,2)	(0,7), (1,2), (2,0), (3,3), (4,6), (5,8), (6,5), (7,1), (8,4)
(0,3), (1,5), (2,7), (3,1), (4,6), (5,0), (6,2), (7,4), (8,8)	(0,7), (1,2), (2,4), (3,1), (4,8), (5,5), (6,3), (7,6), (8,0)
(0,3), (1,5), (2,7), (3,2), (4,0), (5,6), (6,4), (7,1), (8,8)	(0,7), (1,2), (2,4), (3,8), (4,0), (5,5), (6,3), (7,1), (8,6)
(0,3), (1,5), (2,8), (3,2), (4,0), (5,7), (6,1), (7,4), (8,6)	(0,7), (1,3), (2,0), (3,6), (4,4), (5,1), (6,5), (7,8), (8,2)
(0,3), (1,7), (2,0), (3,4), (4,6), (5,1), (6,5), (7,2), (8,8)	(0,7), (1,3), (2,6), (3,8), (4,1), (5,5), (6,0), (7,2), (8,4)
(0,3), (1,7), (2,4), (3,2), (4,0), (5,5), (6,1), (7,8), (8,6)	(0,7), (1,3), (2,8), (3,0), (4,4), (5,1), (6,5), (7,2), (8,6)
(0,3), (1,7), (2,4), (3,2), (4,0), (5,6), (6,1), (7,5), (8,8)	(0,7), (1,3), (2,8), (3,2), (4,4), (5,6), (6,0), (7,5), (8,1)
(0,3), (1,6), (2,0), (3,2), (4,8), (5,5), (6,7), (7,4), (8,1)	(0,7), (1,3), (2,8), (3,2), (4,5), (5,1), (6,6), (7,4), (8,0)
(0,3), (1,6), (2,0), (3,5), (4,8), (5,1), (6,7), (7,4), (8,2)	(0,7), (1,3), (2,8), (3,6), (4,2), (5,0), (6,5), (7,1), (8,4)
(0,3), (1,6), (2,0), (3,7), (4,4), (5,1), (6,8), (7,2), (8,5)	(0,7), (1,4), (2,0), (3,5), (4,8), (5,1), (6,3), (7,6), (8,2)
(0,3), (1,6), (2,2), (3,5), (4,8), (5,0), (6,7), (7,4), (8,1)	(0,7), (1,4), (2,1), (3,3), (4,0), (5,6), (6,8), (7,2), (8,5)
(0,3), (1,6), (2,2), (3,7), (4,1), (5,4), (6,8), (7,5), (8,0)	(0,7), (1,4), (2,1), (3,3), (4,0), (5,6), (6,8), (7,5), (8,2)
(0,3), (1,6), (2,2), (3,7), (4,5), (5,0), (6,8), (7,1), (8,4)	(0,7), (1,4), (2,1), (3,8), (4,0), (5,3), (6,6), (7,2), (8,5)
(0,3), (1,6), (2,2), (3,7), (4,5), (5,1), (6,8), (7,4), (8,0)	(0,7), (1,4), (2,1), (3,8), (4,6), (5,3), (6,0), (7,2), (8,5)
(0,3), (1,6), (2,4), (3,1), (4,8), (5,0), (6,2), (7,7), (8,5)	(0,7), (1,4), (2,2), (3,5), (4,8), (5,6), (6,0), (7,3), (8,1)
(0,3), (1,6), (2,4), (3,1), (4,8), (5,0), (6,5), (7,7), (8,2)	(0,7), (1,4), (2,2), (3,0), (4,5), (5,1), (6,8), (7,6), (8,3)
(0,3), (1,6), (2,4), (3,1), (4,8), (5,5), (6,7), (7,2), (8,0)	(0,7), (1,4), (2,2), (3,0), (4,6), (5,3), (6,5), (7,8), (8,1)
(0,3), (1,6), (2,8), (3,1), (4,5), (5,0), (6,2), (7,4), (8,7)	(0,7), (1,4), (2,2), (3,8), (4,6), (5,1), (6,3), (7,5), (8,0)
(0,3), (1,6), (2,8), (3,1), (4,4), (5,7), (6,0), (7,2), (8,5)	(0,7), (1,5), (2,1), (3,6), (4,0), (5,3), (6,8), (7,4), (8,2)
(0,3), (1,6), (2,8), (3,5), (4,2), (5,0), (6,7), (7,4), (8,1)	(0,7), (1,5), (2,0), (3,2), (4,4), (5,6), (6,8), (7,3), (8,1)
(0,3), (1,8), (2,2), (3,5), (4,1), (5,6), (6,4), (7,0), (8,7)	(0,7), (1,5), (2,0), (3,2), (4,6), (5,8), (6,3), (7,1), (8,4)
(0,3), (1,8), (2,4), (3,2), (4,0), (5,5), (6,7), (7,1), (8,6)	(0,7), (1,5), (2,2), (3,8), (4,6), (5,0), (6,3), (7,1), (8,4)
(0,3), (1,8), (2,4), (3,2), (4,0), (5,6), (6,1), (7,7), (8,5)	(0,7), (1,5), (2,8), (3,2), (4,0), (5,3), (6,6), (7,4), (8,1)
(0,3), (1,8), (2,4), (3,7), (4,0), (5,2), (6,5), (7,1), (8,6)	(0,8), (1,4), (2,0), (3,3), (4,5), (5,7), (6,1), (7,6), (8,2)
(0,4), (1,0), (2,5), (3,3), (4,1), (5,7), (6,2), (7,8), (8,6)	(0,8), (1,4), (2,0), (3,7), (4,3), (5,1), (6,6), (7,2), (8,5)
(0,4), (1,0), (2,7), (3,3), (4,1), (5,6), (6,8), (7,5), (8,2)	(0,8), (1,4), (2,2), (3,0), (4,5), (5,7), (6,1), (7,3), (8,6)
(0,4), (1,0), (2,7), (3,5), (4,2), (5,6), (6,1), (7,3), (8,8)	(0,8), (1,4), (2,2), (3,0), (4,6), (5,1), (6,7), (7,5), (8,3)
(0,4), (1,1), (2,3), (3,0), (4,6), (5,8), (6,2), (7,5), (8,7)	(0,8), (1,4), (2,2), (3,7), (4,3), (5,6), (6,0), (7,5), (8,1)
(0,4), (1,1), (2,3), (3,8), (4,6), (5,2), (6,0), (7,5), (8,7)	(0,8), (1,4), (2,7), (3,3), (4,0), (5,6), (6,1), (7,5), (8,2)
(0,4), (1,1), (2,5), (3,0), (4,2), (5,6), (6,8), (7,3), (8,7)	(0,8), (1,1), (2,4), (3,6), (4,0), (5,2), (6,7), (7,5), (8,3)
(0,4), (1,1), (2,5), (3,8), (4,2), (5,7), (6,3), (7,6), (8,0)	(0,8), (1,1), (2,4), (3,6), (4,3), (5,0), (6,7), (7,5), (8,2)
(0,4), (1,1), (2,5), (3,8), (4,6), (5,3), (6,0), (7,2), (8,7)	(0,8), (1,1), (2,5), (3,7), (4,2), (5,0), (6,3), (7,6), (8,4)

