Problem 2 - Ghosting (`ghosting`)

On some keyboards, pressing certain keys combinations can result in a additional, unwanted key to be output. This phenomenon is known as *ghosting*.

In this challenge, you will be provided with a sequence of keypresses, and you will need to determine if the provided keystrokes induce ghosting, and if so which was the first key pressed that caused a potential ghost output.

In order understand how ghosting works, it is important to first understand how keyboards work in general.

Keyboard Matrix

Consider the following keyboard:

```
+---+---+---+---+---+---+---+---+---+---+
| Q | W | E | R | T | Y | U | I | O | P |
+---+---+---+---+---+---+---+---+---+---+
| A | S | D | F | G | H | J | K | L | ; |
+---+---+---+---+---+---+---+---+---+---+
| Z | X | C | V | B | N | M | , | . | / |
+---+---+---+---+---+---+---+---+---+---+
```

If we wanted to wire up each key to the keyboard controller, the controller would need 30 inputs, one for each key.

In practice, this is not feasible, and so keyboard controllers use process known as matrix scanning instead.

With a matrix scanning keyboard, each row of the keyboard is represented by a single wire, each column by a wire, and the keyboard keys connect a given row wire to a column wire when pressed.

```
Q--W--E--R--T--Y--U--I--O--P---- 0
 |  |  |  |  |  |  |  |  |  |
A--S--D--F--G--H--J--K--L--;---- 1
 |  |  |  |  |  |  |  |  |  |
Z--X--C--V--B--N--M--,--.--/---- 2
 |  |  |  |  |  |  |  |  |  |
 |  |  |  |  |  |  |  |  |  |
 0  1  2  3  4  5  6  7  8  9
```

In this configuration, when a user presses a key, that key's row and column are shorted together.

For example: When the user presses G, row 1 and column 4 are shorted together.

To detect keypresses, the keyboard controller applies a voltage to each column in turn and then checks each row for that same output. Then, using the row, column values, we can determine the pressed key(s).

Continuing our above example where the user has pressed G:

N.B: In this example, we represent the values of the keyboard rows as a string of bits, where 0 means there is no voltage present, and 1 means there is a voltage present.

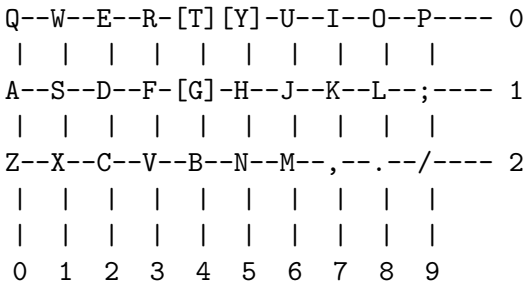
- T0: User presses G
- T1: Controller applies a voltage to column 0

- T2: Controller checks rows $0, 1, 2$ for output $\rightarrow$ 000
- T3: Controller turns off voltage for column 0
- T4: Controller applies a voltage to column 1
- T5: Controller checks rows $0, 1, 2$ for output $\rightarrow$ 000
- ...
- TN: Controller applies a voltage to column 4
- TM: Controller checks rows $0, 1, 2$ for output $\rightarrow$ 010

At TM, the controller detected a connection between column 4 and row 1, and deduces that the key G has been pressed.

Ghosting

Problems start to occur when we press multiple keys simultaneously. Consider the following key presses (T,G,Y):



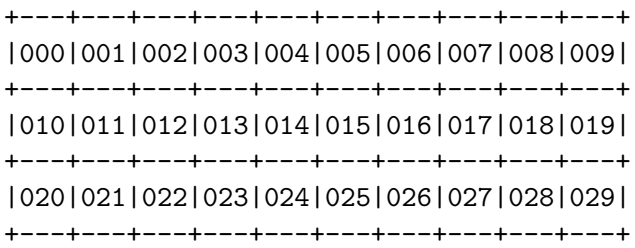
When the controller activates column 5, the voltage is able to flow through the Y switch to row 0, then through switch T into row 4 and finally through G into to row 1.

Therefore, when column 5 is active, rows 0 and 1 are active, and the controller deduces that keys Y and H are pressed!

Infinite keyboards

All the examples thus far have used a standard QWERTY keyboard. However, in this problem, we will consider any $R \times C$ keyboard. In order to not run out of letters, instead of giving each key a letter, we instead give it a number. Numbers are assigned sequentially from left to right, then top to bottom.

For example, a 9×3 keyboard would be numbered as follows:



The challenge

Your algorithm must ingest a sequence of keys, considered to be pressed in the order they appear, and the size of the keyboard, specified as R and C.

For a given sequence of keys and associated keyboard, your algorithm should indicate:

If a ghost key would be emmited by the keyboard controller in this case If so, the index of the first key pressed that introduced the possibility for ghosting.

Do not output the *ghost key*, but rather the key that introduced the possibility for ghosting.

Input data

In each file, the first line contains the number TC of testcases.

Each test case is in the format:

$R\ C\ N\ S_1\ S_2\ ...\ S_N$

- R: The number of rows on the provided keyboard
- C: The number of columns on the provided keyboard
- N: The number of keys pressed
- $S_1\ S_2\ ...\ S_N$ the sequence of key pressed.

Output data

The output file must contains T lines. For each test case in the input file, the output file must contains a line with the words:

Case #t: G

where t is the test case number (from 1 to T) and G is the solution of the testcase, where $G = -1$ if there is no ghosting or $0 \leq G < N$ indicating that the possibility of ghosting is introduced at $(G+1)^{th}$ key pressed.

Constraints

- $T \leq 20$
- $R, C \leq 5000$
- $N \leq 2000$
- $N \leq R \times C$
- The keyboard has a whole number of columns and rows.
- Each key in the provided sequence is unique.

Scoring

- **input 1** : $T = 1$, $R, C \leq 5$ and $N \leq 5$.
- **input 2** : $T = 5$, $R, C \leq 100$ and $N \leq 70$.
- **input 3** : $T = 10$, $R, C \leq 1000$ and $N \leq 150$.
- **input 4** : $T = 15$, $R, C \leq 3000$ and $N \leq 1000$.
- **input 5** : $T = 20$, $R, C \leq 5000$ and $N \leq 2000$.

Examples

input	output
5 2 2 3 0 1 2 5 3 3 8 1 14 5000 5000 1 1 100 2 4 50 60 61 199 5 5 4 3 8 13 18	Case #1: 2 Case #2: 1 Case #3: -1 Case #4: 2 Case #5: -1

Explanation

- On this 2×2 keyboard, the input introduces ghosting when you press the 3rd key in the sequence (2 in this case).
- On this 5×3 keyboard (5 rows and 3 columns), there is no ghosting.
- On this very large keyboard, the key 1 does not introduce ghosting.
- On this 100×2 keyboard (100 rows and 2 columns), the 3rd key (61 produces ghosting).