少年圖靈計畫
Young Turing Program

0_Hello World

(30 points)

Introduction

YTP Contest has started!

Let's verify everything first.

Is the internet setting correct?

Is the source code submission working well?

Do you use STDOUT output for program solutions?

Everything is ready! Go get 30 points now!! Go! Go! Go!

Statement

Please write a program to output Hello World!

Input Format

This problem requires no input.

Output Format

[A~Z][a~z], space, and common English punctuation.

Constraints

[A~Z][a~z], space, and exclamation mark "!".

Test Cases

Input 1

(no input)

Output 1

```
Hello World!
```

Illustrations

Input 1 has no input, simply output Hello World!

1_Stonks

(10 points)
Time Limit: 1 second
Memory Limit: 256 MB

Statement

Nathan is learning about personal finance lately, and after evaluating various investment returns, he decides to venture into the stock market. Now, he wonders if he can make a profit.

Assume that Nathan invests X dollars in a magical company's stock. The company will go through N quarters. Each quarter, the stock price is updated to a_i, and dividends are distributed. Stockholders receive $\lfloor \frac{a_i}{100} \rfloor$ dollars as dividends for each share they own. Nathan saves these dividends and uses them to buy as many shares of the company as possible until he no longer has enough dividends to purchase additional shares.

At the end of the Nth quarter, Nathan sells all his stocks and wants to check if he has made a profit. Can you help him write a program to quickly determine this?

Input Format

The first line contains two positive integers separated by a space: N and X. N represents the number of quarters, and X represents the initial purchase price of the stock.

The second line contains N positive integers separated by spaces: $a_1, a_2, \ldots, a_N$. a_i represents the stock price in the ith quarter.

Output Format

Please check if the amount of money Nathan earns after selling all the stocks in the Nth quarter, plus the accumulated dividends, is greater than the initial investment. If it is, output "stonks" (without quotes); otherwise, output "not stonks" (without quotes).

Constraints

- $1 \leq N, X, a_i \leq 1000$

Test Cases

Input 1

```
5 100
45 55 51 100 120
```

Output 1

```
stonks
```

Input 2

```
5 100
120 140 80 90 70
```

Output 2

```
not stonks
```

Input 3

```
10 100
130 180 200 300 400 500 250 100 15 60
```

Output 3

```
stonks
```

Illustrations

In test case 1 and 2, Nathan has only one stock and earns a dividend of $2.

In test case 3, Nathan earns a total dividend of 19 in the first 8 quarters and buys one stock in the 9th quarter. In the end, he has a total of 2 stocks and 4 dividends.

2_Learn_Japanese

(10 points)
Time Limit: 1 second
Memory Limit: 256 MB

Statement

One day, Nathan was doing his homework when he noticed a letter on the Jikuai's desk. The letter contained the following message:

> 尊敬なる雞塊様
>
> お世話になっております。恐れ入りますが、私どもより大切なお知らせがございます。YTP大会に関して、ご参加いただけることを心よりお願い申し上げます。
>
> 貴殿のお名前は、その創造性と技術によって多くの人々に尊敬されております。ですから、このような大会にご参加いただけることは、私たちにとって非常に光栄でございます。

Nathan couldn't understand the letter as he didn't know Japanese. Just then, Jikuai came over and explained it. It turned out to be a letter written in Japanese, inviting Jikuai to participate in the YTP competition. Nathan was envious of Jikuai's proficiency in Japanese and decided to learn Japanese in his spare time.

However, Nathan soon got stuck while learning verb conjugations, particularly the dictionary form (辞書型) to the ます form. It was challenging for him! Helpless, he turned to Jikuai for guidance. Jikuai explained that, first of all, Japanese verbs always end with the "u" (ウ段) sound. Then, without considering other exceptions, Japanese verbs can be roughly classified into the following five categories in the dictionary form:

- Godan Verbs: Verbs that end with a sound other than "ru" (る), or end with "ru" (る) but have an "a" (ア段), "u" (ウ段), or "o" (オ段) sound before "ru" (る). Examples: 行く (iku), 取る (toru).
- i-Ichidan Verbs: Verbs that end with "ru" (る) and have an "i" (イ段) sound before "ru" (る). Example: 信じる (shinjiru).
- e-Ichidan Verbs: Verbs that end with "ru" (る) and have an "e" (エ段) sound before "ru" (る). Example: 食べる (taberu).
- s-Irregular Verbs: Verbs that end with "suru" (する), do not belong to the Godan verbs. Example: 勉強する (benkyousuru).
- k-Irregular Verbs: Verbs that end with "kuru" (来る) and do not belong to the Godan verbs. Example: 来る (kuru).