(0,4), (1,1), (2,7), (3,0), (4,6), (5,8), (6,2), (7,5), (8,3)	(0,8), (1,3), (2,5), (3,7), (4,1), (5,6), (6,0), (7,2), (8,4)
(0,4), (1,1), (2,7), (3,0), (4,3), (5,6), (6,8), (7,5), (8,2)	(0,8), (1,3), (2,5), (3,7), (4,2), (5,0), (6,6), (7,4), (8,1)
(0,4), (1,1), (2,7), (3,2), (4,6), (5,8), (6,0), (7,5), (8,3)	(0,8), (1,3), (2,0), (3,4), (4,7), (5,1), (6,6), (7,2), (8,5)
(0,4), (1,1), (2,7), (3,2), (4,6), (5,3), (6,0), (7,8), (8,5)	(0,8), (1,3), (2,7), (3,0), (4,2), (5,5), (6,1), (7,6), (8,4)
(0,4), (1,1), (2,8), (3,0), (4,5), (5,7), (6,2), (7,6), (8,3)	(0,8), (1,3), (2,1), (3,6), (4,2), (5,5), (6,7), (7,0), (8,4)
(0,4), (1,1), (2,8), (3,5), (4,2), (5,6), (6,3), (7,0), (8,7)	(0,8), (1,3), (2,1), (3,4), (4,7), (5,5), (6,0), (7,2), (8,6)
(0,4), (1,2), (2,0), (3,5), (4,1), (5,8), (6,6), (7,3), (8,7)	(0,8), (1,6), (2,1), (3,3), (4,0), (5,7), (6,4), (7,2), (8,5)
(0,4), (1,2), (2,0), (3,5), (4,7), (5,1), (6,3), (7,6), (8,8)	(0,8), (1,6), (2,2), (3,7), (4,1), (5,4), (6,0), (7,5), (8,3)
(0,4), (1,2), (2,0), (3,6), (4,1), (5,7), (6,5), (7,3), (8,8)	(0,8), (1,6), (2,3), (3,1), (4,7), (5,5), (6,0), (7,2), (8,4)
(0,4), (1,2), (2,5), (3,8), (4,1), (5,7), (6,0), (7,3), (8,6)	(0,8), (1,2), (2,5), (3,1), (4,6), (5,0), (6,3), (7,7), (8,4)
(0,4), (1,2), (2,5), (3,8), (4,6), (5,0), (6,3), (7,1), (8,7)	(0,8), (1,2), (2,5), (3,1), (4,6), (5,4), (6,0), (7,7), (8,3)
(0,4), (1,2), (2,5), (3,8), (4,6), (5,1), (6,3), (7,7), (8,0)	(0,8), (1,2), (2,5), (3,3), (4,0), (5,7), (6,4), (7,6), (8,1)
(0,4), (1,2), (2,5), (3,8), (4,6), (5,3), (6,0), (7,7), (8,1)	(0,8), (1,2), (2,4), (3,1), (4,7), (5,0), (6,6), (7,3), (8,5)
(0,4), (1,2), (2,7), (3,3), (4,1), (5,8), (6,5), (7,0), (8,6)	(0,8), (1,5), (2,1), (3,6), (4,0), (5,2), (6,4), (7,7), (8,3)
(0,4), (1,2), (2,7), (3,3), (4,6), (5,8), (6,1), (7,5), (8,0)	(0,8), (1,5), (2,3), (3,1), (4,7), (5,4), (6,6), (7,0), (8,2)
(0,4), (1,2), (2,7), (3,5), (4,1), (5,8), (6,0), (7,3), (8,6)	(0,8), (1,5), (2,3), (3,6), (4,0), (5,7), (6,1), (7,4), (8,2)
(0,4), (1,2), (2,7), (3,5), (4,1), (5,8), (6,6), (7,0), (8,3)	(0,8), (1,5), (2,7), (3,1), (4,3), (5,0), (6,6), (7,4), (8,2)
(0,4), (1,2), (2,8), (3,3), (4,1), (5,7), (6,5), (7,0), (8,6)	(0,8), (1,5), (2,2), (3,0), (4,7), (5,4), (6,1), (7,3), (8,6)
(0,4), (1,2), (2,8), (3,5), (4,7), (5,1), (6,3), (7,0), (8,6)	(0,8), (1,5), (2,2), (3,6), (4,1), (5,7), (6,4), (7,0), (8,3)

