A. Omkar and Bad Story

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Omkar has received a message from Anton saying "Your story for problem A is confusing. Just make a formal statement." Because of this, Omkar gives you an array $a = [a_1, a_2, \ldots, a_n]$ of n distinct integers. An array $b = [b_1, b_2, \ldots, b_k]$ is called **nice** if for any two distinct elements b_i, b_j of b, $|b_i - b_j|$ appears in b at least once. In addition, all elements in b must be distinct. Can you add several (maybe, 0) integers to a to create a **nice** array b **of size at most** 300? If a is already **nice**, you don't have to add any elements.

For example, array $[3, 6, 9]$ is **nice**, as $|6 - 3| = |9 - 6| = 3$, which appears in the array, and $|9 - 3| = 6$, which appears in the array, while array $[4, 2, 0, 6, 9]$ is not **nice**, as $|9 - 4| = 5$ is not present in the array.

For integers x and y, $|x - y| = x - y$ if $x > y$ and $|x - y| = y - x$ otherwise.

Input

Each test contains multiple test cases. The first line contains t ($1 \le t \le 50$), the number of test cases. Description of the test cases follows.

The first line of each test case contains a single integer n ($2 \le n \le 100$) — the length of the array a.

The second line of each test case contains n **distinct** integers $a_1, a_2, \cdots, a_n$ ($-100 \le a_i \le 100$) — the elements of the array a.

Output

For each test case, output one line containing YES if Omkar can create a **nice** array b by adding elements to a and NO otherwise. The case of each letter does not matter, so yEs and nO will also be accepted.

If the first line is YES, output a second line containing a single integer k ($n \le k \le 300$).

Then output one line containing k **distinct** integers $b_1, b_2, \cdots, b_k$ ($-10^9 \le b_i \le 10^9$), the elements of the **nice** array b. $b_1, b_2, \cdots, b_k$ can be in any order. For each a_i in a, a_i must appear at least once in b.

It can be proved that if Omkar can create such an array b, then he can also do so in a way that satisfies the above constraints.

If multiple solutions exist, you can print any.

Example

input

```
4
3
3 0 9
2
3 4
5
-7 3 13 -2 8
4
4 8 12 6
```

output

```
yes
4
6 0 3 9
yEs
5
5 3 1 2 4
NO
Yes
6
8 12 6 2 4 10
```

Note

For the first case, you can add integers to a to receive the array $b = [6, 0, 3, 9]$. Note that $|6 - 3| = |9 - 6| = |3 - 0| = 3$ and 3 is in b, $|6 - 0| = |9 - 3| = 6$ and 6 is in b, and $|9 - 0| = 9$ is in b, so b is **nice**.

For the second case, you can add integers to a to receive the array $b = [5, 3, 1, 2, 4]$. We have that $|2 - 1| = |3 - 2| = |4 - 3| = |5 - 4| = 1$ is in b, $|3 - 1| = |4 - 2| = |5 - 3| = 2$ is in b, $|4 - 1| = |5 - 2| = 3$ is in b, and $|5 - 1| = 4$ is in b, so b is **nice**.

For the fourth case, you can add integers to a to receive the array $b = [8, 12, 6, 2, 4, 10]$. We have that $|4 - 2| = |6 - 4| = |8 - 6| = |10 - 8| = |12 - 10| = 2$ is in b, $|6 - 2| = |8 - 4| = |10 - 6| = |12 - 8| = 4$ is in b,

$|8-2|=|10-4|=|12-6|=6$ is in b, $|10-2|=|12-4|=8$ is in b, and $|12-2|=10$ is in b, so b is **nice**.

It can be proven that for all other test cases it is impossible to create a **nice** array b.

B. Prinzessin der Verurteilung

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

I, Fischl, Prinzessin der Verurteilung, descend upon this land by the call of fate an — Oh, you are also a traveler from another world? Very well, I grant you permission to travel with me.

It is no surprise Fischl speaks with a strange choice of words. However, this time, not even Oz, her raven friend, can interpret her expressions! Maybe you can help us understand what this young princess is saying?

You are given a string of n lowercase Latin letters, the word that Fischl just spoke. You think that the MEX of this string may help you find the meaning behind this message. The MEX of the string is defined as the shortest string that **doesn't** appear as a contiguous substring in the input. If multiple strings exist, the lexicographically smallest one is considered the MEX. Note that the empty substring does NOT count as a valid MEX.

A string a is lexicographically smaller than a string b if and only if one of the following holds:

- a is a prefix of b, but $a \neq b$;
- in the first position where a and b differ, the string a has a letter that appears earlier in the alphabet than the corresponding letter in b.

A string a is a substring of a string b if a can be obtained from b by deletion of several (possibly, zero or all) characters from the beginning and several (possibly, zero or all) characters from the end.

Find out what the MEX of the string is!

