A. Angry Students

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

It's a walking tour day in SIS.Winter, so t groups of students are visiting Torzhok. Streets of Torzhok are so narrow that students have to go in a row one after another.

Initially, some students are angry. Let's describe a group of students by a string of capital letters "A" and "P":

- "A" corresponds to an angry student
- "P" corresponds to a patient student

Such string describes the row from the last to the first student.

Every minute every angry student throws a snowball at the next student. Formally, if an angry student corresponds to the character with index i in the string describing a group then they will throw a snowball at the student that corresponds to the character with index $i+1$ (students are given from the last to the first student). If the target student was not angry yet, they become angry. Even if the first (the rightmost in the string) student is angry, they don't throw a snowball since there is no one in front of them.

Let's look at the first example test. The row initially looks like this: PPAP. Then, after a minute the only single angry student will throw a snowball at the student in front of them, and they also become angry: PPAA. After that, no more students will become angry.

Your task is to help SIS.Winter teachers to determine the last moment a student becomes angry for every group.

Input

The first line contains a single integer t — the number of groups of students ($1 \le t \le 100$). The following $2t$ lines contain descriptions of groups of students.

The description of the group starts with an integer k_i ($1 \le k_i \le 100$) — the number of students in the group, followed by a string s_i, consisting of k_i letters "A" and "P", which describes the i-th group of students.

Output

For every group output single integer — the last moment a student becomes angry.

Examples

input

```
1
4
PPAP
```

output

```
1
```

input

```
3
12
APPAPPPAPPPP
3
AAP
3
PPA
```

output

```
4
1
0
```

Note

In the first test, after 1 minute the state of students becomes PPAA. After that, no new angry students will appear.

In the second tets, state of students in the first group is:

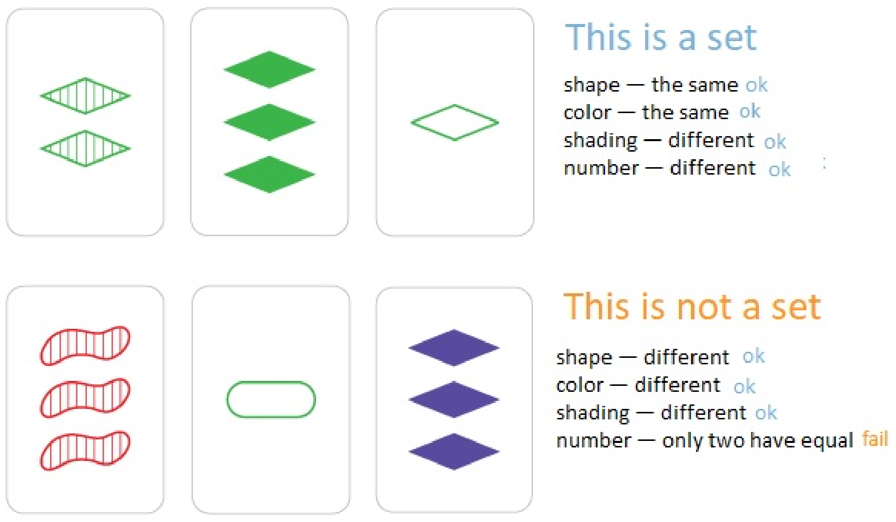
- after 1 minute — AAPAAPPAAPPP
- after 2 minutes — AAAAAAPAAAPP
- after 3 minutes — AAAAAAAAAAAP
- after 4 minutes all 12 students are angry

In the second group after 1 minute, all students are angry.

B. Hyperset

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

Bees Alice and Alesya gave beekeeper Polina famous card game "Set" as a Christmas present. The deck consists of cards that vary in four features across three options for each kind of feature: number of shapes, shape, shading, and color. In this game, some combinations of three cards are said to make up a *set*. For every feature — color, number, shape, and shading — the three cards must display that feature as either all the same, or pairwise different. The picture below shows how sets look.



Polina came up with a new game called "Hyperset". In her game, there are n cards with k features, each feature has three possible values: "S", "E", or "T". The original "Set" game can be viewed as "Hyperset" with $k = 4$.

Similarly to the original game, three cards form a *set*, if all features are the same for all cards or are pairwise different. The goal of the game is to compute the number of ways to choose three cards that form a *set*.

Unfortunately, winter holidays have come to an end, and it's time for Polina to go to school. Help Polina find the number of sets among the cards lying on the table.

Input

The first line of each test contains two integers n and k ($1 \le n \le 1500$, $1 \le k \le 30$) — number of cards and number of features.