Note that verbs ending with "suru" (する) do not belong to the Godan Verbs and are categorized as s-Irregular Verbs, while verbs ending with "kuru" (来る) do not belong to the Godan verbs and are categorized as k-Irregular Verbs.

Jikuai further explained that the conjugation to the ます form is as follows:

- Godan Verbs: Change the ending "u" sound (ウ段) to "i" sound (イ段) and add "ます" (masu). Examples: 行く (iku) -> 行きます (ikimasu), 取る (toru) -> 取ります (torimasu).
- i-Ichidan Verbs: Remove the ending "ru" (る) and add "ます" (masu). Example: 信じる (shinjiru) -> 信じます (shinjimasu).

- e-Ichidan Verbs: Remove the ending "ru" (る) and add "ます" (masu). Example: 食べる (taberu) -> 食べます (tabemasu).
- s-Irregular Verbs: Change the ending "suru" (する) to "します" (simasu). Example: 勉強する (benkyousuru) -> 勉強します (benkyousimasu).
- k-Irregular Verbs: Change the ending "kuru" (来る) to "来ます" (kimasu). Example: 来る (kuru) -> 来ます (kimasu).

Here is a table mapping the hiragana characters to the romanized pronunciation for the Japanese syllables used in this context:

ア段(a)	イ段(i)	ウ段(u)	エ段(e)	オ段(o)	
あ(a)	い(i)	う(u)	え(e)	お(o)	
か(ka)	き(ki)	く(ku)	け(ke)	こ(ko)	
さ(sa)	し(si)	す(su)	せ(se)	そ(so)	
た(ta)	ち(ti)	つ(tu)	て(te)	と(to)	
な(na)	に(ni)	ぬ(nu)	ね(ne)	の(no)	
は(ha)	ひ(hi)	ふ(fu)	へ(he)	ほ(ho)	
ま(ma)	み(mi)	む(mu)	め(me)	も(mo)	
や(ya)		ゆ(yu)		よ(yo)	
ら(ra)	り(ri)	る(ru)	れ(re)	ろ(ro)	
わ(wa)				を(wo)	ん(n)

The input for this problem will consist of Japanese sentences composed of the romanized syllables from the table above, ensuring that the sentences end with the dictionary form of a verb. Based on the given information, please transform the verb at the end of the sentence into its ます form and output it. It is guaranteed that all verb conjugations in the test cases can be completed using the provided information.

Note: The verb conjugation method described in this problem may differ from actual Japanese verb conjugation. Please focus on the description provided in the problem while solving it.

Input Format

The first line contains an integer T, which represents the number of test cases that follow.

Each test case consists of one line containing a string S, which is the input mentioned in the problem.

Output Format

For each test case, output the string that satisfies the requirements mentioned in the problem.

Constraints

- $1 \leq T \leq 100$
- $1 \leq |S| \leq 100$, $|S|$ represents the length of string S.

Test Cases

Input 1

```
3
taberu
okiru
sinjiru
```

Output 1

```
tabemasu
okimasu
sinjimasu
```

Input 2

```
5
aisiteiru
kyounaniwotaberu
yorunikakeru
bakuretusuru
okaesimousu
```

Output 2

```
aisiteimasu
kyounaniwotabemasu
yorunikakemasu
bakuretusimasu
okaesimousimasu
```

Illustrations

In example 1, the verbs `taberu`, `okiru`, and `sinjiru` are all verbs from i-Ichidan Verbs or e-Ichidan Verbs. The way to change them to the ます (masu) form is to remove "ru" and add "masu."

In example 2：

- `aisiteiru` ends with ru, and ru is preceded by i, which belongs to i-Ichidan Verbs, so the ます type should be `aisiteimasu`
- `kyounaniwotaberu` ends with ru, and ru is preceded by e, which belongs to e-Ichidan Verbs, so the ます type should be `kyounaniwotabemasu`
- `yorunikakeru` ends with ru, and ru is preceded by e, which belongs to e-Ichidan Verbs, so the ます type should be `yorunikakemasu`
- `bakuretusuru` ends in suru and belongs to s-Irregular Verbs, so the ます form should be `bakuretusimasu`
- `okaesimousimasu` ends in a sound other than ru and belongs to Godan Verbs, so the ます form should be `okaesimousimasu`

3_Eleven

(15 points)
Time Limit: 1 second
Memory Limit: 256 MB

Statement

"In certain spiritual and mystical belief systems, the number 11 is regarded as a highly spiritual number. It is believed to be associated with spiritual growth, intuition, and awakening of the soul."

