Problem A. PROCOM Day 2

Time limit 2000 ms
Mem limit 262144 kB

Problem Statement

Ashar woke up early on the morning of **Procom Day 2**, still buzzing with excitement over the success of Day 1. The highlight, of course, was the **competitive programming competition**, masterfully organized under the guidance of **Huzaifa Rashid**, the CS Competition Head.

However, as Ashar reached the venue, he noticed a **major problem**—the **main stage for the final ceremony** had become uneven overnight! Since the closing ceremony needed to be **flawless**, the stage planks had to be **non-decreasing in height from to create a smooth transition for the final event.**

The stage consists of n wooden planks arranged in a line, where the i-th plank has a height of a[i]. To fix the stage, Ashar can perform the following operation:

- **In one operation, he can select any contiguous subsegment of planks that is already non-decreasing in height and increase the height of each plank in that segment by 1.**

With the closing ceremony approaching, Ashar needs to determine the minimum number of operations required to make the stage perfectly smooth and ready for the grand finale. Can you help him close the event on a perfect note?

Input

The input consists of **multiple test cases**.

The first line contains a single integer **t** ($1 \le t \le 100$) — the number of test cases.

Each test case consists of an integer **n** ($1 \le n \le 2 \cdot 10^5$) — the number of planks on the stage.

The next line contains **n** space separated integers $\mathbf{a_1, a_2, \ldots, a_n}$ ($0 \le a_i \le 10^9$) — the heights of the planks.

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 single integer — **the minimum number of operations required to make the planks non-decreasing.**

Examples

Input	Output
3 4 5 3 2 5 5 1 2 3 5 3 3 1 1 1	3 2 0

Explanation

For the first test case, the subarray with which Ashar performs the operation is bolded.

In the first test case:

- First operation:

  $[5, 3, \mathbf{2}, 5] \to [5, 3, \mathbf{3}, 5]$

- Second operation:

  $[5, \mathbf{3}, \mathbf{3}, 5] \to [5, \mathbf{4}, \mathbf{4}, 5]$

- Third operation:

  $[5, \mathbf{4}, \mathbf{4}, 5] \to [5, \mathbf{5}, \mathbf{5}, 5]$

In the third test case, the array is already nondecreasing, so Ashar does 0 operations.

Problem B. Welcome SunderStem

Time limit 2000 ms
Mem limit 262144 kB

Problem Statement

Sir Ahmed Imran and **Umair Mirza**, two of Pakistan's top competitive programmers, have arrived at FAST Karachi! While taking a stroll in the garden, they noticed that there are n trees, neatly numbered from 1 to n from left to right.

Initially, the i-th tree is painted with color c_i. However, since they recognize only m different colors, each color satisfies $0 \leq c_i \leq m$, where a value of 0 means that the tree is uncolored.

The duo decides to paint only the uncolored trees (those with $c_i = 0$). They can choose any of the m colors (numbered from 1 to m) for each uncolored tree. Painting the i-th tree with color j requires exactly $p_{i,j}$ litres of paint.

They define the *beauty* of a coloring as the **minimum** number of contiguous groups into which the n trees can be partitioned so that every tree in a group shares the same color. For example, if the trees are colored (from left to right) as $2, 1, 1, 1, 3, 2, 2, 3, 1, 3$, the beauty of this coloring is 7 since the trees can be split into 7 groups:
$\{2\}, \{1, 1, 1\}, \{3\}, \{2, 2\}, \{3\}, \{1\}, \{3\}$.

Sir Ahmed Imran and Umair Mirza wish to paint all the uncolored trees so that the overall beauty of the coloring is exactly k. Your task is to help them determine the minimum amount of paint (in litres) needed to achieve this, bearing in mind that the already colored trees must remain unchanged.

Input

The first line contains three integers, n, m and k ($1 \leq k \leq n \leq 100$, $1 \leq m \leq 100$) — the number of trees, number of colors and beauty of the resulting coloring respectively.

The second line contains n integers $c_1, c_2, \ldots, c_n$ ($0 \leq c_i \leq m$), the initial colors of the trees. c_i equals to 0 if the tree number i is uncolored, otherwise the i-th tree has color c_i.

