Problem A : Stolen necklace

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

Nami, recently acquired a priceless necklace. Eager to share her new treasure,She invited her **n** friends to The Going Merry (the straw hats' pirate ship) , where she proudly displayed the necklace.
The friends, **one by one**, entered and exited the room. At no point were there more than one friend in the room. In other words, the first friend entered and left first, then the second, and so on.
At the beginning (before any friend visited), **the necklace was in the room**.However, by the end (after the n-th friend), **the map mysteriously vanished**.The exact moment of disappearance remains shrouded in uncertainty.
Determined to identify the culprit, Nami questioned her friends one by one in the same order they visited the room. Each friend was asked if the map was still there when they entered. Each friend answered using one of three answers:

- No (represented with **0**)
- Yes (represented with **1**)
- Can't remember (represented with **?**)

Everyone except the thief either couldn't remember or **told the truth**. The thief, however, **can say anything** (any of the three options).
Nami cannot find who the thief is. She asks you to find out **the number of potential suspects** according to the answers of her friends.

Input :

The first line contains a single integer **T** $(1 \leq T \leq 10^4)$ — the number of test cases. Then the test cases follow. Each test case consists of two lines.

For each test case:
The first line contains a single integer **n** $(1 \leq n \leq 2 \cdot 10^5)$ — the length of the string s that describes the friends answers.

The second line s describes the friends answers in the order they visited the map, where each character s_i in the string is either 0 or 1 or ? indicates the answer of the i-th friend.

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

Output :

For each test case, output a positive integer (strictly greater than zero) representing the number of friends who could have stolen the necklace based on the provided responses (those whom Nami could suspect of stealing the necklace).

Example:

Input :

```
3
7
1110000
8
1?1??0?0
4
??11
```

Output :

```
2
4
1
```

Note:

In the first case, the suspects are the third and fourth friends (one-indexed).It can be shown that no one else could be the thief.

Problem B : Luffy is a Foodie

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

Monkey D. Luffy, the main character of “One Piece,” is known for his love of food, particularly meat.

You will be given a string consisting of the characters ‘**.**’, ‘**#**’ and ‘**|**’:
‘**.**’ is an empty spot, ‘**#**’ is a piece of meat and ‘**|**’ is a wall. This string represents a **1D** grid.

Luffy needs to choose a starting point and a walking direction(left or right) such that he maximizes the meat eaten.

So he asked for your help.

Your goal is to put Luffy in one of the empty spots in order to maximize the meat eaten such that:

- Luffy can move in only 1 direction in the grid(left or right), in other words he can’t change his direction after choosing it.
- Luffy will stop if he finds a wall or the edge of the grid.

It is guaranteed that the grid contains at least one empty spot.

Input :

The first line of input contains a single integer **T** ($1 \le T \le 100$) — the number of test cases. Then the test cases follow. Each test case consists of two lines.

The first line of each test case contains an integer **n** ($1 \le n \le 10^6$) — the length of the string

The second line of each test case contains a string **s** of length **n**, consisting of characters ‘**#**’, ‘**.**’, ‘**|**’.

It is guaranteed that the sum of **n** over all test cases doesn’t exceed 10^6.

Output :

For each test case, output the maximum quantity of meat Luffy can get.

Example:

Input :

```
2
8
##.#|#.#
12
|.#|#.#.#..#
```

Output :

```
2
3
```

Note:

In the **first** testcase, the first empty spot is picked (position with index 3), the direction that maximizes the answer is left.
In the **second** testcase, there are 2 possible empty spots, the 4th empty spot (position with index 10) and the 5th empty spot (position with index 11), the direction that maximizes the answer is left.

Problem C: Currensea

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

The world of one piece is vast and contains numerous mystical islands. One such island is “Currensea”: an island where its only currency are coins with values (11, 111, 1111, ...).

Upon arrival on this island, Nami wants to sell her new jewelry she acquired in Skypeia but doesn’t know how to price her goods.

She wants to know, if it’s possible to price her jewelry to some integer **n** so that it can be bought using this island’s unique currency?

Note that you can use any coin any number of times to buy something.

Input :

The first line contains a single integer **T** $(1 \leq T \leq 1000)$ — the number of test cases. Then the test cases follow.

Each test case consists of one line. the integer **n** $(0 \leq n \leq 10^9)$

Output :

For each testcase, you should output a single string. If you can buy n using the island’s currency, output “YES” (without quotes). Otherwise, output “NO”.