"Some people believe that 11 represents a resonance or connection with the universe. It is seen as a reminder for individuals to become aware of their connection with the universe and can guide them towards higher goals and purposes."

The number 11 holds special significance in certain cultures and belief systems. According to legends, constructing a number that satisfies the following conditions before 11:11 PM can bring good luck:

- The number should not contain the digit 0.
- The sum of the digits of the number should be equal to n.
- The number should be divisible by 11.
- Among the numbers that satisfy the above three conditions, it should be the largest possible number.

To obtain good luck, given the number n, please output a number that satisfies the above conditions, or report if such a number does not exist.

The sum of the digits of a number refers to the total sum of its individual digits. For example, the sum of digits of 123 is 1 + 2 + 3 = 6.

Input Format

There is one line of input containing a positive integer n, which has the same meaning as described in the problem statement.

Output Format

Please output a single line containing the number that satisfies all the given conditions, or `-1` if no such number exists.

Constraints

- $1 \leq n \leq 10^5$

Test Cases

Input 1

```
2
```

Output 1

```
11
```

Input 2

```
3
```

Output 2

```
-1
```

Illustrations

It can be proven that there are no numbers that satisfy the given conditions when $n = 3$.

4_Street_Light

(15 points)
Time Limit: 1 second
Memory Limit: 256MB

Description

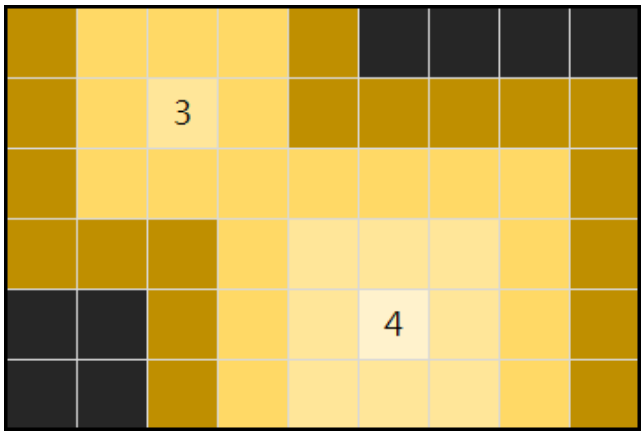
Daniel is the mayor of Starry City. He wants to install some street lights to light up the city.

The brightness of Starry City is shown on a $N \times M$ table. Daniel plans to install some street lights in some places. The power of the street lights can be different. The place where the street light will have the higest brightness, and the brightness decreases as the distance gets farther.

More specifically, we represent the cell in the r-th row and c-th column as (r, c). Assuming the i-th street light is located at (x_i, y_i) with the power of p_i, the brightness that this street light can provide to cell (r, c) is $\max(0, p_i - \max(|r - x_i|, |c - y_i|))$.

If there are multiple street lamps, the brightness of a cell will be the maximum brightness among all the lamps illuminating it. In other word, if K street lamps are installed, the brightness of cell (r, c) can be calculated as $\max\limits_{1 \le i \le K}(\max(0, p_i - \max(|r - x_i|, |c - y_i|)))$.

Consider the following example. The size of the city is 6×9. The street lamp located at $(2, 3)$ has a power of 3, while the street lamp located at $(5, 6)$ has a power of 4. The lighter the color of a cell, the higher its brightness, while dark gray cells indicate a brightness of 0.



Daniel told you what brightness he wants for each cell in the Starry City. Can you write a program to help Daniel arrange the positions and powers of each street light?

You can install as many street light as you want because Daniel is rich. However, no more than one street light can be installed on one single cell.

Input Format

The first line of the input contains two integers N, M -- the size of Starry City. Then N lines follow, the i-th line contains M integers $b_{i,j}$ -- the brightness that Daniel wants the cell (i, j) to be.

Output Format

If it is impossible to satisfy Daniel's requirement, output -1 in one line.

Otherwise, output an integer K -- the number of street lights in the first line. Then K lines follow, the i-th line contains three integers x_i, y_i, p_i -- the position and the power of the i-th street light.

The integer K must satisfy $0 \leq K \leq N \times M$. The position of the street lights (x_i, y_i) must satisfy $1 \leq x_i \leq N$ and $1 \leq y_i \leq M$, and the positions of any two street lights must be different. The power of the street lights p_i must satisfy $1 \leq p_i \leq 1000$.

