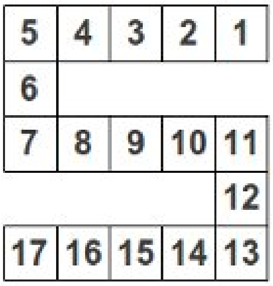
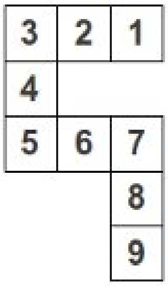
World Record Lunch

A group of people is trying to beat the world record for the largest number of people having lunch at the same time. In order achieve this goal, they are using the country's largest bridge and they have decided to arrange the tables following the shape of the letter 'S'.

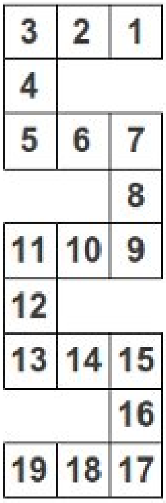
The table layout can be described by 4 integers: **NH**, **NV**, **H** and **V**. The two first integers, **NH** and **NV**, represent respectively the number or rows and number of columns in the layout. The last two integers represent respectively the number of tables in each row and column. For a given layout, the tables are numbered consecutively, starting with table #1 in the top-right corner. The following figure illustrates several possible layouts:



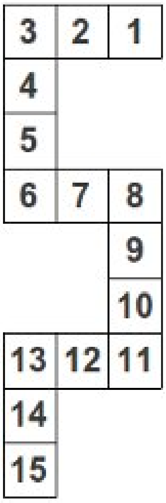
NH=3,NV=2,H=5,V=3



NH=3,NV=2,H=5,V=3



NH=3,NV=2,H=5,V=3



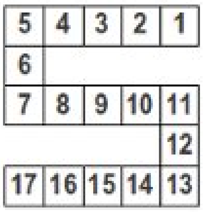
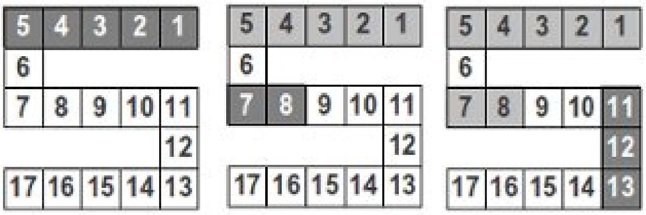
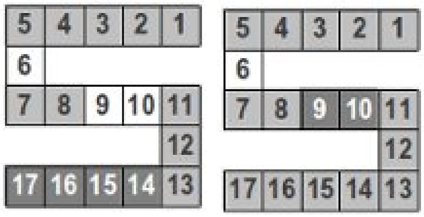
NH=3,NV=2,H=5,V=3

Thousands of groups of people are expected to come, and the organizers have to define where to seat everyone. Each group needs a certain number of tables and they do not share tables with other groups. Furthermore, a group wants their tables to be together and not split among rows and columns, that is, they want a set of consecutive tables either on the same row or on the same column. If this condition cannot be met, the group prefers to go away and have lunch at another place. The groups also enjoy having some privacy and prefer unoccupied adjacent tables, that is, no one at the table exactly before the first table of the group, and no one at the table exactly after the last table of the group. If this happens, we say that the group found a **private** place.

Whenever a group arrives, the organization must decide where to seat that group based solely on the table occupation at the moment, without taking into account further groups that may be coming. Since a group will always be seated on consecutive tables, the position of a group can be defined by the index of the first table of the group. So, when a group arrive, the organization wants the following:

- If there is a suitable private place for the group, then choose the lowest possible index guaranteeing privacy;
- If no privacy can be ensured, but there is space for the group, then choose the lowest possible index with space for the entire group;
- If there is no space available on a single row or column where the group would fit, then the organizers must send the group away.

An example would be the following. Imagine a layout with **NH**=3, **NV**=2, **H**=5 and **V**=3 (the first layout give on the figure above). And now imagine that groups needing 5, 2, 3, 5, 4 and 2 tables arrive, in that order. This is what happens:

	In the beginning all tables are empty
	Group 1 needs 5 tables. It is put on index 1. Group 2 needs 2 tables. It is put on index 7. Group 3 needs 3 tables. It is put on index 11. Group 4 needs 5 tables. No position is available.
	Group 5 needs 4 tables. It is put on index 14 (with no privacy). Group 6 needs 2 tables. It is put on index 9 (with no privacy).

Can you help the organizers on this task?

The Problem

Given a table layout (number of rows, columns and number of tables per row and column) and the description of groups arriving (specifying number of tables needed per group), your task is to calculate where each group should be seated following the rules described above.

Input

The first line contains 5 integers **NH NV H V N**, separated by single spaces. **NH** and **NV** indicate respectively the number of rows and columns of the table layout. **H** and **V** indicate respectively the number of tables per row and column. **N** indicates the number of groups arriving.

Then come **N** lines, each one indicating **G$_i$**, the number of tables the *i*-th group needs. These lines come in order of arrival of the groups.

Output

The output should have exactly **N** lines, each one indicating where the respective group should seat. If there is a suitable position, the line should contain an integer indicating the index of the first table of the group. If no position is available, the line should contain the string `"no"`, without the quotes.

Restrictions

The following limits are guaranteed for all the test cases that will be used for evaluating your program:

$1 \le NH \le 10\,000$	Number of rows
$NH-1 \le NV \le NH$	Number of columns
$3 \le H,V \le 1\,000$	Number of tables in each row/column
$1 \le N \le 50\,000$	Number of groups
$1 \le G_i \le 1\,000$	Number of tables each group needs

Example Input 1

```
3 2 5 3 6
5
2
3
5
4
2
```

Example Output 1

```
1
7
11
no
14
9
```

Input Explanation 1

It is the example given before in the problem statement.

Example Input 2

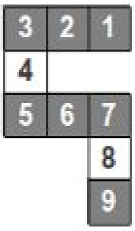
```
2 2 3 3 3
3
3
1
```

Example Output 2

```
1
5
9
```

Input Explanation 2

In the end the tables would look like this:



A few notes:

- This is the first problem I add to SPOJ, hope you like it :)
- The time limit is at least 6x the runtime of my C++ solution with no low-level optimizations or fast input/output. I have a Java solution passing within these time limits.
- I'm not proficient enough in languages like Python to ensure the time limit is ok for those languages. I can increase the limits if needed.