Example:

Input :

```
3

55

1122

150
```

Output :

```
Yes

Yes

No
```

Note :

- 55 = 5*11
- 1122 = 11 + 1111
- It can be proved that 150 can’t be written as the sum of $11a + 111b + 1111c + 11111d + ...$

Problem D : Celebration

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

The Straw Hat Pirates, composed of **k** members **including their captain** , are eagerly anticipating a festive celebration, and their chosen delight for the occasion is pies. Each pie is segmented into **x** pieces. However, their captain, Monkey D. Luffy, has a formidable appetite and will eat up to **y** pieces all by himself.

The Straw Hat Pirates are not really good with math, so they ask for your help to decide the minimum number of pies they need to buy to satisfy all the crew members' appetites.

Input

The first line contains a single integer **T** $(1 \leq T \leq 1000)$ — the number of test cases. Then the test cases follow.

Each testcase consists of a single line which contains three space separated integers: x $(1 \leq x \leq 10^4)$ – the total number of pieces in each pie, y $(0 \leq y \leq 10^4)$ – the number of pieces Luffy intends to eat alone and k $(1 \leq k \leq 10^4)$ – the total number of crew members.

Output

Output a single integer – the minimum number of pies the Straw Hat Pirates need to order.

Example

Input:

```
2
8 3 10
4 1 2
```

Ouput:

```
2
1
```

Problem E: Bounty Challenge

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

Trafalgar Law, captain of the heart pirates, is facing a bit of a problem. In front of him there exists **n** strong enemies that he needs to fight. If the value of his current bounty is strictly greater than the strength of his enemy then he can win the fight, otherwise he will choose to run away.
The problem is that, with every fight he wins, his bounty will increase by a 100 and if his bounty exceeds the value **L** then Kizaru, one of the strongest fighters in the marine will be dispatched to get rid of him.

Given the number of fights n, the maximum allowable bounty L and an array of integers *fights* representing the strengh of the opponents, the procedure for simulating the fights is as follows:

1. Start with an initial bounty B

2. For each fight; if the current bounty is strictly greater than the strength of the opponent $fights_i$ $(1<=i<=n)$, add 100 to the current bounty.

Your objective is to find the maximum initial bounty B such that after all the fights, the final bounty is less than or equal to L.

Input :

The first line of the input contains integers n and L $(1<=n<=10^5, 0<=L<=10^{18})$, the number of fights, and the maximum allowable bounty respectively.

The second line of the input contains n space-separated integers $fights_i$, the strength of the *i-th* opponent. $(1<=fights_i <= 10^9)$

Output :

Print a single integer, the maximum initial bounty B such that after all the fights, the final bounty is less than or equal to L.

Example:

Input :

```
5 700
300 200 500 100 700
```

Output :

```
300
```

Input :

```
4 500
700 200 300 100
```

Output :

```
200
```

Note:

First example:

For initial bounty $B = 300$, the bounty will change as follows :

i = 1: 300 <= 300, Trafalgar will run away.

i = 2: 300 > 200, Trafalgar will win and bounty will become 400

i = 3: 400 <= 500, Trafalgar will run away.

i = 4: 400 > 100, Trafalgar will win and bounty will become 500

i = 5: 500 <= 700, Trafalgar will run away.

After all the fights, the final bounty is less than or equal to L

If $B = 301$, we will have:

i = 1: 301 > 300, Trafalgar will win and bounty will become 401.

i = 2: 401 > 200, Trafalgar will win and bounty will become 501.

i = 3: 501 > 500, Trafalgar will win and bounty will become 601.

i = 4: 601 > 100, Trafalgar will win and bounty will become 701.

i = 5: 701 > 700, Trafalgar will win and bounty will become 801.

After all the fights, the final bounty will be greater than L.

$B = 300$ is then the maximum initial bounty.

Problem F: Blood Transfusion(easy version)

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

Chopper is the doctor of the Straw Hat pirates. After the big war at the country of wano, there are **n** injured people that need a blood transfusion.
Chopper is determined to save everyone, so he rounds up **n** donors.
Each person has a specific blood type A, B, AB or O.
There are certain rules to the blood transfusion operations:

- A donor can only donate to one injured person
- Blood type A donors can only donate to recipients with blood types A and AB.
- Blood type B donors can only donate to recipients with blood types B and AB.
- Blood type AB donors can only donate to recipients with blood type AB only.
- Blood type O donors can donate to recipients with blood types A, B, AB and O.