You don't need to minimize the number of street lights, and you can output any of the arrangement if there is more than one valid answer.

Constraints

- $1 \leq N, M \leq 1000$.
- $0 \leq b_{i,j} \leq 1000$ ($1 \leq i \leq N, 1 \leq j \leq M$).

Test Cases

Input 1

```
6 9
1 2 2 2 1 0 0 0 0
1 2 3 2 1 1 1 1 1
1 2 2 2 2 2 2 2 1
1 1 1 2 3 3 3 2 1
0 0 1 2 3 4 3 2 1
0 0 1 2 3 3 3 2 1
```

Output 1

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

Input 2

```
3 3
3 1 4
1 5 9
2 6 5
```

Output 2

```
-1
```

Illustrations

For Example 1, please refer to the figure in the description part.

For Example 2, it's impossible to satisfy Daniel's requirement.

5_Game_Stamina

(15 points)
Time Limit: 1 second
Memory Limit: 256 MB

Description

In the game YTP (Young Turing Protection), players can protect Young Turing from enemies in defense battles, enjoying an exciting gaming experience. However, to impose certain limitations on players, each time they participate in a defense battle, they must consume "stamina points." The stamina points increase by one every k minutes. Therefore, players often have to wait for some time to earn enough stamina points to continue playing the game.

Little Y, who is addicted to YTP, has played several defense battles today. However, for a certain period of time, he will not be able to play a full defense battle. Instead, he occasionally checks the current stamina points in the game to calculate how many defense battles he can play when he has free time.

However, even though he knows that the stamina points increase by one every k minutes, Little Y is unaware of the specific minutes at which the increase occurs. For example, if $k = 3$, the stamina points could increase at minutes $3, 6, 9, 12, \ldots$, or at minutes $1, 4, 7, 10, \ldots$, or even at minutes $2, 5, 8, 11, \ldots$. In other words, there is definitely a remainder $0 \leq r < k$ such that whenever t minutes have passed and the remainder of t divided by k is r, the stamina points increase by one.

Given that Little Y will perform N stamina checks in the upcoming time period, where the i-th check occurs at minute t_i with stamina points s_i, and he has not consumed any stamina points during the intervals between checks. Your task is to output, for each remainder, whether it is a possible r after each check.

Input Format

The first line contains two integers N and k, representing the number of stamina checks performed by Little Y and the time interval for stamina point increments, respectively.

The next N lines contain two integers each, t_i and s_i, representing the minute of the i-th check performed by Little Y and the stamina points at that time.

Output Format

For each check, output a line containing a `01` string $p_0 p_1 p_2 \ldots p_{k-1}$, where p_j is `1` if remainder j is a possible r, and `0` otherwise. This indicates the possible remainders r up to the i-th check.

Constraints

- $1 \leq N, k \leq 1000$.
- $1 \leq t_i, s_i \leq 10^9$ $(1 \leq i \leq N)$.
- $t_1 < t_2 < \cdots < t_N$.
- There is guaranteed to be at least one remainder r that satisfies all the input conditions, meaning no contradictory situations will occur.

Test Cases

Input 1

```
5 3
1 1
3 2
6 3
8 3
11 4
```

Output 1

```
111
101
101
100
100
```

Input 2

```
10 10
3971 82568
35370 85708
39553 86126
43370 86508
60282 88199
63915 88562
65994 88770
77566 89927
81695 90340
96803 91851
```

Output 2

```
1111111111
1011111111
1000111111
1000111111
1000111111
1000001111
1000001111
1000000111
1000000111
1000000111
```

Illustrations

For Input 1, the five checks correspond to the following:

- Little Y performs his first check at minute 1, and his stamina points are 1. Since there is not enough information yet, all remainders are possible values of r.
- Little Y performs his second check at minute 3, and his stamina points are 2. If $r = 1$, the stamina points would only increase at minutes $1, 4, 7$, and so on. Therefore, the stamina points should not have increased at minute 3, so 1 is not a valid remainder r.
- Little Y performs his third check at minute 6, and his stamina points are 3. No new information is obtained.
- Little Y performs his fourth check at minute 8, and his stamina points are 3. If $r = 2$, the stamina points would increase at minutes $2, 5, 8$, and so on. Therefore, the stamina points must have increased at minute 8, so 2 is not a valid remainder r.
- Little Y performs his fifth check at minute 11, and his stamina points are 4. No new information is obtained.

For Input 2, the checks are provided as a larger test case for self-evaluation, with increased input size.