Each of the following n lines contains a card description: a string consisting of k letters "S", "E", "T". The i-th character of this string decribes the i-th feature of that card. All cards are distinct.

Output

Output a single integer — the number of ways to choose three cards that form a set.

Examples

input

```
3 3
SET
ETS
TSE
```

output

```
1
```

input

```
3 4
SETE
ETSE
TSES
```

output

```
0
```

input

```
5 4
SETT
TEST
EEET
ESTE
STES
```

output

```
2
```

Note

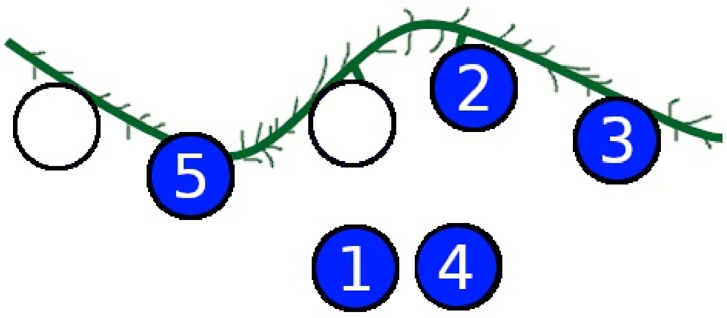
In the third example test, these two triples of cards are *sets*:

1. "SETT", "TEST", "EEET"
2. "TEST", "ESTE", "STES"

C. Garland

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Vadim loves decorating the Christmas tree, so he got a beautiful garland as a present. It consists of n light bulbs in a single row. Each bulb has a number from 1 to n (in arbitrary order), such that all the numbers are distinct. While Vadim was solving problems, his home Carp removed some light bulbs from the garland. Now Vadim wants to put them back on.



Vadim wants to put all bulb back on the garland. Vadim defines *complexity* of a garland to be the number of pairs of adjacent bulbs with numbers with different parity (remainder of the division by 2). For example, the complexity of $1\ 4\ 2\ 3\ 5$ is 2 and the complexity of $1\ 3\ 5\ 7\ 6\ 4\ 2$ is 1.

No one likes complexity, so Vadim wants to minimize the number of such pairs. Find the way to put all bulbs back on the garland, such that the complexity is as small as possible.

Input

The first line contains a single integer n $(1 \le n \le 100)$ — the number of light bulbs on the garland.

The second line contains n integers $p_1,\ p_2,\ \ldots,\ p_n$ $(0 \le p_i \le n)$ — the number on the i-th bulb, or 0 if it was removed.

Output

Output a single number — the minimum *complexity* of the garland.

Examples

input

```
5
0 5 0 2 3
```

output

```
2
```

input

```
7
1 0 0 5 0 0 2
```

output

```
1
```

Note

In the first example, one should place light bulbs as 1 5 4 2 3. In that case, the complexity would be equal to 2, because only $(5, 4)$ and $(2, 3)$ are the pairs of adjacent bulbs that have different parity.

In the second case, one of the correct answers is 1 7 3 5 6 4 2.

D. Numbers on Tree

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Evlampiy was gifted a rooted tree. The vertices of the tree are numbered from 1 to n. Each of its vertices also has an integer a_i written on it. For each vertex i, Evlampiy calculated c_i — the number of vertices j in the subtree of vertex i, such that $a_j < a_i$.

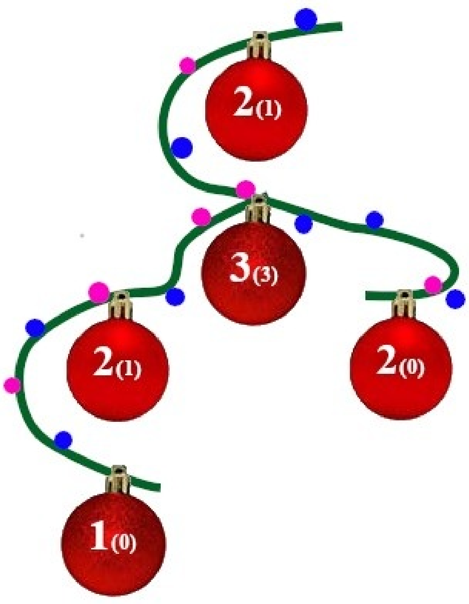


Illustration for the second example, the first integer is a_i and the integer in parentheses is c_i

After the new year, Evlampiy could not remember what his gift was! He remembers the tree and the values of c_i, but he completely forgot which integers a_i were written on the vertices.

Help him to restore initial integers!