Help chopper figure out if he can save every injured person using the donors he picked.

Input :

The first line contains T $(1 \leq T \leq 100)$, the number of testaces.
The first line of each testcase contains n $(1 \leq n \leq 2 * 10^6)$ the number of injured people
The second line of each testcase containe **n** space separated strings denoting the blood types of the injured people
The third line of each testcase contains **n** space separated strings denoting the blood types of the donors
It is guaranteed that the sum of n over all the testcases does not exceed $2 * 10^6$

Output :

For each query, output **"YES"**(without quotes) if the injured people can be saved and **"NO"**(without quotes) otherwise.

Example:

Input :

```
3
2
A A
A B
5
A B AB O A
O O O O O
5
A A A B AB
O O B B B
```

Output :

```
NO
YES
NO
```

Note:

In the first test case, we can only save one person of blood type A.
In the second testcase, every person can receive a donation from an O donor
In the third testcase, we can only save two people of blood type A.

Problem G: Blood Transfusion (hard version)

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

Chopper is the doctor of the Straw Hat pirates. After the big war at the country of wano, there are **n** injured people that need a blood transfusion.

Chopper is determined to save everyone, so he rounds up **n** donors.

However, in the vast world of One Piece, there are many races and more blood types than those of the human world. Also, the rules of compatibility between blood types are different.

There are certain rules to the blood transfusion operations:

- A donor can only donate to one injured person.
- A donor can donate blood to a person of the same blood type
- The blood types are denoted by uppercase english letters and the relations between them are denoted by **m** couples **(x,y)** $(x \neq y)$ which represent the fact that a person with blood type **x** can donate to a person of blood type **y**

Help chopper figure out if he can save every injured person using the donors he picked.

Input :

The first line contains two space separated integers n $(1 \leq n \leq 100)$ and m $(0 \leq m \leq 650)$ the number of injured people and the number of relations between the different blood types
The second line of each testcase contains **n** space separated strings denoting the blood types of the injured people
The third line of each testcase contains **n** space separated strings denoting the blood types of the donors
The next **m** lines each contains two distinct, space separated uppercase english characters denoting the relations between the blood types.

Output :

For each query, output **"YES"** (without quotes) if the injured people can be saved and **"NO"** (without quotes) otherwise.

Example:

Input :

```
5 4
A B C Z A
A I O C O
O B
O Z
I A
Y X
```

Output :

```
YES
```

Note:

One possible solution in the testcase: A donates to A, I donates to A, O donates to B, O donates to Z, C donates to C

Problem H : Clone Fruit

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

Kaido, one of the four emperors of the sea, would like to build a huge army in order to obtain the one piece.
He first acquired a huge number of Clone Fruits: These fruits, like the name indicates, have the unique ability to clone a person
Then, he chose **m** of his best crewmates and started cloning them as follows:

- At first the crewmates stand in a queue
- Each time a crewmate is cloned, the two resulting persons rejoin the end of the queue

Kaido would like to keep track of the person that ate the **n-th** fruit. Help him figure this out.

Input :

The first line contains a single integer**n** $(1 \leq n \leq 10^9)$ representing the mentioned **n**.
The second line contains a single integer **m** $(1 \leq m \leq 1000)$ indicating the initial number of people in the queue.
The next **m** lines contains the names of the m individuals with the i-th line representing the name of the i-th person
It is guaranteed that the length of the names does not exceed 10

Output :

For each test case, output a single line — **the name of the person who consumes the n-th cloning fruit**. The fruits are numbered starting from 1.

Example:

Input :

```
3
5
King
Queen
Jack
Ulti
Sasaki
```

Output :

```
Jack
```

Input :

```
6
3
Mohamed
Aziz
Mansour
```

Output :

```
Aziz
```

Note :

In the second test case, we want to keep track of the 6th fruit. Initially, the queue has 3 people, each one of them is cloned. The queue becomes Mohamed Mohamed Aziz Aziz Mansour Mansour, so the 6th fruit is eaten by Aziz.

