Problem B
Grid Partition
Time Limit: 2.5 Seconds

Consider a grid of size $n \times n$ which has n^2 cells. Each cell in the grid is labeled with an integer less than or equal to n. Cells with the same label belong to a same group.

If two cells that share a side in the grid belong to the same group, they are said to be *connected*. If cell A is connected to cell B, and cell B and cell C are connected, then cell A is said to be connected to cell C as well.

Given a grid of size $n \times n$, we want to partition the cells in the grid into n groups by assigning labels to cells. The conditions that must be observed during partition are as follows.

1. Each group must contain n cells.
2. Cells belonging to the same group must be connected.

For example, when $n = 3$, the partition shown in Figure B-1(A) does not meet the conditions because cells in the group 1 (cells labeled 1) are not connected. Cells in the group 2 are not connected either. On the other hand, the other partitions (i.e., except (A)) shown in Figure B-1 satisfy the above conditions.

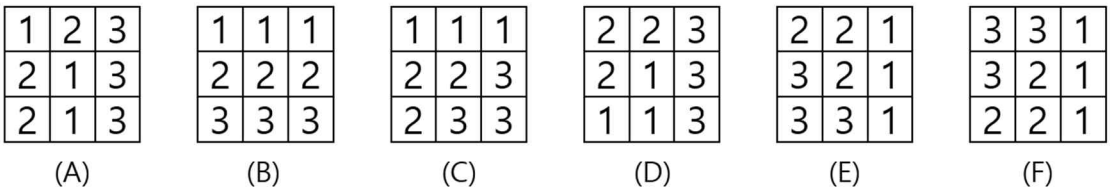

Figure B-1. Examples of grid partition

If you rotate the grid shown in Figure B-1(C) 90 degrees clockwise, it will be equal to Figure B-1(E) and if you flip it upside down, it will be Figure B-1(F). In short, the partition shown in Figure B-1(C) becomes the same partition as the partition shown in Figure B-1(F) through rotation and flipping, so these are considered the *same partition*. In this way, all partitions that obtained by applying consecutive rotations or flips to a partition are considered the same partition.

Comparing the partition shown in Figure B-1(D) with the partition shown in Figure B-1(F), only the assigned group numbers are different, however the partition itself can be seen as the same. Hence, the partition shown in Figure B-1(D) and that in shown B-1(F) are considered the same partition. In this way, grids with different labeling numbers but the same partitioning structure are all the same partition.

To be more precise, partition (C), partition (D), partition (E), and partition (F) shown in Figure B-1 are all the same.

Given an $n \times n$ grid with numbers pre-assigned to some cells, we want to assign a number to each unnumbered cell such that the resulting grid is partitioned so that it satisfies the conditions described above. Partitions can differ depending on how they assign numbers to unnumbered cells.

For example, suppose we are given as input the 4×4 grid shown in Figure B-2. Cells numbered 0 indicate that they are not assigned any number.

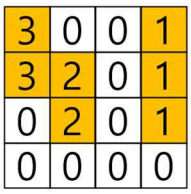

Figure B-2. An example of a 4x4 input grid

Figure B-3 shows three different results for partitioning the grid while maintaining the pre-assigned cell numbers in the input grid shown in Figure B-2.

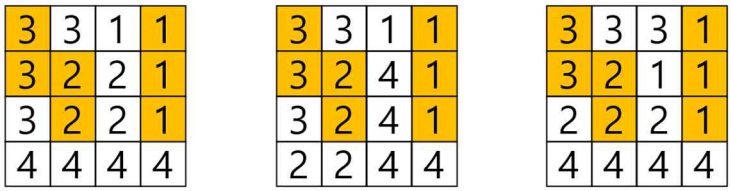

Figure B-3. Different partitions obtained from the input shown in Fig. B-2

Given an integer n representing the size of the grid, a positive integer k, and information on numbers pre-assigned to some cells in the $n \times n$ grid, you are to write a program which computes the total number of different partitions and finds k different partitions while maintaining the pre-assigned cell numbers. If there are no pre-assigned cells in the input grid, your program computes the number of all different partitions that exist in the input grid and finds any k different partitions among all the partitions.

Input

Your program is to read from standard input. The input starts with a line containing two integers, $n(4 \leq n \leq 6)$ and $k(1 \leq k \leq 20)$, where n indicates the size of a grid and k the number of different partitions to print out.

The i-th line of the following n lines contains n integers with no blank each of which is between 0 and n and represents the pre-assigned number for the corresponding cell on the i-th row of the grid. The integer 0 means that the corresponding cell is not assigned any number.

Note that given an input grid with pre-assigned numbers, following properties hold:

1. Cells pre-assigned by the same number are *connected*. In other words, cells in the same group are connected. And cells of number 0, which are actually not pre-assigned any number, are also connected.
2. The number of groups of cells determined by the pre-assigned numbers is 4 at most.

See that the above properties hold for the input shown in Figure B-2.

Output

Your program is to write to standard output. The first line contains the total number of all different partitions maintaining the pre-assigned cell numbers. In the following lines, the k different partitions will be printed.

In the first line of the i-th $(1 \leq i \leq k)$ partition, print '`Partition #i`' as shown in the following samples. In following n lines, print numbers assigned to cells in the grid. Note that the numbers assigned to cells are integers between 1 and n.

Since there may be many different partitions, you may print out any k different partitions in any order. You can assume that k possible partitions always exist for any input.

The following shows sample input and output for one test case.

Sample Input 1	Output for the Sample Input 1
4 3 3001 3201 0201 0000	11 Partition #1 3311 3221 3241 3444 Partition #2 3241 3241 3241 3241 Partition #3 3211 3241 3241 3244