6_Paintball_Game

(15 points)
Time Limit: 1 second
Memory Limit: 256 MB

Description

Recently, Bamboo has been passionately involved in a paintball game. This game is played on an $N \times N$ gameboard and involves a total of M players. In the game, players are numbered from 1 to M, and they take turns performing actions in sequential order based on their numbers. When it is player i's turn, ze activates a paintball bomb with an explosive power of l_i at position (x_i, y_i) on the gameboard. After that, the bomb uses color i to color the grids in a cross-shaped pattern with a width of l_i centered at the activation point. And it will cover the existing colors on the grids. That is, the color of all grids (x', y') satisfies $\min(|x' - x_i|, |y' - y_i|) \leq \frac{l_i - 1}{2}$ will change to i. And these squares that are colored with player i 's color are referred to as their "territory".

The game ends when all players have completed their actions. At this point, the player with the largest territory is the winner. In order to stay updated on the battlefield, after each round, Bamboo wants to calculate the number of territories controlled by player p_i within the rectangular area formed by the grid (a_i, b_i) and grid (c_i, d_i). With time constraints on decision-making during the game, Bamboo hopes to develop a program that can quickly compute the desired information. Can you help Bamboo to achieve his goal?

Input Format

The first line contains two integers N, M, representing the width of the gameboard and the number of players.

And there will be M group of input, each group contains two lines, representing the operation in one turn.

The first line of the i-th group contains three integers, l_i, x_i, y_i, representing the explosive power, and the activation position.

And the second line of the i-th group contains five integers, p_i, a_i, b_i, c_i, d_i, representing the target player and the two points of the investigated area.

Output Format

The output should contain M lines.

The i-th line should contain a single integer, representing the number of territory that belongs to player p_i in the specific area.

Constraints

- $1 \leq N, M \leq 5000$.
- $1 \leq x_i, y_i \leq N$ ($1 \leq i \leq M$).
- $1 \leq l_i \leq N$ ($1 \leq i \leq M$).

- l_i is odd ($1 \leq i \leq M$).
- $1 \leq a_i \leq c_i \leq N$ ($1 \leq i \leq M$).
- $1 \leq b_i \leq d_i \leq N$ ($1 \leq i \leq M$).
- $1 \leq p_i \leq M$ ($1 \leq i \leq M$).

Test Cases

Input 1

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

Output 1

```
5
3
7
```

Input 2

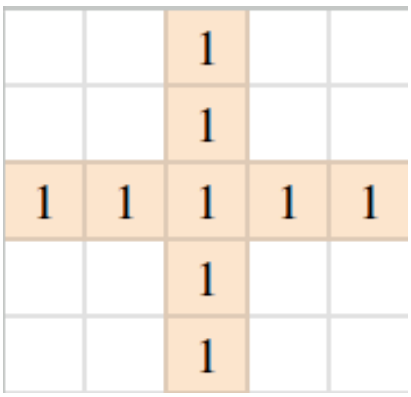
```
6 4
5 2 1
3 4 5 5 6
1 3 2
1 3 1 5 4
3 3 2
2 1 1 6 6
3 6 6
1 1 2 3 6
```

Output 2

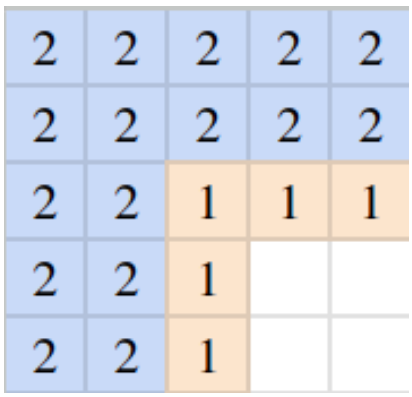
```
0
5
0
1
```

Illustrations

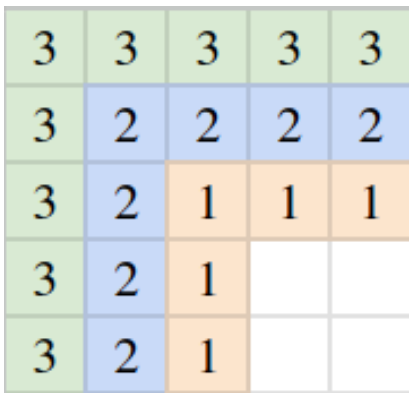
In Example 1, the board after the moves of three players are shown in the following pictures:



