**DAA ASSIGNMENT-4**

****

**SUBMITTED TO:**

**PROF. SUBHASH TATALE**

**VISHWAKARMA INSTITUTE OF INFORMATION TECHNOLOGY, PUNE**

**COMPUTER ENGINEERING DEPARTMENT**

**BY:**

**NAME: ABHISHEK MORE**

**G.R NO.: 21810033**

**ROLL NO.: 323036**

**CLASS: T.Y COMP**

**BATCH: COMP C2**

**Assignment-4**

**Aim:**

Implement Backtracking approach to find the solution for n-Queens problem.

**Objective:**

Our objective is to find solution for n-Queen problem using backtracking.

**Source code:**

**package** assignment4\_nQueenProblem;

**public** **class** NQueenProblem

{

**final** **int** N = 4;

**void** printSolution(**int** board[][])

{

**for** (**int** i = 0; i < N; i++)

{

**for** (**int** j = 0; j < N; j++)

{

System.***out***.print(" " + board[i][j] + " ");

}

System.***out***.println();

}

}

**boolean** isSafe(**int** board[][], **int** row, **int** col)

{

**int** i, j;

**for** (i = 0; i < col; i++)

{

**if** (board[row][i] == 1)

{

**return** **false**;

}

}

**for** (i = row, j = col; i >= 0 && j >= 0; i--, j--)

{

**if** (board[i][j] == 1)

{

**return** **false**;

}

}

**for** (i = row, j = col; j >= 0 && i < N; i++, j--)

{

**if** (board[i][j] == 1)

{

**return** **false**;

}

}

**return** **true**;

}

**boolean** solveNQUtil(**int** board[][], **int** col)

{

**if** (col >= N)

{

**return** **true**;

}

**for** (**int** i = 0; i < N; i++)

{

**if** (isSafe(board, i, col))

{

board[i][col] = 1;

**if** (solveNQUtil(board, col + 1) == **true**)

{

**return** **true**;

}

board[i][col] = 0; // BACKTRACK

}

}

**return** **false**;

}

**boolean** solveNQ()

{

**int** board[][] = { { 0, 0, 0, 0 },

{ 0, 0, 0, 0 },

{ 0, 0, 0, 0 },

{ 0, 0, 0, 0 } };

**if** (solveNQUtil(board, 0) == **false**) {

System.***out***.print("Solution does not exist");

**return** **false**;

}

printSolution(board);

**return** **true**;

}

**public** **static** **void** main(String args[])

{

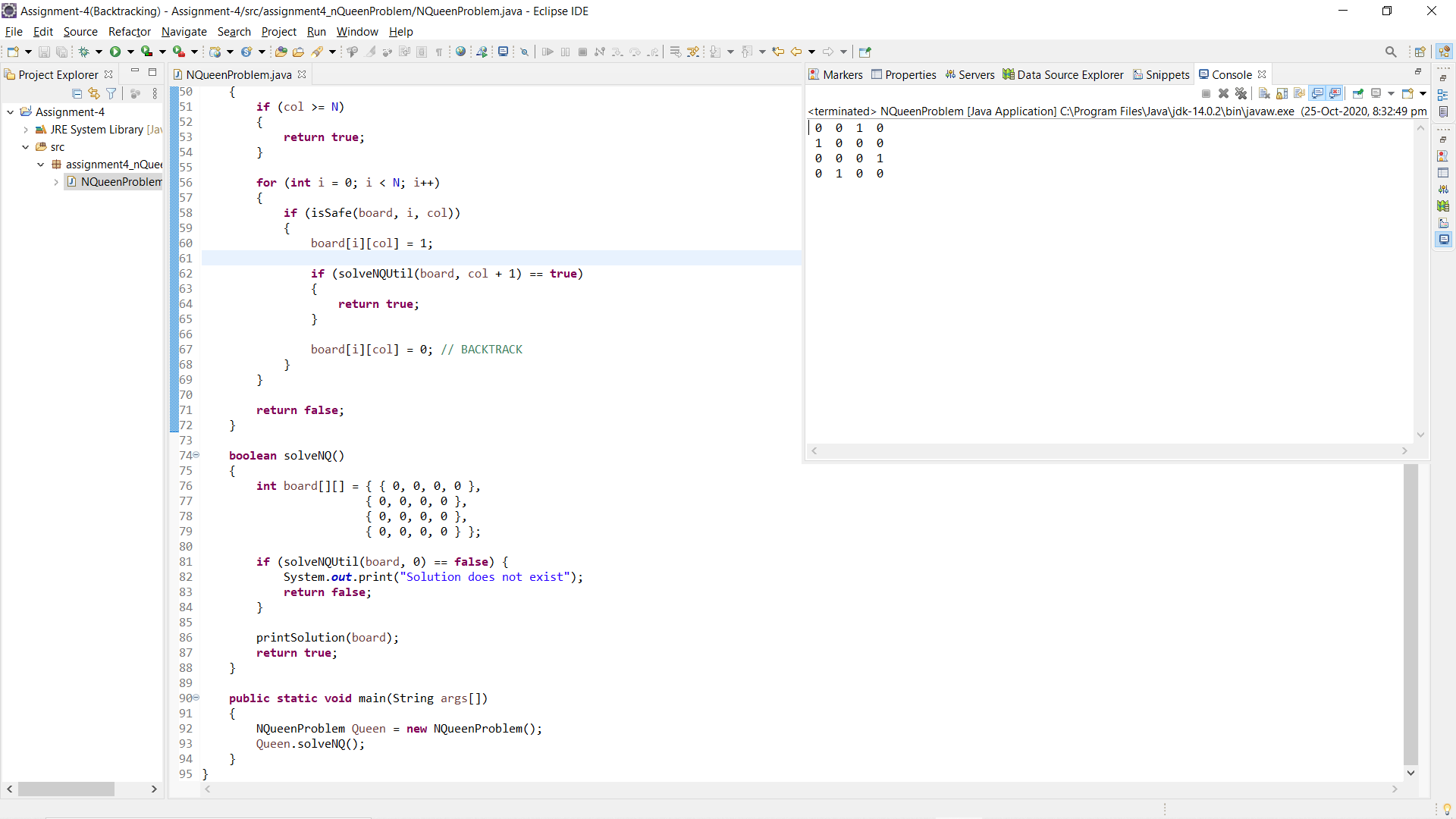
NQueenProblem Queen = **new** NQueenProblem();

Queen.solveNQ();

}

}

**Output:**

****

**Conclusion:**

Successfully found the solution for n-Queens problem using backtracking programming approach.