Problem I : Hidden Treasures

time limit per test: 6 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Nami is the navigator of the Straw Hat Pirates in “One Piece”, and one of her primary skills is cartography.
She has recently drawn a map for her team, this map is a matrix of dimensions **n×m**, where **n** and **m** denote the number of rows and columns respectively, composed of integers. This matrix serves as a guide to predict positions of hidden treasures.
The diagonals of a matrix are defined as follows:

- **Main diagonal**: diagonal starting from element (1, 1) and passing through elements (d, d) for 1≤d≤min(n, m).
- **Upper diagonals**: diagonals starting from elements (1, 1+d) and passing through elements (k, k+d) for 1≤k≤n, 1≤d<m, k+d<m.
- **Lower diagonals** : diagonals starting from elements (1+d, 1) and passing through elements (k+d, k) for 1≤k≤m, 1≤d<n, k+d≤n.

Each element is then part of only one diagonal.
The diagram below illustrates the diagonals of a **3×4** matrix:

1	2	3	4
5	6	7	8
9	10	11	12

Elements 1, 6 and 11 make a diagonal, Element 4 makes a diagonal by itself too...
An element of position **(x, y)** contains a treasure if and only if it divides the diagonal into two parts with equal sum. In other words, the sum the elements above our element in the diagonal is equal to the sum of elements under our element in the diagonal.
More formally $\mathbf{\Sigma_{elements(i,\ j)\in diagonal\ containing(x,\ y)\ |\ i<x,\ j<y}\ M[i][j] = \Sigma_{elements(i',\ j')\in diagonal\ containing(x,\ y)\ |\ x<i',\ y<j'}\ M[i'][j']}$.
You will be given the matrix, and q queries, for each query you will be given a position **(x, y)**
Output **“YES”** if that position contains a treasure, and **“NO”** otherwise.

Input :

The first line contains 3 integers **n**, **m** $(2 \leq n, m \leq 2.10^3)$, number of rows and number of columns of our matrix respectively and **q** queries $(1 \leq q \leq 10^6)$.
The next n lines each contain **m** space separated integers describing rows of the matrix. Each integer of the matrix a_{ij} $(-10^9 \leq a_{ij} \leq 10^9)$
The next q lines each contain 2 integers **x** and **y** (1≤x≤n, 1≤y≤m), a specific row and column of our matrix respectively.

Output :

For each query, output **“YES”**(without quotes) if the position **(x, y)** can contain a treasure, and **“NO”**(without quotes) otherwise.

Example:

Input :

```
4 4 4
9 4 7 8
6 5 0 3
11 4 2 5
10 3 8 14
1 4
3 3
3 2
4 2
```

Output :

```
YES
YES
NO
NO
```

Note:

9	4	7	8
6	5	0	3
11	4	2	5
10	3	8	14

In the **first** query the answer is YES because both sum are equal to 0 (there are no elements in the upper or lower part of the diagonal).
In the **second** query the answer is YES because: 9 + 5 = 14. (m[1][1] + m[2][2] = 14 and m[4][4] = 14)
In the **third** query the answer is NO because: m[2][1] = 6 $\neq$ m[4][3] = 8.
In the **fourth** query the answer is NO bcause: m[3][1] = 11 $\neq$ 0 (there are no elements in the lower part of the diagonal).

Important

Since the input/output for this problem can be very large, you need to use **fast input/output** to optimize performance, it is listed below how to do so for the available programming languages:

- **C++**:

Add these lines to your code:

```
ios::sync_with_stdio(false);
cin.tie(NULL);
cout.tie(NULL);
```

Instead of using **endl** for a new line, use **"\n"**

- **Python**: Import sys library

```
//input
sys.stdin.readline()
//output
sys.stdout.write("YES\n")
sys.stdout.write("NO\n")
```

- **Java**: Use the buffered reader and buffered writer classes

Import these java classes: java.io.BufferedReader; java.io.InputStreamReader; java.io.BufferedWriter; import java.io.OutputStreamWriter; java.util.StringTokenizer;

```
BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));
BufferedWriter writer = new BufferedWriter(new OutputStreamWriter(System.out));
```

For each line, create a string tokenizer:

```
StringTokenizer tokenizer = new StringTokenizer(reader.readLine());
```

To get a value use:

```
tokenizer.nextToken();
Integer.parseInt(tokenizer.nextToken()); //for example
```

