A. Binary Matrix

```
Input file:     standard input
Output file:    standard output
Time limit:     1 second
Memory limit:   512 megabytes
```

A matrix is called binary if all its elements are either 0 or 1.

Ecrade calls a binary matrix A *good* if the following two properties hold:

- The bitwise XOR of all numbers in each row of matrix A is equal to 0.
- The bitwise XOR of all numbers in each column of matrix A is equal to 0.

Ecrade has a binary matrix of size $n \cdot m$. He is interested in the minimum number of elements that need to be changed for the matrix to become *good*.

However, it seems a little difficult, so please help him!

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 400$). The description of the test cases follows.

The first line of each test case contains two integers n, m ($1 \le n, m \le 100$).

This is followed by n lines, each containing exactly m characters consisting only of 0 and 1, describing the elements of Ecrade's matrix.

It is guaranteed that the sum of $n \cdot m$ across all test cases does not exceed $5 \cdot 10^4$.

Output

For each test case, output a single integer, the minimum number of elements that need to be changed.

Standard Input	Standard Output
7 3 3 010 101 010 3 3 000 000 000 3 3 100 010 001 3 3 101 010 000 3 3 000	2 0 3 3 1 2 2

010 000 1 4 0101 4 1 0 1 0 1	

Note

In the first test case, he needs to change 2 elements to obtain the following matrix $\begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix}$.

In the second test case, he can make no changes to obtain the following matrix $\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$.

In the third test case, he needs to change 3 elements to obtain the following matrix $\begin{pmatrix} 1 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 1 \end{pmatrix}$.

,

B. Floor or Ceil

```
Input file:     standard input
Output file:    standard output
Time limit:     1 second
Memory limit:   512 megabytes
```

Ecrade has an integer x. There are two kinds of operations.

1. Replace x with $\left\lfloor \frac{x}{2} \right\rfloor$, where $\left\lfloor \frac{x}{2} \right\rfloor$ is the greatest integer $\le \frac{x}{2}$.
2. Replace x with $\left\lceil \frac{x}{2} \right\rceil$, where $\left\lceil \frac{x}{2} \right\rceil$ is the smallest integer $\ge \frac{x}{2}$.

Ecrade will perform exactly n first operations and m second operations in any order. He wants to know the minimum and the maximum possible value of x after $n+m$ operations. However, it seems a little difficult, so please help him!

Input

Each test contains multiple test cases. The first line contains the number of test cases t $(1 \le t \le 10^4)$. The description of the test cases follows.

The only line of each test case contains three integers x, n, and m $(0 \le x, n, m \le 10^9)$.

Output

For each test case, print two integers in one line, representing the minimum and the maximum possible value of x after $n+m$ operations.

Standard Input	Standard Output
5 12 1 2 12 1 1 12 0 0 12 1000000000 1000000000 706636307 0 3	1 2 3 3 12 12 0 0 88329539 88329539

Note

For simplicity, we call the first operation $\text{OPER } 1$ and the second operation $\text{OPER } 2$.

In the first test case:

- If we perform $12 \xrightarrow{\text{OPER } 2} 6 \xrightarrow{\text{OPER } 2} 3 \xrightarrow{\text{OPER } 1} 1$, we can obtain the minimum possible value 1.
- If we perform $12 \xrightarrow{\text{OPER } 2} 6 \xrightarrow{\text{OPER } 1} 3 \xrightarrow{\text{OPER } 2} 2$, we can obtain the maximum possible value 2.

C. Math Division

Input file:	standard input
Output file:	standard output
Time limit:	2 seconds
Memory limit:	512 megabytes

Ecrade has an integer x. He will show you this number in the form of a binary number of length n.

There are two kinds of operations.

1. Replace x with $\left\lfloor \frac{x}{2} \right\rfloor$, where $\left\lfloor \frac{x}{2} \right\rfloor$ is the greatest integer $\leq \frac{x}{2}$.
2. Replace x with $\left\lceil \frac{x}{2} \right\rceil$, where $\left\lceil \frac{x}{2} \right\rceil$ is the smallest integer $\geq \frac{x}{2}$.

Ecrade will perform several operations until x becomes 1. Each time, he will independently choose to perform either the first operation or the second operation with probability $\frac{1}{2}$.

