Basement | Advanced Division

Problem I: Camera Commands

Base Program Constraints: 1s, 256 MB

Clear Reward: +1 Point

Now that the crew is inside the casino's basement, they'll need an **exit route**! Unfortunately, the only exit is through a hallway covered with **security cameras**.

The exit hallway is split up into n **segments** sequentially labelled from 1 to n, with segment 1 at the opening of the hallway and segment n leading directly to the exit door. Each segment is monitored by exactly 1 **camera**, and each camera is in one of two states: `ON` or `OFF`.

The current state of all n cameras in the hallway can be represented by an n-bit bitstring, where the i-th bit from the left represents the state of the i-th camera (`0` for `OFF` and `1` for `ON`).

The security system has m camera **configurations** loaded onto it. Each of the m configurations has a unique ID from 1 to m and is represented by an n-bit bitstring as described above. Currently, configuration 1 is loaded onto the system.

After some digging, Olivia finds a terminal that she can use to send **commands** to the camera system. At any moment, she can send one of the following two commands:

- `DISABLE` a specific camera (set its state to `OFF` across all configurations)
- `SWITCH` to a specific camera configuration

Both operations are **instant**, and a person can only be in **one segment** at a time. (In other words, they cannot change segments during a `SWITCH`).

Using the terminal, Olivia thinks she can help everyone in the heist escape through the basement exit **without ever appearing** on a camera feed! Each command Olivia sends is logged, so to arouse the least suspicion, Olivia wants to do so with the **fewest number of commands**.

Help Olivia find an optimal sequence of commands to allow navigation from segment 1 to segment n of the hallway without ever being in a segment where a camera is `ON`.

Input

Each test contains multiple test cases. The first line of input contains the number of test cases t $(1 \leq t \leq 100)$.

The first line of each test case contains two integers, n and m $(1 \leq n, m \leq 1000)$, the number of cameras and the number of configurations respectively.

The next m lines each contain an n-bit bitstring, where the i-th line represents the i-th camera configuration.

It is guaranteed that the sum of $n * m$ across all test cases does not exceed 10^6.

Output

For each test case, output one integer - the minimum number of commands in the optimal sequence.

Polly starts outside of the hallway (not in segment 1), so Olivia can `SWITCH` the camera configuration before Polly enters the hallway. (This still counts as a command.)

Sample Test Cases

Sample 1 - Input

```
4
8 2
00000000
11111111
8 2
11111111
00000000
8 2
00011111
11111000
10 3
0010010110
1000100001
1111001000
```

Sample 1 - Output

```
0
1
3
3
```

Notes

To demonstrate the test cases, let's help Polly navigate from segment 1 to segment n of the hallway.

In the third test case, the following sequence of events uses only 3 commands:

- `DISABLE` the camera on segment 4
- `DISABLE` the camera on segment 5
- Polly moves to segment 5
- `SWITCH` to configuration 2

- Polly moves to segment 10

In the fourth test case, the following sequence of events uses only 3 commands:

- Polly moves to segment 2
- `SWITCH` to configuration 2
- Polly moves to segment 4
- `DISABLE` the camera on segment 5
- Polly moves to segment 9
- `SWITCH` to configuration 3
- Polly moves to segment 10

In both cases 3 and 4, there are additional sequences that also work, but none use less than 3 commands.