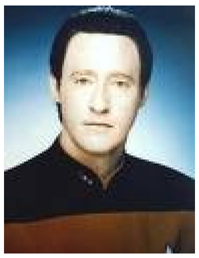
Optimal Strategy for the ICPC



Thanks in large part to the success of Mr. Data as an officer in Star Fleet, the International Collegiate Programming Contest, in a striking move towards android rights, has established a contest open to androids teams. Biological entities are allowed to compete, if they dare.

The following is taken from the official rules for the International Collegiate Programming Contest, as reported at **http://www.acmicpc-pacnw.org/rules.htm**

> The total time is the sum of the time consumed for each problem solved. The time consumed for a solved problem is the time elapsed from the beginning of the contest to the submittal of the first accepted run plus 20 penalty minutes for every previously rejected run for that problem. There is no time consumed for a problem that is not solved.

Quite simply, one element of the optimal strategy is ***not*** to have any erroneous submissions, so the androids do not have to worry about the penalty minutes. All that remains is to determine the order in which they should submit problems.

Let's assume perfect knowledge — hey, these androids are ***good*** — so that they can make a very good estimate of the development time required for each of the problems. The task is to determine the *order* in which the problems should be submitted. The androids realize that their best approach is for each to think independently about different problems rather than having all three work on a single problem. Furthermore, each android types infinitely fast, and does not use the computer terminal while thinking. Hence, up to three problems can be simultaneously in progress at any given time, and it is actually possible for all three bots to submit a problem within the same minute. For the same reason, the number of problems posed is larger than those posed in the contest for biological entities. Being innately fussy, if there are multiple ways to submit the problems and obtain the same score, they will submit the problem order that comes lexicographically first.

Determine the algorithm to solve the most problems and to obtain the best possible score for those problems. Then implement it.

Input

The first line of input to your program is a single integer **n** (**0 < n < 100**), giving the number of data sets — one for each set of problems. Following that are exactly **n** lines, giving information about each data set. The first number is the number of problems in that dataset as an integer **k** (**5 ≤ k ≤ 15**). On the same line, separated by single spaces, are **k** integers, all between 1 and 300 inclusive, giving the estimated time required to solve each problem. The problems themselves are labeled by alphabetic characters starting with A. Note that there are exactly 300 minutes in the contest.

Output

Each data set generates one line of output, giving the data set number, the sequence the problems are submitted, the total number solved, and the final penalty score. See the sample output for format — all non-blank entries are separated by single blank spaces.

Sample input

```
4
9 25 50 100 150 100 100 150 225 300
10 60 120 99 129 15 150 225 135 50 123
12 6 60 99 45 135 66 231 63 96 39 50 123
15 75 75 75 75 75 75 75 75 75 75 75 75 75 75 75
```

Sample output

```
Data set 1: A B C D E F G H 8 1450
Data set 2: E I A J C B F H D 9 1473
Data set 3: A J D B K F H I C E L 11 1452
Data set 4: A B C D E F G H I J K L 12 2250
```