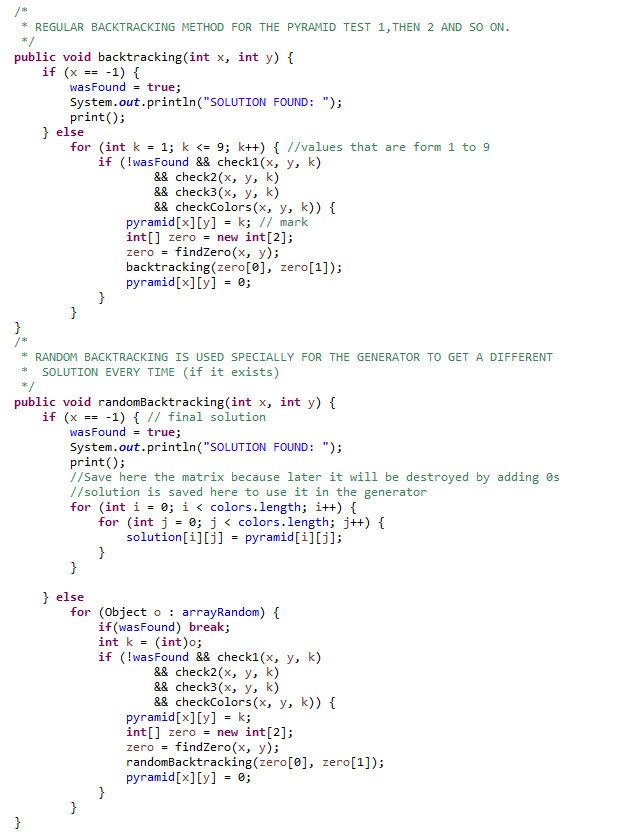
**Used algorithms in Pyramid Solver**

The main algorithm is the backtracking one which has a complexity of O(n!). For instance, in our case we must fill a pyramid with a Sudoku like algorithm. We have n = 9 as every cell can contain numbers from 1 to 9 (both included) then our algorithm will try with all the values removing one of them in every iteration. If the value accomplishes a series of conditions defined in the wording the value will be placed

I have created 2 versions:



**Used algorithms in Pyramid Generator**

In this case the most important algorithm is still the random backtracking solver, as it creates a different solution every time from an empty pyramid. Another important algorithm this time is the one which deals with placing the colors in a random position inside method createPyramid() which selects a random coordinate x and a coordinate y depending on the x in order to avoid iterations entering in the non pyramid area of the matrix.

**EXECUTION TIMES FROM PYRAMID SOLVER WITH RANDOM BACKTRACKING**

Case 1: 1 ms

Case 2: 1 ms

Case 3: 5 ms

Case 4: 5 ms

Case 5: 4 ms

Case 6: 1 ms

Case 7: 1 ms

Case 8: 0 ms

Case 9: 0 ms

Case 10: 0 ms

Case 11: 1 ms

Case 12: 3 ms

Case 13: 1 ms

Case 14: 2 ms

Case 15: 43 ms

Case 16: 106896 ms

Case 17: 1236 ms