Problem 5 - Password

This year the Reply Code Challenge - Teen Edition is organized together with Gaspare and Luca (the two winners of the last Reply Code Challenge). To work with the old Code Masters they must log in to the Reply system but they forgot the password!

They are too scared to make a bad impression with the other Code Masters to ask for a password reset, so they decide to recover it by themselves.

Gaspare and Luca know that the Reply system is a very secure platform. They remember that the password is a binary number of length **N** and that inside the system there is a special alarm that is triggered each time someone inserts a wrong password which contains a specific binary number **S** of length **M**.

The only way to recover the password is to try all the combinations without sending the alarm to the old Code Masters.

Help the two new Code Masters to count how many safe passwords exist!

Important note: since the number of passwords could be very high, you are requested to only print the remainder[1] of this number when divided by 1 000 000 007.

Input data

The first line of the input file contains an integer **T**, the number of test cases to solve, followed by **T** testcases, numbered from **1** to **T**.

In each test case the first line contains the two integers **N** and **M**, the length of the password and the length of the forbidden sequence.

The second line contains the binary string **S** of length **M**.

Output data

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

Case #t: c

where *t* is the test case number (from **1** to **T**) and *c* is the total number of safe combinations (modulo 1 000 000 007).

Constraints

- $1 \leq \mathbf{T} \leq 4$.
- $1 \leq \mathbf{N} \leq 1\,000\,000\,000$.
- $1 \leq \mathbf{M} \leq 1100$.

[1]We remind you that the remainder of the division between one number and another is generally called modulo, and in many programming languages it is performed with the operator % (e.g. 13 % 5 = 3). Also remember that to avoid overflows is recommended to do ((A % R) + (B % R)) % R rather than (A + B) % R.

Scoring

- **input 1** : **T** $= 1$, **N** ≤ 5 and **M** ≤ 3.
- **input 2** : **T** $= 2$, **N** ≤ 20 and **M** ≤ 10.
- **input 3** : **T** $= 2$, **N** ≤ 1000, **M** ≤ 50 and **S** contains only 1.
- **input 4** : **T** $= 4$, **N** $\leq 100\,000$ and **M** ≤ 100.
- **input 5** : **T** $= 4$, **N** $\leq 1\,000\,000\,000$ and **M** ≤ 1100.

Examples

input	output
3 3 2 01 5 3 000 8 10 0000000000	Case #1: 4 Case #2: 24 Case #3: 256

Explanation

In the first test case we have 4 safe passwords (000, 100, 110, 111) and 4 passwords which contain the sequence **01** (0**01**, **01**0, **01**1, 1**01**).

In the second test case only 8 out of 32 possible passwords contain the sequence 000: **000**00, **000**01, **000**10, **000**11, 01**000**, 1**000**0, 1**000**1, 11**000**. The remaning 24 are instead safe.

In the last test case the length of the forbidden sequence is longer than the password, so all the possibile sequences (which are $2^8 = 256$) are safe.