Problem I: Who needs 8 Queens when you can have N?

The N-Queens problem is an obvious expansion of the eight-queens problem that has been around for a *long* time: Given a board with NxN squares and N queens, position the queens on the board such that no two queens can attack each other; in other words, so that no two queens sit on the same horizontal row, vertical column, or along either possible diagonal (row+column = k for one, row–column = k for the other).

Finding *all* the solutions to the problem for a given N is known to be worse than exponential in difficulty — O(N!). One can, however, find a *single* solution to the problem in significantly less time if one looks beyond the standard backtracking solution to another possible solution strategy.

This problem asks you to find such a solution strategy. Note that the solution found may not necessarily be the one obtained first in the backtracking approach.

Since there is a huge number of candidate solutions for any but the very smallest values of N, the judges have access to a solution validation program. Consequently it is critical that you abide by the output specifications, since they constitute the input specifications for the validator.

The Input (from file i.in)

The input file begins with a line containing a single integer (no white space) specifying the number of problem specifications in the file. Exactly that many lines follow, each giving the value of N (with no white space) for which you are to solve the N-queens problem. The values of N may range from 4 up through 300; in other words, you should be able to find ***some*** solution to the 300-Queens problem in less than 120 seconds.

The Output (to stdout)

For each problem, output on a single line that number (N). Following that, give the permutation vector of column positions (0 through N–1) that specify the queen's position on each succeeding row. The permutation vector is to be of N integers separated by white space. You may choose for yourself whether you use simple blanks for white space (giving the solution vector on one line) or you put each value on a separate line. In the interest of printing, the sample output here will use an approach that generates lines of values that do not exceed 65 characters in length. You are not held to that format.

Sample Input

```
4
4
8
25
50
```

Sample Output — of a very *large* number of possible outputs

```
4
 2 0 3 1
8
 2 7 3 6 0 5 1 4
25
 14 7 18 22 13 10 24 11 1 20 6 0 15 8 5 16 23 17 4 21 12 2 19 3 9
50
 22 13 15 44 27 3 1 4 19 40 20 5 31 49 7 29 18 6 2 36 28 12 38 43
 39 11 26 14 0 30 34 8 41 9 16 10 17 33 45 42 46 24 47 35 32 23
 25 37 21 48
```