**Week 3: Pseudo-code and Flowchart for solving 8 queens problem**

*Submitted by: Sandip Sahani, Iksha Gurung, Milan Thapa*

Before we write the pseudo-code so solving 8 queens problem, we first give a brief explanation of the ideas that will be used to solve the 8 queen problem. As per our ongoing progress of project we have developed the concept of what 8 queen problem is and how we use backtracking to solve this problem.

The given pseudo-code uses following steps to solve the 8 queen’s problem:

1. We begin by placing first queen in first column, first row.
2. We then go on to place second queen in second column, first row. If the queen is under attack, we move it down the row (keeping the column constant) by one position until it is at safe position.
3. We carry out the same step for remaining queens. If we run out of safe position in a given column for given queen, we backtrack the process by moving the previously placed queen down the row and so forth. Until we find a safe spot for the current queen in hand.
4. Finally we have a solution when we have placed all the queens in the board so that no two queens can attack each other.

For the given pseudo-code to work we assume the existence of a data structure of record type called Queen whose schema is given below:

|  |
| --- |
| **Record** Queen  **Integer** row  **Integer** col  **End Record** |

Following is the pseudo-code for the process:

|  |
| --- |
| **SOLVE** (**Integer** boardSize, **Queen** queen[boardSize]);  i <- 0 *//Begin by placing the queen number 0*  **while** i < boardSize  queen[i].row <- queen[i].row + 1 *//Place queen[i] to next row*    */\* If queen[i] exceeds the row count, reset the queen and*  *re-place queen[i-1]*  */\**  **if**(queen[i].row >= boardSize)  queen[i] <- -1;  i <- i - 1;  **else**  *//While the queen[i] is under attack move it down the row*  **while**(is queen[i] under attack by any of the previously placed queens)  queen[i].row <- queen[i] + 1;  *//if queen[i] exceeds the row count, reset it, re-place queen[i-1]*  **if**(queen[i].row >= boardSize)  queen[i].row <- -1  i <- i - 1;  **else**  i++;  **end while** |

The flow chart for given algorithm is:

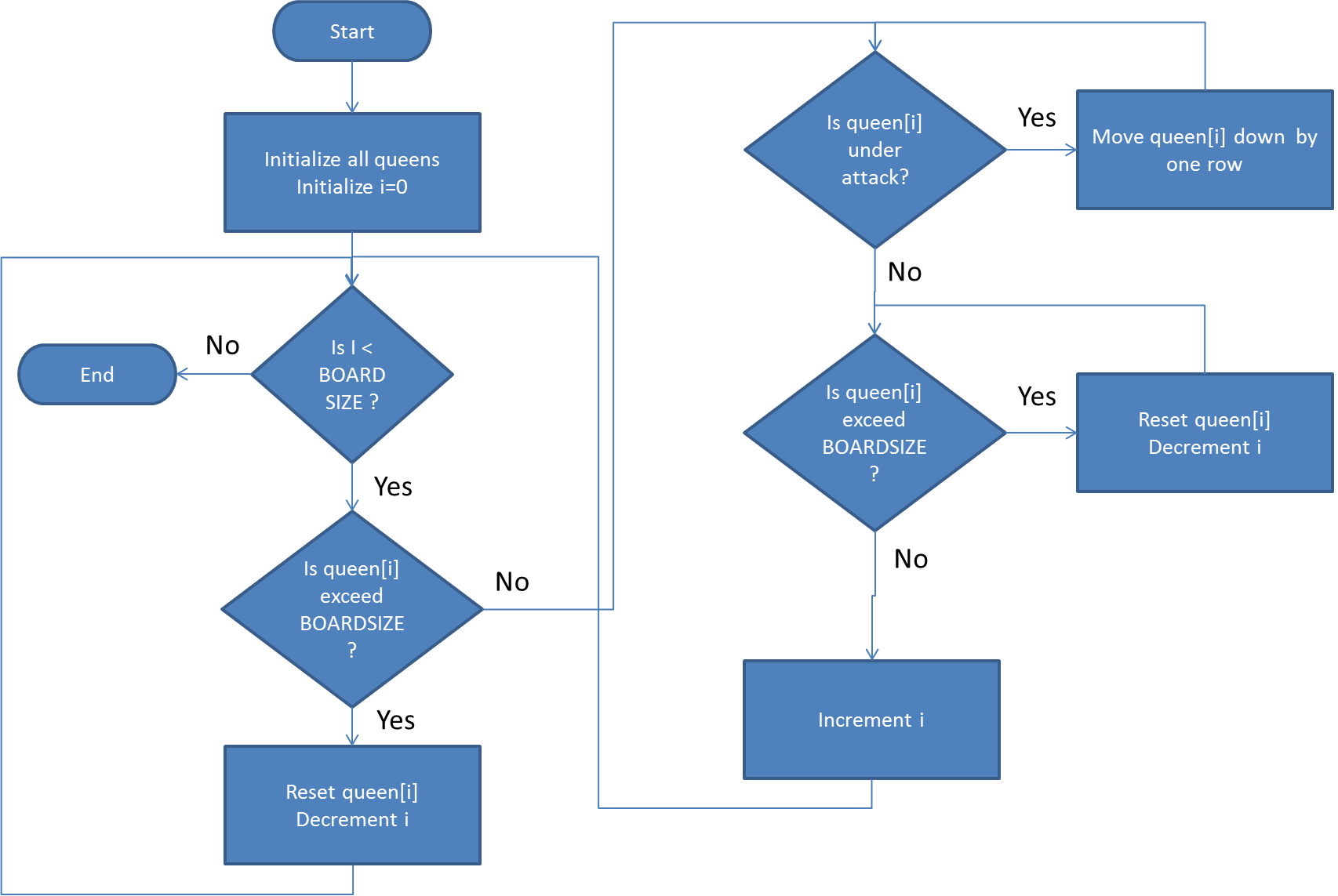


Figure : Flowchart for solving 8 queens problem

**Conclusion:**

Given pseudo-code and its corresponding flow chart can be used to solve in general n queen problem (n = 4, 5, 6…). Proposed backtracking method for solving 8 queens problem uses an iterative approach to solve the problem; however we can also find recursive implementation of backtracking algorithm to solve the same problem.

**Reference:**

1. <http://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=3&ved=0CGwQFjAC&url=http%3A%2F%2Fwww.cs.colorado.edu%2F~main%2Fsupplements%2Fpdf%2Fnotes07.pdf&ei=VYSqT6mtLMrorAf55oxX&usg=AFQjCNFqcGkDO7KuMFmfs5nAxvAnOkR31A> ( Accessed on May 9, 2012)
2. <http://en.wikipedia.org/wiki/Eight_queens_puzzle> (Accessed on May 9, 2012)
3. <http://stackoverflow.com/questions/4883118/8-queens-problem> (Accessed on May 9, 2012)
4. <http://stackoverflow.com/questions/3816437/asking-for-help-to-troubleshoot-a-c-eight-queens-puzzle-code> (Accessed on May 9, 2012)