953 The Incredible Pile Machine

The "WeePileIT Industries" intends to conquer the market with an automatic system that organizes piles of scrambled things (bottles, cans, batteries, etc.) into individual piles. They are developing a multi-purpose machine that detects the type of an item placed anywhere in a scrambled pile and moves it to its destination pile (let us call this an "operation"). The problem is that, with more than two types of items available, it becomes difficult to determine which pile should be allocated to each type in order to minimize the number of move operations. For example, let us suppose we have three scrambled piles of three types of items (named "0", "1" and "2"), such as in figure 1

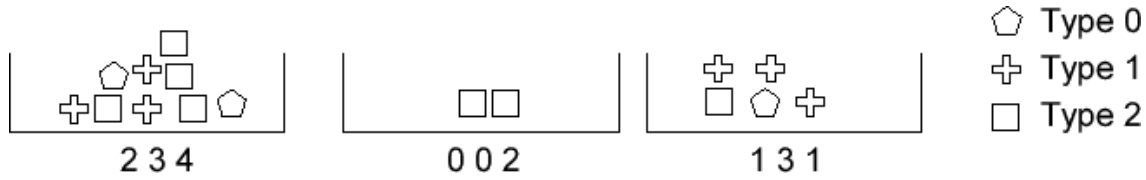


Figure 1: Three scrambled piles.

The numbers below each pile represent the distribution of types in that pile (in pile 1, we have 2 items of type 0, 3 items of type 1 and 4 items of type 2, hence "2 3 4"). Now, if we move the four items of type 2 from pile 1 to pile 2 and the single item of type 2 from pile 3 to pile 2, we get one unscrambled pile (pile 2), as we can see in figure 2

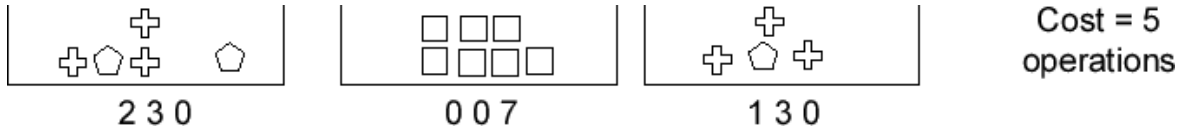


Figure 2: With 5 operations, pile 2 collects the elements of type 2.

The same reasoning could be followed to organize piles 1 and 3. In figure 3, we show a possible final result.

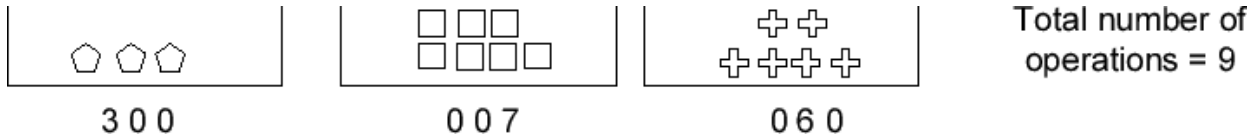


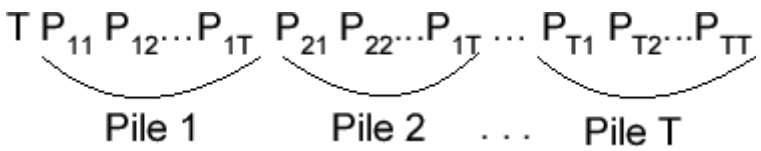
Figure 3: The final result, with 9 move operations.

As you can imagine, there is a large number of ways to organize our piles. And some surely need more operations than others...

Your task is to write a program for the "WeePileIT Industries" that calculates the minimum number of operations necessary for unscrambling a set of piles. The program must receive a description of the piles and return an indication of which pile will receive which type and the number of operations necessary. We assume that, for organizing T types, we will always need exactly T piles.

Input

The input starts with an isolated integer $N < 100$, indicating the number of test cases that follows. Then, each line of the input corresponds to an independent test case. Each test case starts with the number T (with $T < 10$) of different types of items (which also corresponds to the number of available piles). Then, for each pile P, there is a set of integers P_{ij} ($1 \leq i \leq T$; $1 \leq j \leq T$; $0 \leq P_{ij} \leq 1000$) representing all the types of items in that pile. Thus the input will be:



The input stream will end with an EOF sign. In the example above, the input would be:

```
3 2 3 4 0 0 2 1 3 1
```

Output

For each test case, your program must print a set of T integers, indicating, for each pile, which type should be assigned. The first integer will correspond to the type assigned to pile 1, the second to pile 2, and so on. After these integers, your program must print the minimum number, Min, of operations determined. Thus, the output will be:

$$P_1 P_2 ... P_T \; Min$$

In case there are more than one possibility of pile assignment, choose always the one with lower order (e.g. if 012 and 210 yield 9 operations, choose 012). In the example above, the output would be:

```
021 9
```

Sample Input

```
4
3 2 3 4 0 0 2 1 3 1
3 66 66 0 66 66 0 66 66 66
6 476 357 874 50 594 394 320 803 817 348 252 940 453 500 647 299
94 143 800 947 561 885 389 172 301 276 612 130 540 731 774 306 96
239 227 907
2 99 1 1 99
```

Sample Output

```
021 9
012 264
251340 12741
01 2
```