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \leq t \leq 1000$). Description of the test cases follows.

The first line of each test case contains an integer n ($1 \leq n \leq 1000$) — the length of the word. The second line for each test case contains a single string of n lowercase Latin letters.

The sum of n over all test cases will not exceed 1000.

Output

For each test case, output the MEX of the string on a new line.

Example

input

```
3
28
qaabzwsxedcrfvtgbyhnujmiklop
13
cleanairactbd
10
aannttoonn
```

output

```
ac
f
b
```

C. Diluc and Kaeya

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

The tycoon of a winery empire in Mondstadt, unmatched in every possible way. A thinker in the Knights of Favonius with an exotic appearance.

This time, the brothers are dealing with a strange piece of wood marked with their names. This plank of wood can be represented as a string of n characters. Each character is either a `'D'` or a `'K'`. You want to make some number of cuts (possibly 0) on this string, partitioning it into several contiguous pieces, each with length at least 1. Both brothers act with dignity, so they want to split the wood as evenly as possible. They want to know the maximum number of pieces you can split the wood into such that the ratios of the number of occurrences of `'D'` to the number of occurrences of `'K'` in each chunk are the same.

Kaeya, the curious thinker, is interested in the solution for multiple scenarios. He wants to know the answer for every **prefix** of the given string. Help him to solve this problem!

For a string we define a ratio as $a:b$ where 'D' appears in it a times, and 'K' appears b times. Note that a or b can equal 0, but not both. Ratios $a:b$ and $c:d$ are considered equal if and only if $a \cdot d = b \cdot c$.

For example, for the string 'DDD' the ratio will be $3:0$, for 'DKD' — $2:1$, for 'DKK' — $1:2$, and for 'KKKKDD' — $2:4$. Note that the ratios of the latter two strings are equal to each other, but they are not equal to the ratios of the first two strings.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 1000$). Description of the test cases follows.

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

The second line of each test case contains a string s of length n. Every character of s will be either 'D' or 'K'.

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

Output

For each test case, output n space separated integers. The i-th of these numbers should equal the answer for the prefix $s_1, s_2, \dots, s_i$.

Example

input

```
5
3
DDK
6
DDDDDD
4
DKDK
1
D
9
DKDKDDDDK
```

output

```
1 2 1
1 2 3 4 5 6
1 1 1 2
1
1 1 1 2 1 2 1 1 3
```

Note

For the first test case, there is no way to partition 'D' or 'DDK' into more than one block with equal ratios of numbers of 'D' and 'K', while you can split 'DD' into 'D' and 'D'.

For the second test case, you can split each prefix of length i into i blocks 'D'.

D. Omkar and Medians

time limit per test: 2 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

Uh oh! Ray lost his array yet again! However, Omkar might be able to help because he thinks he has found the *OmkArray* of Ray's array. The *OmkArray* of an array a with elements $a_1, a_2, \dots, a_{2k-1}$, is the array b with elements $b_1, b_2, \dots, b_k$ such that b_i is equal to the median of $a_1, a_2, \dots, a_{2i-1}$ for all i. Omkar has found an array b of size n ($1 \le n \le 2 \cdot 10^5$, $-10^9 \le b_i \le 10^9$). Given this array b, Ray wants to test Omkar's claim and see if b actually is an *OmkArray* of some array a. Can you help Ray?

The median of a set of numbers $a_1, a_2, \dots, a_{2i-1}$ is the number c_i where $c_1, c_2, \dots, c_{2i-1}$ represents $a_1, a_2, \dots, a_{2i-1}$ sorted in nondecreasing order.

Input

Each test contains multiple test cases. The first line contains a single integer t ($1 \le t \le 10^4$) — the number of test cases. Description of the test cases follows.

The first line of each test case contains an integer n ($1 \le n \le 2 \cdot 10^5$) — the length of the array b.

The second line contains n integers $b_1, b_2, \dots, b_n$ ($-10^9 \le b_i \le 10^9$) — the elements of b.

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

Output

For each test case, output one line containing YES if there exists an array a such that b_i is the median of $a_1, a_2, \dots, a_{2i-1}$ for all i, and NO otherwise. The case of letters in YES and NO do not matter (so yEs and No will also be accepted).

Examples

input

```
5
4
6 2 1 3
1
4
5
4 -8 5 6 -7
2
3 3
4
2 1 2 3
```

output

```
NO
YES
NO
YES
YES
```

input

```
5
8
-8 2 -6 -5 -4 3 3 2
7
1 1 3 1 0 -2 -1
7
6 12 8 6 2 6 10
6
5 1 2 3 6 7
5
1 3 4 3 0
```

output

```
NO
YES
NO
NO
NO
```

Note

In the second case of the first sample, the array $[4]$ will generate an OmkArray of $[4]$, as the median of the first element is 4.