Then n lines follow. Each of them contains m integers. The j-th number on the i-th of them line denotes $p_{i,j}$ ($1 \le p_{i,j} \le 10^9$) — the amount of litres the friends need to color i-th tree with color j. $p_{i,j}$'s are specified even for the initially colored trees, but such trees still can't be colored.

Output

Print a single integer, the minimum amount of paint needed to color the trees. If there are no valid tree colorings of beauty k, print -1.

Example 1

Input	Output
3 2 2 0 0 0 1 2 3 4 5 6	10

In this case, coloring the trees with colors $2, 1, 1$ minimizes the amount of paint used, which equals to $2 + 3 + 5 = 10$. Note that $1, 1, 1$ would not be valid because the beauty of such coloring equals to 1 ($\{1, 1, 1\}$ is a way to group the trees into a single group of the same color).

Example 2

Input	Output
3 2 2 2 1 2 1 3 2 4 3 5	-1

In the this case, all the trees are colored, but the beauty of the coloring is 3, so there is no valid coloring, and the answer is -1.

Example 3

Input	Output
3 2 2 2 0 0 1 3 2 4 3 5	5

Example 4

Input	Output
3 2 3 2 1 2 1 3 2 4 3 5	0

In this case, all the trees are colored and the beauty of the coloring matches k, so no paint is used and the answer is 0.

Problem C. Vali in Code Sprint

Time limit 1000 ms
Mem limit 131072 kB

Problem Statement

Vali had just completed the **final round** of the **Code Sprint Competition**—the ultimate showdown of Procom Day 2. Now, all that remained was the final score calculation. Had he performed well enough to prove himself among the best?

In this round, Vali attempted **N** out of **20** easy coding problems and **M** out of **15** hard problems. However, not all of his answers were correct—he made mistakes on **X** easy problems and **Y** hard problems. The rest of his attempts were correct.

Each easy coding problem is worth **1 point**, while each hard problem is worth **2 points**. Unattempted or incorrectly solved problems yield **0 points**.

To consider his performance a success, Vali's total score must be **strictly greater than half** of the maximum possible score for the contest.

Can you determine if Vali's final score is enough to cement his status as one of the standout performers of the competition?

Input

The input consists of a single line containing four integers: N ($0 \le N \le 20$), M ($0 \le M \le 15$), X ($0 \le X \le N$), Y ($0 \le Y \le M$) — the number of easy coding problems Vali attempted, the number of hard coding problems Vali attempted, the number of easy problems he got wrong and the number of hard problems he got wrong respectively

Output

Output a single line:

- **"LOLOS"** if Vali's total score is strictly greater than half of the maximum possible score, meaning he qualifies.
- **"TIDAK LOLOS"** if his score does not meet the required threshold.

Make sure to take care of case-sensitivity. Print everything in capital.

Example 1

Input	Output
11 14 2 4	LOLOS

Vali successfully solved **9** easy coding problems and **10** hard coding problems correctly. His total score is **29**, which is enough to qualify for victory in this intense showdown.

Example 2

Input	Output
0 15 0 4	TIDAK LOLOS

Vali answered 0 easy coding problems and 11 hard problems correctly, so the total score is 22, which means he will not pass the selection.

Problem D. Algo Showdown Final

Time limit 250 ms
Mem limit 1024 kB

Problem Statement

The final round of **Algo Showdown** has just concluded, and the competition hall is buzzing with excitement—except for a few participants from the **MCA team**. They did not perform **up to the mark** in the contest and, feeling dejected, they now seek to **manipulate the results** in a last-ditch effort to save face.

The final challenge was a **True/False problem set**, consisting of **K** questions. Each of the **N** participants submitted their answers, but due to an error in the judging system, the correct answer key **was lost** before grading could take place!

Seizing the opportunity, the MCA team **volunteers to reconstruct the answer key**. However, their true intention is to **set the answer key in a way that maximizes the lowest score any participant receives**—ensuring that even their worst-performing member isn't completely embarrassed.