Input

The first line contains an integer n $(1 \le n \le 2000)$ — the number of vertices in the tree.

The next n lines contain descriptions of vertices: the i-th line contains two integers p_i and c_i ($0 \le p_i \le n$; $0 \le c_i \le n-1$), where p_i is the parent of vertex i or 0 if vertex i is root, and c_i is the number of vertices j in the subtree of vertex i, such that $a_j < a_i$.

It is guaranteed that the values of p_i describe a rooted tree with n vertices.

Output

If a solution exists, in the first line print "YES", and in the second line output n integers a_i $(1 \le a_i \le 10^9)$. If there are several solutions, output any of them. One can prove that if there is a solution, then there is also a solution in which all a_i are between 1 and 10^9.

If there are no solutions, print "NO".

Examples

input

```
3
2 0
0 2
2 0
```

output

```
YES
1 2 1
```

input

```
5
0 1
1 3
2 1
```

```
3 0
2 0
```

output

```
YES
2 3 2 1 2
```

E1. Madhouse (Easy version)

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

This problem is different with hard version only by constraints on total answers length

It is an interactive problem

Venya joined a tour to the madhouse, in which orderlies play with patients the following game. Orderlies pick a string s of length n, consisting only of lowercase English letters. The player can ask two types of queries:

- `? l r` – ask to list all substrings of $s[l..r]$. Substrings will be returned in random order, and in every substring, all characters will be randomly shuffled.
- `! s` – guess the string picked by the orderlies. This query can be asked exactly once, after that the game will finish. If the string is guessed correctly, the player wins, otherwise he loses.

The player can ask **no more than 3 queries** of the first type.

To make it easier for the orderlies, there is an additional limitation: the total number of returned substrings in all queries of the first type must not exceed $(n+1)^2$.

Venya asked you to write a program, which will guess the string by interacting with the orderlies' program and acting by the game's rules.

Your program should immediately terminate after guessing the string using a query of the second type. In case your program guessed the string incorrectly, or it violated the game rules, it will receive verdict `Wrong answer`.

Note that in every test case the string is fixed beforehand and will not change during the game, which means that the interactor is not adaptive.

Input

First line contains number n ($1 \le n \le 100$) — the length of the picked string.

Interaction

You start the interaction by reading the number n.

To ask a query about a substring from l to r inclusively ($1 \le l \le r \le n$), you should output

```
? l r
```

on a separate line. After this, all substrings of $s[l..r]$ will be returned in random order, each substring exactly once. In every returned substring all characters will be randomly shuffled.

In the case, if you ask an incorrect query, ask more than 3 queries of the first type or there will be more than $(n+1)^2$ substrings returned in total, you will receive verdict **Wrong answer**.

To guess the string s, you should output

```
! s
```

on a separate line.

After printing each query, do not forget to flush the output. Otherwise, you will get `Idleness limit exceeded`. To flush the output, you can use:

- fflush(stdout) or cout.flush() in C++;
- System.out.flush() in Java;
- flush(output) in Pascal;
- stdout.flush() in Python;
- see documentation for other languages.

If you received `-` (dash) as an answer to any query, you need to terminate your program with exit code 0 (for example, by calling `exit(0)`). This means that there was an error in the interaction protocol. If you don't terminate with exit code 0, you can receive any unsuccessful verdict.

Hack format

To hack a solution, use the following format:

The first line should contain one integer n ($1 \le n \le 100$) — the length of the string, and the following line should contain the string s.

Example

input

```
4

a
aa
a

cb
b
c

c
```

output

```
? 1 2

? 3 4

? 4 4

! aabc
```

E2. Madhouse (Hard version)

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

This problem is different with easy version only by constraints on total answers length

It is an interactive problem

Venya joined a tour to the madhouse, in which orderlies play with patients the following game. Orderlies pick a string s of length n, consisting only of lowercase English letters. The player can ask two types of queries:

- `? l r` – ask to list all substrings of $s[l..r]$. Substrings will be returned in random order, and in every substring, all characters will be randomly shuffled.
- `! s` – guess the string picked by the orderlies. This query can be asked exactly once, after that the game will finish. If the string is guessed correctly, the player wins, otherwise he loses.

The player can ask **no more than 3 queries** of the first type.

To make it easier for the orderlies, there is an additional limitation: the total number of returned substrings in all queries of the first type must not exceed $\lceil 0.777(n+1)^2 \rceil$ ($\lceil x \rceil$ is x rounded up).

Venya asked you to write a program, which will guess the string by interacting with the orderlies' program and acting by the game's rules.