In the fourth case of the first sample, the array $[3, 2, 5]$ will generate an OmkArray of $[3, 3]$, as the median of 3 is 3 and the median of $2, 3, 5$ is 3.

In the fifth case of the first sample, the array $[2, 1, 0, 3, 4, 4, 3]$ will generate an OmkArray of $[2, 1, 2, 3]$ as

- the median of 2 is 2
- the median of $0, 1, 2$ is 1
- the median of $0, 1, 2, 3, 4$ is 2
- and the median of $0, 1, 2, 3, 3, 4, 4$ is 3.

In the second case of the second sample, the array $[1, 0, 4, 3, 5, -2, -2, -2, -4, -3, -4, -1, 5]$ will generate an OmkArray of $[1, 1, 3, 1, 0, -2, -1]$, as

- the median of 1 is 1
- the median of $0, 1, 4$ is 1
- the median of $0, 1, 3, 4, 5$ is 3
- the median of $-2, -2, 0, 1, 3, 4, 5$ is 1
- the median of $-4, -2, -2, -2, 0, 1, 3, 4, 5$ is 0
- the median of $-4, -4, -3, -2, -2, -2, 0, 1, 3, 4, 5$ is -2
- and the median of $-4, -4, -3, -2, -2, -2, -1, 0, 1, 3, 4, 5, 5$ is -1

For all cases where the answer is `NO`, it can be proven that it is impossible to find an array a such that b is the OmkArray of a.

E. Omkar and Forest

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Omkar's most recent follower, Ajit, has entered the Holy Forest. Ajit realizes that Omkar's forest is an n by m grid ($1 \le n, m \le 2000$) of some non-negative integers. Since the forest is blessed by Omkar, it satisfies some special conditions:

1. For any two adjacent (sharing a side) cells, the absolute value of the difference of numbers in them is at most 1.
2. If the number in some cell is strictly larger than 0, it should be strictly greater than the number in **at least one** of the cells adjacent to it.

Unfortunately, Ajit is not fully worthy of Omkar's powers yet. He sees each cell as a "0" or a "#". If a cell is labeled as "0", then the number in it must equal 0. Otherwise, the number in it can be any nonnegative integer.

Determine how many different assignments of elements exist such that these special conditions are satisfied. Two assignments are considered different if there exists at least one cell such that the numbers written in it in these assignments are different. Since the answer may be enormous, find the answer modulo 10^9+7.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 100$). Description of the test cases follows.

The first line of each test case contains two integers n and m ($1 \le n, m \le 2000, nm \ge 2$) – the dimensions of the forest.

n lines follow, each consisting of one string of m characters. Each of these characters is either a "0" or a "#".

It is guaranteed that the sum of n over all test cases does not exceed 2000 and the sum of m over all test cases does not exceed 2000.

Output

For each test case, print one integer: the number of valid configurations modulo 10^9+7.

Example

input

```
4
3 4
0000
00#0
0000
2 1
#
#
1 2
##
6 29
#############################
#000##0###0##0#0####0####000#
#0#0##00#00##00####0#0###0#0#
#0#0##0#0#0##00###00000##00##
#000##0###0##0#0##0###0##0#0#
#############################
```

output

```
2
3
3
319908071
```

Note

For the first test case, the two valid assignments are

$$\begin{matrix}0000\\0000\\0000\end{matrix}$$

and

$$\begin{matrix}0000\\0010\\0000\end{matrix}$$

F. Omkar and Akmar

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Omkar and Akmar are playing a game on a circular board with n ($2 \le n \le 10^6$) cells. The cells are numbered from 1 to n so that for each i ($1 \le i \le n-1$) cell i is adjacent to cell $i+1$ and cell 1 is adjacent to cell n. Initially, each cell is empty.

Omkar and Akmar take turns placing either an A or a B on the board, with Akmar going first. The letter must be placed on an empty cell. In addition, the letter cannot be placed adjacent to a cell containing the same letter.

A player loses when it is their turn and there are no more valid moves.

Output the number of possible distinct games where both players play optimally modulo 10^9+7. Note that we only consider games where some player has lost and there are no more valid moves.

Two games are considered distinct if the number of turns is different or for some turn, the letter or cell number that the letter is placed on were different.

A move is considered optimal if the move maximizes the player's chance of winning, assuming the other player plays optimally as well. More formally, if the player who has to move has a winning strategy, they have to make a move after which they will still have a winning strategy. If they do not, they can make any move.

Input

The only line will contain an integer n ($2 \le n \le 10^6$) — the number of cells on the board.

Output

Output a single integer — the number of possible distinct games where both players play optimally modulo $10^9 + 7$.

Examples

input
```
2
```
output
```
4
```

input
```
69420
```
output
```
629909355
```

input
```
42069
```
output
```
675837193
```

Note

For the first sample case, the first player has 4 possible moves. No matter what the first player plays, the second player only has 1 possible move, so there are 4 possible games.