after first player

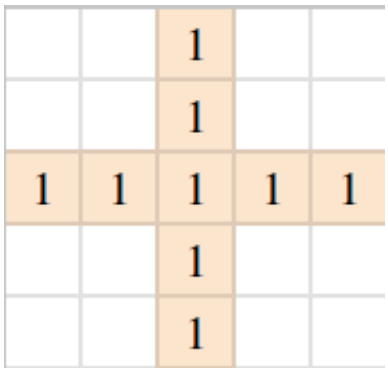


after second player

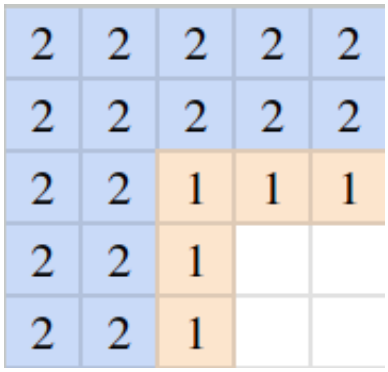


after third player

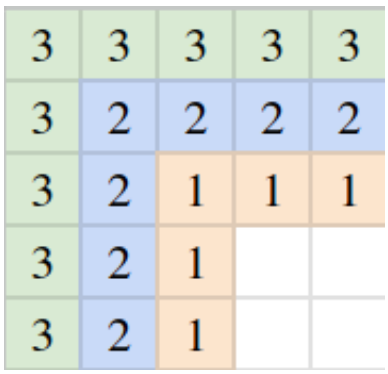
after the player 1 operation, the gameboard is like below:



After the player 2 operation, the gameboard is like below:



After the player 3 operation, the gameboard is like below:



7_Toll_Plaza_Simulator

(15 points)
Time Limit: 3 seconds
Memory Limit: 512 MB

Description

VVivvi has recently become obsessed with Truck Simulator. However, after playing for a while, she realized that she wasn't very good at driving trucks. "I should play Toll Plaza Simulator instead," she thought when she got stuck in the middle of the road because of taking the wrong lane.

The game is quite simple: there is a toll plaza located somewhere along a one-way highway with N lanes. Each lane has a toll booth responsible for charging the fees from the vehicles passing through the toll booth. The toll booths and lanes are numbered from left to right as $1, 2, \ldots, N$. During the night, there are no vehicles passing through the toll booth, but as soon as the day breaks, numerous vehicles suddenly appear. VVivvi has anticipated that tomorrow morning, at sunrise, exactly 1 vehicle will appear on each lane.

Initially, this was a straightforward game, but sometimes unexpected situations arise. Some toll booths may malfunction and become inoperative. If a driver realizes that the toll booth in their lane is closed, they will move their vehicle to the nearest lane with an functioning toll booth. If there are two lanes with functioning toll booths at the same distance, the driver will choose the one with the smaller lane number. The distance of two lanes is the absolute difference of their lane numbers. The objective of this game is to allocate manpower to each toll booth, avoiding situations where employees are overworked or have too little to do.

Currently, it is nighttime, and all toll booths are functioning normally. Before sunrise, there will be Q events, each of which can be one of the following types:

- $1 \ \ell \ r$: For all i ranging from ℓ to r, the toll booth i is malfunctioning. Note that toll booth i may already be malfunctioning.
- $2 \ \ell \ r$: For all i ranging from ℓ to r, the toll booth i is functioning normally. Note that toll booth i may already be functioning.

Although everything happens before sunrise when no vehicles are passing through, VVivvi wants to predict the level of chaos at sunrise after each event. The level of chaos at sunrise is defined as the total distance traveled by all vehicles present at that time. If a vehicle comes from lane i and pays at toll booth j, its moving distance is $|i - j|$.

Since toll booths 1 and N are located at the edges and can be easily managed by VVivvi, no events will occur for these toll booths. In other words, these two toll booths are always open.

Input Format

The first line contains two integers, N and Q, representing the number of lanes and the number of events, respectively.

The next Q lines each contain three integers, t, ℓ, and r, representing the type and range of the i-th event.

Output Format

Output Q lines, where the i-th line contains an integer representing the predicted level of chaos at sunrise after the i-th event.

Constraints

- $3 \leq N \leq 10^6$.
- $1 \leq Q \leq 10^6$.
- $t \in \{1, 2\}$.
- $1 < \ell \leq r < N$.