Problem J : The Turn Game

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

Monkey D. Luffy and Roronoa Zoro are engaged in a strategic number swapping game. The game involves two numbers, A and B, each of length n. In each turn, the players will perform actions based on the values of digits of the numbers denoted by a_i and b_i ($1 \leq i \leq n$).
The rules of the game are:

- Game starts at turn 1 from the left digits of A and B.
- in the i-th turn, if $a_i > b_i$, the current player keeps playing the next turn.
- in the i-th turn, if $a_i < b_i$, the current player loses his turn and the other player plays the next turn.
- in the i-th turn, if $a_i = b_i$, the current player has two choices. Either he keeps playing the next turn or lets the other player play the next turn.
- The game ends in the $(n+1)$-th turn. The winner is the player who has the turn in the $(n+1)$-th turn.

Given A and B, your task is determine the winner of the game if Luffy begins playing in the first turn and both players play optimally.

Input:
The first line of input contains a single integer n ($1 \leq n \leq 4 \times 10^6$), the number of digits of A and B.

The second line of input contains a single integer A.

The second line of input contains a single integer B.

Output:
Print a single line containing "**Luffy**" if Luffy wins or "**Zoro**" if Zoro wins.

Note:
input:

```
3
234
155
```

output:

```
Luffy
```

input:

```
4
2394
1595
```

output:

```
Zoro
```

Explanation:
First example:
In the first turn, 2 > 1 so Luffy keeps playing. In the second turn, 3 < 5 so Luffy loses his turn. In the third turn, 4 < 5 so Zoro also loses his turn. In the fourth turn, the game ends with Luffy having this turn, so Luffy wins.

Second example:
In the first turn, 2 > 1 so Luffy keeps playing. In the second turn, 3 < 5 so Luffy loses his turn. In the third turn, Zoro choses to let Luffy play the next turn. In the fourth turn, 4 < 5 so Luffy loses his turn. In the fifth turn, the game ends with Zoro having this turn, so Zoro wins.

Problem K: Sanji The Gentleman

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

If you are a man, Sanji will give you hell. However, even if it means killing him, Sanji would never lay a finger on a woman. A true, nice Gentleman, right? Yeah, until the crew is attacked. Sanji is very strong and would manage an army by himself unless there is a woman in the army then he is screwed. The Strawhats crew are used to this fact, so, they would send Nami with him if they were sure there is at least one woman in the army so he doesn't get beaten to death.
Usopp uses his binoculars to scout the enemy troupes and wants to know whether he needs to send Nami with Sanji or if Sanji will manage by himself.
Given the list of the genders of the enemies, tell Usopp if he needs to send Nami too or not.

Input :

The first line contains the number ***N*** the length of the following string. $1 \leq N \leq 10^6$
The second line contains a string **S** the list of the genders of the enemies: **'W'** for *woman*, **'M'** for *man*.

Output :

Output "YES" without quotes if they should send Nami. Otherwise, output "NO"

Example:

Input :

```
22
MMMMMMMMMMMMMMMMMMMMMM
```

Output :

```
NO
```

Input :

```
24
MWWWMMMMMMMMMMMMMMMMMMMW
```

Output :

```
YES
```

Problem L : The circle of death

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

In the world of One Piece, there is a certain legend about a circle at sea where numerous ships vanished without a trace. This circle came to be known as the circle of death.
The straw hat pirates found themselves obligated to pass through this circle to reach a certain island.
Nami, their bright navigator, uncovered the secret behind this phenomenon.
She found out three key observations about the circle:

- The circle of death is formed by **n** specific points.
- Within the circle of death, there are danger zones and safe zones.
- The danger zones are the **right triangles** that can be formed using any 3 points from the **n** points forming the circle.

Help Nami find the number of danger zones so that the ship can avoid them to reach their destination safely.

Input Format

The first line contains an integer n $(3 \leq n \leq 10^5)$, the number of points.

The second line contains n space-separated integers $\mathbf{a_i}$ $(1 \leq a_i \leq 10^9)$ – the **length of the arc** between point i and point i+1 $(1 \leq i \leq n-1)$, and a_n is the distance between the point number n and the first point.

Output Format

Output the number of unique danger zones that appear in the circle.

Example:

Input :

```
4
3 2 1 4
```

Output :

```
2
```

Input :

```
3
1 2 3
```

Output :

```
1
```

Notes:

In the first example we can see that there are two possible right triangles ABC and ABD

Here AC = 3, CB = 2, BD = 1 and DA = 4