Problem G. K-th Bishop Covering

Input file:	*standard input*
Output file:	*standard output*
Time limit:	5 seconds
Memory limit:	256 mebibytes

Little Dasha is learning to play chess. Recently she met the following problem: how many chess bishops are necessary to cover the whole chess board? The board is covered if each cell is covered. A cell is covered if a bishop stands there or at least one bishop attacks this cell. Recall that a chess bishop attacks a cell if it stands elsewhere but has a common diagonal with this cell, and additionally, there are no other pieces on the diagonal between the bishop and this cell.

The problem looked too hard for Dasha when using the standard 8×8 chess board, so she first solved it on square boards 1×1, 2×2, 3×3... Finally, she managed to place the bishops on the 8×8 board as well. She then looked at the provided answer, and amazingly, the number of bishops was right but the problem author placed them very differently in the answer. Could the problem have some other solutions as well? Dasha took a pen and a block of paper and sat, thinking.

Solve a generalized version of Dasha's problem. Given n and k, print the lexicographically k-th covering of the $n \times n$ square board by the minimum possible number of bishops.

Bishop placements are compared row by row starting from the top, and in each row, the cells are considered from left to right. If one finds the first cell in this order where two placements differ, the one where a bishop stands on this cell is lexicographically less than the other.

Input

The first line of the input contains two integers n and k: the size of the board and the number of the required covering ($1 \le n \le 50$). Coverings are numbered starting from 1. It is guaranteed that the k-th covering exists.

Output

Print exactly n lines, each containing exactly n characters: the lexicographically k-th way to cover an $n \times n$ board by the minimum possible number of bishops. Cells with bishops are denoted by "`*`" (asterisk, ASCII code 42), and empty cells are denoted by "`.`" (dot, ASCII code 46).

Examples

standard input	standard output
2 3	.* .*
3 2	.*. .** ...