Test Cases

Input 1

```
10 5
1 2 3
1 4 5
2 5 7
1 6 9
1 4 6
```

Output 1

```
2
6
4
10
20
```

Input 2

```
1000000 10
1 21586 928842
1 619334 853861
2 274417 572426
1 239830 413748
2 79123 839131
1 2410 978513
2 350282 605034
1 22256 122739
1 463714 491706
1 588774 892375
```

Output 2

```
205779269641
205779269641
47739274528
70206420196
2839716097
238195242756
65125733632
65125733632
65321649641
68424333811
```

Illustrations

In Example 1, each event is explained as follows:

1. Toll booths 2 and 3 are closed, causing vehicles in lane 2 to move to toll booth 1, and vehicles in lane 3 to move to toll booth 4. Vehicles in other lanes will remain at their original toll booths.
2. Toll booths 4 and 5 are closed, resulting in the closure of toll booths 2 to 5. Vehicles in these lanes will move to toll booths 1, 1, 6, and 6, respectively.
3. Toll booths 5, 6, and 7 are opened, while toll booths 2 to 4 are closed. Vehicles in these lanes will move to toll booths 1, 1, and 5, respectively.
4. Toll booths 6 to 9 are closed, leaving only toll booths 1, 5, and 10 opened. Vehicles in each lane will move to toll booths 1, 1, 5, 5, 5, 5, 5, 10, 10, and 10, respectively.
5. Toll booths 4 to 6 are closed, leaving only toll booths 1 and 10 opened. Vehicles in lanes 1 to 5 will move to toll booth 1, while the remaining vehicles will move to toll booth 10.

8_Ice_Skating

(4 points /16 points)

Time Limit: 3 seconds
Memory Limit: 512MB

Description

As a Pokémon enthusiast, you enjoy traveling around and challenging gyms.
One day, you learn that Pokémon master Ash has opened a brand new gym, and you can't wait to go challenge it.
Upon arriving at the gym, you encounter a problem: you can't find the entrance to the challenge.
The gym is actually an $N \times M$ ice rink, with the challenge entrance located in one of the grid cells.
Inside the ice rink, you can only move in the following ways:

1. Choose a direction (up, down, left, or right).
2. Move in the chosen direction until you hit an obstacle, at which point you must stop.

Initially, you will start on the grid marked as `S` on the ice rink.
Please find the minimum number of moves required to reach the challenge entrance marked as `E`.
If it is impossible to reach the entrance, output -1.

Please note that the areas outside the ice rink are considered obstacles, and reaching the challenge entrance counts as reaching it even if you don't stop at the entrance.

Input Format

The first line of input consists of two integers, N and M, representing the length and width of an ice rink.

The next N lines each contain M characters $s_{i,1}, s_{i,2}, \ldots, s_{i,M}$, each of which is one of `#.SE`.

- `#` represents an obstacle.
- `.` represents an empty space.
- `S` represents your starting point.
- `E` represents the entrance to the challenge arena.

Output Format

Please output a single line containing a integer representing the answer. If it is impossible to reach the entrance, output -1.

Constraints

- $2 \leq N, M \leq 100\,000$.
- $N \times M \leq 1\,000\,000$.
- $s_{i,j} \in \{\mathtt{\#}, \mathtt{.}, \mathtt{S}, \mathtt{E}\}$ $(1 \leq i \leq N, 1 \leq j \leq M)$.

- It is guaranteed that there is exactly one `S` and one `E` on the ice rink.

Subtasks

- Subtask 1 (4 points) $N \times M \leq 10\,000$.
- Subtask 2 (16 points) No additional constraints.

Test Cases

Input 1

```
5 5
S....
.....
.....
.....
....E
```

Output 1

```
2
```

Input 2

```
5 5
S.#..
.....
#.###
.....
..E..
```

Output 2

```
3
```

Input 3

```
2 2
S#
#E
```

Output 3

```
-1
```

Input 4

```
4 4
....
.S..
..E.
....
```

Output 4

```
-1
```

Illustrations

In the first example, one possible sequence of moves is to go right and then down. It is clear that it is impossible to reach the destination in just one move.

In the second example, the only way to reach the destination within three moves is to go right, down, and then right. Even though the last move doesn't make you stop at the challenge entrance, passing through it still counts as reaching it.

In the third example, you are unable to make any moves.

In the fourth example, please note that you can only stop after hitting an obstacle.

9_Rectangles

(2 points /3 points /15 points)
Time Limit: 3 seconds
Memory Limit: 512MB

Description

On an $n \times n$ grid G, some cells have obstacles, while others do not.

The question is: How many rectangles are there that do not contain any obstacles?

In other words, how many sets of two-dimensional subarrays are there in the $n \times n$ matrix where there are no obstacles present?

Input Format

The first line of the input will contain a positive integer n.