Ecrade wants to know the expected number of operations he will perform to make x equal to 1, modulo 10^9+7. However, it seems a little difficult, so please help him!

Input

The first line contains a single integer t ($1 \leq t \leq 10^5$) — the number of test cases. The description of the test cases follows.

The first line of each test case contains a single integer n ($1 \leq n \leq 10^5$) — the length of x in binary representation.

The second line of each test case contains a binary string of length n: the number x in the binary representation, presented from the most significant bit to the least significant bit. It is guaranteed that the most significant bit of x is 1.

It is guaranteed that the sum of n across all test cases does not exceed 10^5.

Output

For each test case, print a single integer representing the expected number of operations Ecrade will perform to make x equal to 1, modulo 10^9+7.

Formally, let $M = 10^9+7$. It can be shown that the exact answer can be expressed as an irreducible fraction $\frac{p}{q}$, where p and q are integers and $q \not\equiv 0 \pmod{M}$. Output the integer equal to $p \cdot q^{-1} \bmod M$. In other words, output such an integer x that $0 \leq x < M$ and $x \cdot q \equiv p \pmod{M}$.

Standard Input	Standard Output
3 3 110 3 100 10 1101001011	500000006 2 193359386

Note

For simplicity, we call the first operation $\text{OPER } 1$ and the second operation $\text{OPER } 2$.

In the first test case, $x = 6$, and there are six possible series of operations:

- $6 \xrightarrow{\text{OPER } 1} 3 \xrightarrow{\text{OPER } 1} 1$, the probability is $\frac{1}{4}$.
- $6 \xrightarrow{\text{OPER } 1} 3 \xrightarrow{\text{OPER } 2} 2 \xrightarrow{\text{OPER } 1} 1$, the probability is $\frac{1}{8}$.
- $6 \xrightarrow{\text{OPER } 1} 3 \xrightarrow{\text{OPER } 2} 2 \xrightarrow{\text{OPER } 2} 1$, the probability is $\frac{1}{8}$.
- $6 \xrightarrow{\text{OPER } 2} 3 \xrightarrow{\text{OPER } 1} 1$, the probability is $\frac{1}{4}$.
- $6 \xrightarrow{\text{OPER } 2} 3 \xrightarrow{\text{OPER } 2} 2 \xrightarrow{\text{OPER } 1} 1$, the probability is $\frac{1}{8}$.
- $6 \xrightarrow{\text{OPER } 2} 3 \xrightarrow{\text{OPER } 2} 2 \xrightarrow{\text{OPER } 2} 1$, the probability is $\frac{1}{8}$.

Thus, the expected number of operations is
$2 \cdot \frac{1}{4} + 3 \cdot \frac{1}{8} + 3 \cdot \frac{1}{8} + 2 \cdot \frac{1}{4} + 3 \cdot \frac{1}{8} + 3 \cdot \frac{1}{8} = \frac{5}{2} \equiv 500\,000\,006 \pmod{10^9 + 7}$.

,

D. Balancing

```
Input file:     standard input
Output file:    standard output
Time limit:     2 seconds
Memory limit:   512 megabytes
```

Ecrade has an integer array $a_1, a_2, \ldots, a_n$. It's guaranteed that for each $1 \le i < n$, $a_i \ne a_{i+1}$.

Ecrade can perform several operations on the array to make it strictly increasing.

In each operation, he can choose two integers l and r ($1 \le l \le r \le n$) and replace $a_l, a_{l+1}, \ldots, a_r$ with any $r - l + 1$ integers $a'_l, a'_{l+1}, \ldots, a'_r$ satisfying the following constraint:

- For each $l \le i < r$, the comparison between a'_i and a'_{i+1} is the same as that between a_i and a_{i+1}, i.e., if $a_i < a_{i+1}$, then $a'_i < a'_{i+1}$; otherwise, if $a_i > a_{i+1}$, then $a'_i > a'_{i+1}$; otherwise, if $a_i = a_{i+1}$, then $a'_i = a'_{i+1}$.

Ecrade wants to know the minimum number of operations to make the array strictly increasing. However, it seems a little difficult, so please help him!

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 10^4$). The description of the test cases follows.

The first line of each test case contains a single integer n ($2 \le n \le 2 \cdot 10^5$).

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($-10^9 \le a_i \le 10^9$).