Your task is to determine the **highest possible minimum score** that can be achieved if the MCA team sets the answer key **as optimally as possible**.

Input

The first line contains two integers N ($1 \le N \le 10^3$), K ($1 \le K \le 10$) — the number of participants and the number of True/False questions.

The next N lines each contain a string of length K, consisting only of the characters 'T' (True) and 'F' (False), representing the answers submitted by each participant.

Output

Output a single integer — **the maximum possible** lowest score that can be achieved by optimally setting the answer key.

Example 1

Input	Output
5 4 TFTF TFFF TFTT TFFT TFTF	2

Example 2

Input	Output
3 5 TFTFT TFTFT TFTFT	5

Problem E. Job Fair

Time limit 1000 ms
Mem limit 262144 kB

Problem Statement

The **Procom Job Fair** has just wrapped up, and representatives from **n companies** are now heading to a networking dinner. To transport them, the organizers have arranged a **single shuttle bus** with **r rows**, where each row has exactly **2 seats**.

Each company **i** has $\mathbf{a_i}$ representatives, and all representatives must be seated in the bus. A representative is considered **happy** if:

- They are seated in the same row as another representative from their company, or
- They are sitting **alone in a row**, with the seat next to them empty.

The event organizers want to **maximize the number of happy people** by seating the representatives in the most optimal way possible.

Given that everyone **must** be seated in the bus, determine the **maximum number of happy people** that can be achieved.

It is guaranteed that all representatives will fit on the bus. Formally, it is guaranteed that $\sum_{i=1}^{n} a_i \leq 2r$.

Input

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

The first line of each test case contains two integers n and r ($1 \leq n \leq 100$; $1 \leq r \leq 500$) — the number of families and the number of rows in the bus.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \leq a_i \leq 10$) — the number of family members in each family.

Output

For each test case, output the maximum number of happy people in an optimal seating arrangement.

Example

Input	Output
4 3 3 2 3 1 3 3 2 2 2 4 5 1 1 2 2 4 5 3 1 1 3	4 6 6 6

Explanation

In the first test case, the two representatives from the first company can sit together in the first row, while the two representatives from the second company can sit together in the second row. The remaining representative from the second company can sit in the third row along with a representative from the third company. This seating arrangement is shown below, where the **4 happy people** are colored green.

1	1
2	2
2	3

In the second test case, a possible seating arrangement with **6 happy people** is shown below.

3	3
1	1
2	2

Problem F. Debug or Die!

Time limit 1000 ms
Mem limit 131072 kB

Problem Statement

At the **Procom Debug or Die** competition, participants must navigate through a series of increasingly difficult bug-infested codebases. Only those with enough debugging expertise can resolve every issue and emerge victorious!

A participant starts with an initial debugging skill level X.

There are N buggy code files, each with a bug complexity level A_i for $1 \le i \le N$.

The debugging process follows these rules:

- In each round, the participant must choose one of the buggy code files to fix.
- If their debugging skill level is less than or equal to the bug's complexity, the codebase crashes, and they fail the challenge.
- Otherwise, they successfully resolve the bug, and their debugging skill level increases by the complexity of the bug they just fixed.
- This process continues until all bugs are fixed or the participant fails.
- If they manage to fix all bugs, they win the challenge.

Determine the minimum initial debugging skill level X required for the participant to fix all bugs without failing.

Input

The input consists of two lines:

- The first line contains a single integer N ($1 \le N \le 200000$) — the number of buggy code files.
- The second line contains N space-separated integers $A_1, A_2, \ldots, A_N (1 \le A_i \le 10^9)$ — the complexity level of each buggy code file.

Output

Output a single integer — **the minimum** initial debugging skill level X required to fix all bugs without failing.

