COLEGIUL NAȚIONAL DE INFORMATICĂ TUDOR VIANU

ROMANIAN MASTER OF INFORMATICS 2019

Devil's Share

You are given a number, **X**. The devil wants his share of the number. He will take the largest subnumber with **K** digits. Minimize the devil's share by reordering the digits in number **X**.

Formally, you have at your disposal **S** ($1 \le S \le 100\ 000$) digits between **1** and **9**, inclusively. Given an integer **K** ($1 \le K \le S$), you are to create a number **X** using **all** the digits at your disposal, such that the largest length K substring of **X** is as small as possible.

Clarification: A length **K** substring of **X** is a base 10 integer comprising of **K** consecutive digits of **X** in the very same order. There are $S - K + 1$ such substrings in number **X**.

Input

The first line of input contains one integer **T** ($1 \le T \le 100\ 000$) - the number of test scenarios to analyse.

The description of **T** test scenarios follows. Each test scenario consists of two lines:

The first line contains one integer **K** - the length of all the substrings to consider.

The second line contains 9 space-separated integers: $D_1, D_2, ..., D_9$, where D_i represents the number of digits **i** at your disposal. ($0 \le D_i$, $D_1 + D_2 + ... + D_9 = S$).

The sum of **S** over all test scenarios will not exceed **1 000 000**.

Output

For each test scenario, print **X** - the number you created, on a separate line.

If there are several numbers **X** with the same smallest possible length **K** substring you can output any of them.

Subtasks

(1) $0 \le D_1, D_2, D_3, D_4 \le 3$, $D_5 = D_6 = ... = D_9 = 0$, $1 \le T \le 1536$, scenarios will not repeat **(13 points)**

(2) $K = 2$ **(14 points)**

(3) $D_3 = D_4 = ... = D_9 = 0$ **(29 points)**

(4) no additional constraints **(44 points)**

Example(s)

Standard Input	Standard Output
3 2 1 1 2 0 0 0 0 0 0 7 2 4 2 0 0 6 2 2 2 7 3 3 3 0 0 6 2 2 2	2313 62616236261623778899 62361623616236177889 9

Explanation:
There are three test scenarios to consider in the example.

In the first scenario **K = 2** and you have to arrange digits `1233`.
One optimal **X** is `2313`, with the following length `2` substrings: `23`, `31` and `13`, the largest being `31`. No other **X** has a smaller largest length `2` substring.
Another optimal **X** would be `3123`, since its largest length `2` substring is also `31`.

In the second scenario **K = 7** and you have to arrange digits `11222233666666778899`.
One optimal **X** is `62616236261623778899` with the largest length `7` substring `6261623`.

In the third scenario **K = 7** and you have to arrange digits `111222333666666778899`.
One optimal **X** is `623616236162361778899` with the largest length `7` substring `6236177`.