Your program should immediately terminate after guessing the string using a query of the second type. In case your program guessed the string incorrectly, or it violated the game rules, it will receive verdict `Wrong answer`.

Note that in every test case the string is fixed beforehand and will not change during the game, which means that the interactor is not adaptive.

Input

First line contains number n ($1 \le n \le 100$) — the length of the picked string.

Interaction

You start the interaction by reading the number n.

To ask a query about a substring from l to r inclusively ($1 \le l \le r \le n$), you should output

```
? l r
```

on a separate line. After this, all substrings of $s[l..r]$ will be returned in random order, each substring exactly once. In every returned substring all characters will be randomly shuffled.

In the case, if you ask an incorrect query, ask more than 3 queries of the first type or there will be more than $\lceil 0.777(n+1)^2 \rceil$ substrings returned in total, you will receive verdict **Wrong answer**.

To guess the string s, you should output

```
! s
```

on a separate line.

After printing each query, do not forget to flush the output. Otherwise, you will get `Idleness limit exceeded`. To flush the output, you can use:

- fflush(stdout) or cout.flush() in C++;
- System.out.flush() in Java;
- flush(output) in Pascal;
- stdout.flush() in Python;
- see documentation for other languages.

If you received `-` (dash) as an answer to any query, you need to terminate your program with exit code 0 (for example, by calling `exit(0)`). This means that there was an error in the interaction protocol. If you don't terminate with exit code 0, you can receive any unsuccessful verdict.

Hack format

To hack a solution, use the following format:

The first line should contain one integer n ($1 \le n \le 100$) — the length of the string, and the following line should contain the string s.

Example

input

```
4

a
aa
a

cb
b
c

c
```

output

```
? 1 2

? 3 4

? 4 4

! aabc
```

F. LCC

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

An infinitely long Line Chillland Collider (LCC) was built in Chillland. There are n pipes with coordinates x_i that are connected to LCC. When the experiment starts at time 0, i-th proton flies from the i-th pipe with speed v_i. It flies to the right with probability p_i and flies to the left with probability $(1 - p_i)$. The *duration of the experiment* is determined as the time of the first collision of any two protons. In case there is no collision, the duration of the experiment is considered to be zero.

Find the expected value of the duration of the experiment.

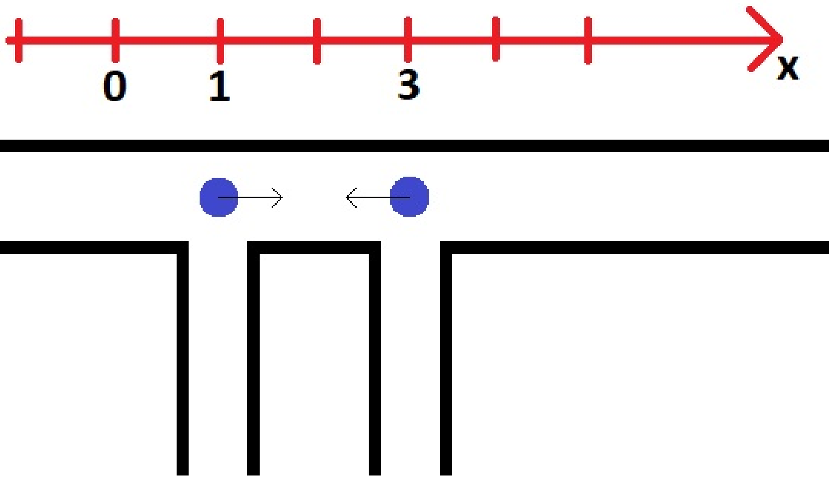


Illustration for the first example

Input

The first line of input contains one integer n — the number of pipes ($1 \le n \le 10^5$). Each of the following n lines contains three integers x_i, v_i, p_i — the coordinate of the i-th pipe, the speed of the i-th proton and the probability that the i-th proton flies to the right in percentage points ($-10^9 \le x_i \le 10^9, 1 \le v \le 10^6, 0 \le p_i \le 100$). It is guaranteed that all x_i are distinct and sorted in increasing order.

Output

It's possible to prove that the answer can always be represented as a fraction P/Q, where P is an integer and Q is a natural number not divisible by $998\,244\,353$. In this case, print $P \cdot Q^{-1}$ modulo $998\,244\,353$.

Examples

input

```
2
1 1 100
3 1 0
```

output

```
1
```

input

```
3
7 10 0
9 4 86
14 5 100
```

output

```
0
```

input

```
4
6 4 50
11 25 50
13 16 50
15 8 50
```

output

```
150902884
```