Example

Input	Output
4 3 1 6 5	3

Explanation

If the initial debugging skill level of `3`. The challenge can be completed using the following steps:

1. Choose the `2`-nd buggy code file (complexity `1`). Since `3 > 1`, the bug is fixed, and the participant's skill level increases to `3 + 1 = 4`.
2. Choose the `1`-st buggy code file (complexity `3`). Since `4 > 3`, the bug is fixed, and the participant's skill level increases to `4 + 3 = 7`.
3. Choose the `3`-rd buggy code file (complexity `6`). Since `7 > 6`, the bug is fixed, and the participant's skill level increases to `7 + 6 = 13`.
4. Choose the `4`-th buggy code file (complexity `5`). Since `13 > 5`, the bug is fixed, and the participant's skill level increases to `13 + 5 = 18`.

At this point, all buggy code files have been fixed, and the challenge is completed successfully.

It can be proven that there is no way to fix all bugs with an initial skill level smaller than `3`, making `3` the minimum required starting skill level.

Problem G. Concert

Time limit 2000 ms
Mem limit 262144 kB

Problem Statement

At **PROCOM**—the flagship tech event of FAST-NUCES Karachi—you're in for a mixed bag. On one hand, there are **mind-bending** technical competitions and a job fair that might actually lead to a decent career. On the other, there's the concert—the **absolute bane** of PROCOM, a cacophony of off-key tunes and uninspired performances that even your worst nightmare wouldn't endorse.

Due to the twisted bureaucratic system at FAST, every event has prerequisites. That means if you want to attend a main event (those technical competitions or job fair sessions that actually matter), you might be forced to slog through a series of prerequisite events—even if that chain includes the dreaded concert. Trust us, if there were any way to remove the concert, **we'd have done it a long time ago.**

Your mission is to help a PROCOM participant design the ultimate schedule: attend all the k main events while enduring as few extra events as possible. Print an order in which to attend the necessary events—each event exactly once—that respects all prerequisite relations.

NOTE: Any event which is not main is extra.

Input

The first line contains two integers n and k ($1 \le k \le n \le 10^5$) — the total number of events and the number of main events.

The second line contains k distinct integers from 1 to n — the numbers of the main events.

Then n lines follow, each describing an event: the i-th line corresponds to event i. Each line starts with an integer t_i ($0 \le t_i \le n - 1$) — the number of events that must be attended before event i. Then follows a sequence of t_i distinct integers from 1 to n — the numbers of events (in random order) that must be attended before event i. It is guaranteed that no event depends on itself.

It is guaranteed that the sum of all values of t_i does not exceed 10^5.

Output

Print -1 if it is impossible to attend all main events.

Otherwise, in the first line print an integer m — the minimum total number of events that must be attended (including all main and required prerequisite events). In the second line, print m distinct integers — the event numbers in the chronological order in which they should be attended. If there are several answers, any of them is allowed.

Example 1

Input	Output
6 2 5 3 0 0 0 2 2 1 1 4 1 5	5 1 2 3 4 5

In the first test you can attend event number 1 and 2, after that you can attend the event number 4, then you can take the event number 5, which is the main. After that you have to attend the event number 3, which is the last not attended main event.

Example 2

Input	Output
9 3 3 9 5 0 0 3 9 4 5 0 0 1 8 1 6 1 2 2 1 2	6 1 2 9 4 5 3

Problem H. Reward Desk

Time limit 3000 ms
Mem limit 262144 kB

Problem Statement

After a long day of coding at **PROCOM**, you're invited to redeem a set of unique digital tokens at the Rewards Desk. Each token carries a value that is considered **special**—that is, it is either a power of two or a factorial. For example, tokens with values like 1, 2, 4, 6, 8, or 24 are powerful because 1 = 1!, 4 = 2^2, and 6 = 3!; however, numbers such as 7, 10, or 18 are not.

You receive a gift card with a total amount of **n** dollars, and your task is to pay exactly **n** dollars by selecting a set of these tokens. Each token can be used at most once, and you must use tokens with distinct values. Determine the minimum number of tokens needed to reach exactly **n** dollars. If it's not possible to form such a combination, you must report that as well.

Input

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

A test case consists of only one line, containing one integer n ($1 \leq n \leq 10^{12}$).

Output

If the amount **n** cannot be exactly redeemed using a combination of **distinct** powerful tokens, print -1.

Otherwise, write a single positive integer — the minimum number of tokens required to exactly match the amount **n**.