In the following n lines, each line will consist of n characters $G_{i,0}, G_{i,1}, \ldots, G_{i,n-1}$ representing the presence or absence of obstacles at each position.

If there is an obstacle, the character will be '#'. If there is no obstacle, the character will be '.'.

Output Format

Please output a single line containing a non-negative integer representing the number of rectangles that do not contain any obstacles.

Constraints

- $1 \leq n \leq 5000$.
- $G_{i,j} \in \{\texttt{\#}, \texttt{.}\}$ $(0 \leq i, j \leq n-1)$.

Subtasks

- Subtask 1 (2 points) $1 \leq n \leq 50$.
- Subtask 2 (3 points) $1 \leq n \leq 200$.
- Subtask 3 (15 points) No additional constraints.

Test Cases

Input 1

```
4
...#
#..#
..#.
.#.#
```

Output 1

```
21
```

Input 2

```
5
.#.#.
#...#
.##.#
###..
..#.#
```

Output 2

```
25
```

Illustrations

In Example 1:

There are 10 rectangles of size 1x1.

There are 4 rectangles of size 1x2.

There is 1 rectangle of size 1x3.

There are 4 rectangles of size 2x1.

There is 1 rectangle of size 2x2.

There are a total of 21 rectangles.

10_Eating_Cake

(5 points /20 points)
Time Limit: 3 seconds
Memory Limit: 512MB

Description

Russian Roulette Takoyaki is a well-known game, in some of the takoyaki balls, there is one with very spicy ingredient mixed in, and each participant randomly picks a takoyaki to eat.

On the other hand, cakes are famous desserts. However, sometimes they may be too sweet for some people.

Therefore, combining Russian Roulette Takoyaki and sweet cake, a new game was created – the Cake Eating Game.

Here are the rules:

There are two players, A and B, each with a sweetness life value, LA and LB, respectively.

At the beginning, there is an $N \times M$ cake in the shape of a chessboard. For each cell on the cake, its sweetness may be too high or moderate.

Each time a player eats a cell that is too sweet, their sweetness life value will be reduced by one.

In the game, the two players take turns moving, with A going first. Each time, they select a cell (i, j) that has not been eaten and cuts out and eats all the cakes in the lower right corner, which means all the cells (x, y) satisfying $i \leq x \leq N$, $j \leq y \leq M$ and are still untouched will be eaten.

As shown in the following figure, if cell $(2, 2)$ is selected as the upper left corner, all the cells shaded in gray must also be eaten.

(1, 1)	(1, 2)	(1, 3)	(1, 4)
(2, 1)	(2, 2)	(2, 3)	(2, 4)
(3, 1)	(3, 2)		

When a player's sweetness life value becomes zero or negative, that player loses the game.

Given an $N \times M$ cake and whether each cell is too sweet or not, as well as the sweetness life values of A and B, please determine which player has a winning strategy.

Input Format

The first line of input contains a positive integer Q, which represents the number of test cases.

For each test case, the first line contains two positive integers N and M, which represent the size of the cake.

Each of the next N lines contains M integers $x_{i,1}, x_{i,2}, \ldots, x_{i,M}$, where $x_{i,j} = 1$ means the sweetness of cell (i, j) is too high, and $x_{i,j} = 0$ means the sweetness of cell (i, j) is moderate.

The last line of each test case contains two positive integers LA and LB, representing the sweetness life value of players A and B, respectively.

Output Format

For each test case, if player A has a winning strategy, output `A`. If player B has a winning strategy, output `B`. Otherwise, output `Tie`.

Constraints

- $1 \leq Q \leq 300$.
- $1 \leq N, M$.
- $1 \leq N \times M \leq 400$.
- $x_{i,j} \in \{0, 1\}$ $(1 \leq i \leq N, 1 \leq j \leq M)$.
- $1 \leq LA, LB \leq N \times M$.

Subtasks

- Subtask 1 (5 points) $N = 1$.
- Subtask 2 (20 points) No additional constraints.

Test Cases

Input 1

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

Output 1

```
A
B
Tie
```

Illustrations

In the first test case, if A eats all the cells in the right two rows in the first round, B then will be forced to eat the over sweet cell at position $(3, 1)$, which will cause B's sweetness life value to become zero and lose the game.

In the second test case, A must eat the too sweet cell at position $(2, 3)$ in the first round, which will cause his sweetness life value to become zero and lose the game.

In the third test case, after A consumes the cell at position $(1, 1)$ in the first round, the entire cake is eaten up. As a result, no one's life value becomes zero or negative, leading to a draw.