It is guaranteed that for each $1 \le i < n$, $a_i \ne a_{i+1}$.

It is guaranteed that the sum of n across all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, output a single integer representing the minimum number of operations to make the array strictly increasing.

Standard Input	Standard Output
4 3 3 2 1 3 3 1 2 4 -2 -5 5 2 7 1 9 1 9 8 1 0	2 1 1 3

Note

In the first test case, a possible way to obtain the minimum number of operations:

- In the first operation, choose $l = 2, r = 2$ and $a'_2 = 4$, then $a = [3, 4, 1]$;

- In the second operation, choose $l = 1, r = 2$ and $a'_1 = -1, a'_2 = 0$, then $a = [-1, 0, 1]$.

In the second test case, a possible way to obtain the minimum number of operations:

- In the first operation, choose $l = 2, r = 3$ and $a'_2 = 4, a'_3 = 5$, then $a = [3, 4, 5]$.

In the third test case, a possible way to obtain the minimum number of operations:

- In the first operation, choose $l = 2, r = 3$ and $a'_2 = -1, a'_3 = 1$, then $a = [-2, -1, 1, 2]$.

,

E. Quaternary Matrix

Input file:	standard input
Output file:	standard output
Time limit:	2 seconds
Memory limit:	512 megabytes

A matrix is called quaternary if all its elements are 0, 1, 2, or 3.

Ecrade calls a quaternary matrix A *good* if the following two properties hold.

- The bitwise XOR of all numbers in each row of matrix A is equal to 0.
- The bitwise XOR of all numbers in each column of matrix A is equal to 0.

Ecrade has a quaternary matrix of size $n \times m$. He is interested in the minimum number of elements that need to be changed for the matrix to become *good*, and he also wants to find one of the possible resulting matrices.

However, it seems a little difficult, so please help him!

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 2 \cdot 10^5$). The description of the test cases follows.

The first line of each test case contains two integers n and m ($1 \le n, m \le 10^3$).

This is followed by n lines, each containing exactly m characters and consisting only of 0, 1, 2, and 3, describing the elements of Ecrade's matrix.

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

Output

For each test case, print the minimum number of elements that need to be changed for the matrix to become *good* on the first line, then print n lines, each containing exactly m characters and consisting only of 0, 1, 2, and 3, describing one of the possible resulting matrices.

If there are multiple possible resulting matrices, output any of them.

Standard Input	Standard Output
5 3 3 313 121 313 3 3 000 000 000 4 4 0123 1230 2301 3012	3 213 101 312 0 000 000 000 0 0123 1230 2301 3012 6

4 4 1232 2110 3122 1311 4 4 1232 2110 3122 1312	0132 2310 3131 1313 5 0132 2310 3120 1302

,

F. MST in Modulo Graph

```
Input file:     standard input
Output file:    standard output
Time limit:     3 seconds
Memory limit:   512 megabytes
```

You are given a complete graph with n vertices, where the i-th vertex has a weight p_i. The weight of the edge connecting vertex x and vertex y is equal to $\max(p_x, p_y) \bmod \min(p_x, p_y)$.

Find the smallest total weight of a set of $n-1$ edges that connect all n vertices in this graph.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 10^4$). The description of the test cases follows.

The first line of each test contains an integer n ($1 \le n \le 5 \cdot 10^5$).

The next line of each test contains n integers $p_1, p_2, \ldots, p_n$ ($1 \le p_i \le 5 \cdot 10^5$).

The sum of n over all test cases does not exceed $5 \cdot 10^5$.

The sum of $\max(p_1, p_2, \ldots, p_n)$ over all test cases does not exceed $5 \cdot 10^5$.

Output

For each test case, output a single integer — the weight of the minimum spanning tree.

Standard Input	Standard Output
4 5 4 3 3 4 4 10 2 10 3 2 9 9 4 6 4 6 12 33 56 48 41 89 73 99 150 55 100 111 130 7 11 45 14 19 19 8 10	1 0 44 10

Note

In the first test case, one of the possible ways to connect the edges is to draw the edges $(1,2)$, $(1,4)$, $(1,5)$, $(2,3)$. The weight of the first edge is $\max(p_1, p_2) \bmod \min(p_1, p_2) = 4 \bmod 3 = 1$, and the weights of all other edges are 0.