Example

Input	Output
4 7 11 240 17179869184	2 3 4 1

In the first test case, a gift card amount of **7** dollars can be redeemed using tokens with values **1** and **6**. Although **7** itself is not a powerful token, combining the tokens **1** (which is 1! or 2^0) and **6** (which is 3!) exactly sums to 7. Thus, the minimum number of tokens required is **2**.

In the second test case, a gift card amount of **11** dollars can be redeemed by selecting tokens with values **1**, **4**, and **6**. Although this valid combination uses three tokens, it can be shown that no combination of fewer than three distinct tokens can sum to 11. Therefore, the minimum number of tokens required is **3**.

In the third test case, a gift card amount of **240** dollars can be redeemed by choosing tokens with values **24**, **32**, **64**, and **120**. Note that even though 240 could be seen as **120 + 120**, using the same token twice is not allowed since all tokens must be distinct. Hence, the minimum number of tokens required is **4**.

In the fourth test case, the gift card amount is **17179869184** dollars, which is exactly equal to 2^{34}. Since this value is itself a powerful token, it can be redeemed with a single token. Therefore, the minimum number of tokens required is **1**.

Problem I. PROCOM's Challenge

Time limit 1000 ms
Mem limit 524288 kB

Problem Statement

In a bold counter to the desperate promotional stunts of the Developer's Day, the **PROCOM** team presents a challenge that epitomizes ingenuity and resourcefulness.

You are given a target number **n**—a symbol of PROCOM's refined taste—and a requirement to select exactly **k** tokens. Each token is represented by one of 10 digits (from **0** to **9**), and an unlimited supply of each type is available.

The twist: the product of the digits on the selected tokens must equal **n**. The tokens are arranged in a sequence, and the cost of the arrangement is determined by reading the tokens as a single number. For example, a sequence with a token **8** followed by a token **3** yields a cost of **83**, while reversing the order gives a cost of **38**.

Your task is to help prove that true innovation beats flashiness by determining the **minimum cost** to assemble a valid sequence of exactly **k** tokens whose digits multiply to **n**. If no such sequence exists, output **-1**.

Let your solution be a testament to Procom's superior problem-solving and setting skills—a clear statement of how Procom's brilliance shines through.

Input

The input contains a single line of two integers n and k ($1 \le n \le 10^9$, $1 \le k \le 10^4$).

Output

Print a single line containing the minimum possible cost to form a valid sequence of exactly**k** tokens whose digits multiply to **n**. The cost is defined as the integer obtained by concatenating the tokens in order (for example, the sequence 8 followed by 3 yields a cost of 83). If it is impossible to form such a sequence, print **-1**.

Example 1

Input	Output
12 2	26

Example 2

Input	Output
34 2	-1

Problem J. Goodbye

Time limit 2000 ms
Mem limit 262144 kB

Problem Statement

After an unforgettable PROCOM event, Owais is finally heading home(with the CP trophy?). On his way, he finds a leftover meal in his bag. He wants to determine if it is still safe to eat.

Food Expiry Rules

- If Owais eats the food on or before its "best-by" date, he finds it **delicious**.
- If the food is expired but within his tolerance of X days, it is still **safe**.
- If the food is expired for more than X days, it is **dangerous** to eat.

Input

A single line containing three integers X, A, and B:

- X ($1 \le X \le 10^9$) — The number of days Owais can tolerate expired food.
- A ($1 \le A \le 10^9$) — The number of days before the "best-by" date Owais bought the food.
- B ($1 \le B \le 10^9$) — The number of days after buying the food that he eats it.

Output

Print one of the following:

- `delicious` if he eats the food on or before the "best-by" date.
- `safe` if the food is expired but within X days.
- `dangerous` if the food is expired for more than X days.

Example 1

Input	Output
4 3 6	safe

He ate the food three days after the "best-by" date. It was not delicious or harmful for him.

Example 2

Input	Output
6 5 1	delicious

He ate the food by the "best-by" date. It was delicious for him.

Example 3

Input	Output
3 7 12	dangerous

He ate the food five days after the "best-by" date. It was harmful for him.