Runtime Comparisons:

Method	1st Sol (N=8)	1st Sol (N=9)	All Sols (N=8)	All Sols (N=9)
Iteration	~ 51 ms *	~ 510 ms *	126.7 ms **	1291.0 ms **
Recursion	~ 0 ms *	~ 1 ms *	4.4 ms **	30.3 ms **

* taken from a singular measurement
** taken from the average of 10 trials

Runtimes Discussion

Runtimes were calculated using the Date.now() function in JavaScript, 1) at the start of the procedure 2) after the first solution was found and 3) after the final board is checked. Once the process is complete then we take the difference between times (based on milliseconds since 1970) and update the disabled HTML fields visible when the timer window is toggled open.

From the table above, we can conclude that the recursion algorithm for solving N queens takes less time to run. To obtain all solutions, it took ~126ms iteratively, versus 4.4ms recursively. This significant difference between the runtimes can be attributed to the difference in run time complexities of the programs.

For the iterative solution, we created all the possible permutations of the rows. This took O(n!) assignments, which, for 8-Queens, is over 40 thousand permutations. For 9-Queens, this is over 300 thousand permutations. The large number of assignments in the code is the cause for the longer running time.

To contrast the iterative solutions run time, the recursive algorithm took less time to run. We approached the recursive algorithm more efficiently, leading to a shorter run time. Although there are

some functions that take O(n^2) time, for the purposes of this assignment, where our n values were not very large, the recursive algorithm worked very well in terms of running time. This recursive algorithm is a less brute-force approach. It is clear that the recursive algorithm is better suited for this problem, compared to iteratively, in terms of time complexity.

Conclusions:

N-Queens is a challenging problem for a programmer, both to solve iteratively and recursively. Many hours were spent on this project, in terms of brainstorming and researching the problem, developing the initial board checks, and deciding on what language to use, etc. Ultimately we elected to try this programming challenge in TypeScript because we wanted a project that would make us more comfortable with TS as well as working with web pages. Rather than printing our findings into the command line and building a CLI based program we elected to use chess board HTML found online (StackOverflow) to make our program come to life. There were many lessons learned throughout this project and some further questions have arised. What Data Structure was most suitable for this problem? How should we have approached building the algorithms required? What are the best ways to debug a program like this when it is not working correctly? In the end the assignment was complete, however, it took a considerable amount of time.

Resources:

1. HTML/CSS Chessboard (StackOverflow)
   https://stackoverflow.com/questions/26432492/chessboard-html5-only

2. Numberphile video
   https://www.youtube.com/watch?v=jPcBU0Z2Hj8

3. Abdul Bari video
   https://www.youtube.com/watch?v=xFv_Hl4B83A

4. Wikipedia page
   Eight queens puzzle - Wikipedia

5. TypeScript / JavaScript / jQuery

6. HTML / CSS / SASS / Bootstrap CSS / PHP

7. Carbon to visualize code
   Carbon | Create and share beautiful images of your source code

8. JetBrains IDE's WebStorm and PHPStorm, CodeWithMe