Maratona de Programação da SBC 2023

This problem set is used in simultaneous contests:
Gran Premio de México 2023
Gran Premio de Centroamérica 2023
The 2023 ICPC Bolivia Preliminary Contest
Torneo Chileno de Programación 2023

September 2nd, 2023

Problems book

General Information

This problem set contains 13 problems; pages are numbered from 1 to 17, without considering this page. Please, verify your book is complete.

A) Program name
1) Solutions written in C/C++ and Python, the filename of the source code is not significant, can be any name.
2) Solutions written in Java, filename should be: *problem_code*`.java` where *problem_code* is the uppercase letter that identifies the problem. Remember in Java the main class name and the filename must be the same.
3) Solutions written in Kotlin, filename should be: *problem_code*`.kt` where *problem_code* is the uppercase letter that identifies the problem. Remember in Kotlin the main class name and the filename must be the same.

B) Input
1) The input must be read from *standard input*.
2) The input is described using a number of lines that depends on the problem. No extra data appear in the input.
3) When a line of data contains several values, they are separated by *single* spaces. No other spaces appear in the input.
4) Every line, including the last one, ends with an end-of-line mark.
5) The end of the input matches the end of file.

C) Output
1) The output must be written to *standard output*.
2) When a line of results contains several values, they must be separated by *single* spaces. No other spaces should appear in the output.
3) Every line, including the last one, must end with an end-of-line.

v1.0

Problem A

Amusement Park Adventure

Meet Carlitos, a spirited adventure enthusiast with an insatiable love for amusement parks. Despite his vibrant passion, Carlitos faces a unique challenge – his height. As he eagerly plans his weekend escapade, he realizes that his vertical limitations might hinder his amusement park experience. It's not just about choosing a park; it's about finding one where he can enjoy the thrill of the rides.

Picture the kaleidoscope of colors, the jubilant laughter, and the heart-pounding rush of the rides. Carlitos has always been drawn to the energy of amusement parks. With the weekend approaching, he pores over park brochures, studying the height requirements of each ride. His goal is to maximize his enjoyment, and that's where you come in.

Your task is to help Carlitos determine the number of rides he can enjoy at a specific park. By considering his height and the minimum height requirements of each ride, guide him in making the most of his amusement park adventure.

Input

The first line contains two integers, N and H ($1 \leq N \leq 6$ and $90 \leq H \leq 200$), representing the number of rides in a park and Carlitos' height in centimeters, respectively.

The second line contains the minimum heights $A_1, \ldots, A_N$ ($90 \leq A_i \leq 200$) of each ride in the park.

Output

Output a single line with an integer indicating the number of rides Carlitos can go on, that is, the number of rides for which Carlitos' height is at least as large as the minimum height required.

Input example 1	**Output example 1**
`1 100` `100`	`1`

Input example 2	**Output example 2**
`6 120` `200 90 100 123 120 169`	`3`

Problem B
Best Fair Shuffles

A Fair shuffle is a specific method of shuffling N distinct cards arranged horizontally from left to right. In a Fair shuffle, the cards are divided into two contiguous partitions which may have different card counts, and one of them may even have zero cards. Let's denote the left partition as L and the right partition as R.

Cards from the left partition (L) are then merged with the cards from the right partition (R) while maintaining the relative order between the cards of each partition.

Given the final permutation obtained after applying K Fair shuffles to an initial sequence of N non-repeating cards numbered from 1 to N, your task is to determine the minimum possible value of K.

For example, if we start with the sequence `1 2 3 4 5` and perform a Fair shuffle by partitioning it into L: `1 2` and R: `3 4 5`, cards from L and R can be merged in various ways:

- `1 3 2 4 5`
- `1 3 4 2 5`
- `3 4 5 1 2`
- `1 2 3 4 5`
- etc

Each arrangement represents a possible outcome of applying a single Fair shuffle. However, `1 3 2 5 4` is not a possible outcome as the relative order of cards `4` and `5` from R is not preserved.

Assume that the outcome of the first Fair shuffle is `1 3 2 4 5`. If we perform a second Fair shuffle on it, we could partition the sequence into L: `1 3 2 4` and R: `5` and merge it as `1 3 2 5 4`.

Input

The first line contains an integer $N (1 \leq N \leq 10^6)$, the number of cards in the deck. The second line contains a permutation of integer numbers from 1 to N describing the final arrangement of the cards.

Output

Output a single line with an integer K indicating the minimum number of Fair shuffles required to transform the initial sequence into the given one.

Input example 1	Output example 1
`5` `3 4 5 1 2`	`1`

Input example 2	Output example 2
`10` `1 6 5 2 10 3 4 8 7 9`	`3`

Input example 3	Output example 3
`5` `5 4 2 3 1`	`2`

Input example 4	**Output example 4**
`8` `1 6 2 3 7 8 4 5`	`1`

Input example 5	**Output example 5**
`5` `1 2 3 4 5`	`0`

Problem C
Challenging Hike

Finally, the exhausting exams are behind you, and the time has come to relish a well-deserved vacation away from the college hustle. With your bags neatly packed, you set out for an adventure to explore one of the most renowned mountains in your region – a destination you've been dreaming of visiting for a considerable period.

This majestic mountain boasts impressive dimensions and offers a plethora of hiking options. Comprising a total of N landmarks, each uniquely identified by a number ranging from 1 to N, the mountain presents a captivating network of $N-1$ interconnected paths. These pathways ensure that you can effortlessly traverse from any landmark to another, connecting the entire mountain in a seamless web.

Each landmark, denoted by the index i, possesses an associated score, aptly named v_i. This score reflects the expected number of likes your social media post of that landmark would attract. Brimming with enthusiasm, you've decided to infuse an extra layer of excitement into your hike by embracing the "photos hype challenge". In this challenge, your objective is nothing short of extraordinary: capture and post photos in a manner that guarantees each successive photo garners more likes than its predecessor.

The rules of this challenge dictate your journey to unfold as follows:

1. You commence your hike from landmark 1.
2. Progressing along the available paths, you navigate through the landmarks, always moving forward and never backtracking.
3. At each landmark you visit, you are presented with the choice of capturing a photo and posting it on your social media platform.

As your journey unfolds, you're curious about the potential outcomes. Specifically, for every landmark i, you seek to determine the maximum number of photos that can be posted if you initiate the hike from landmark 1 and conclude at landmark i (although you not necessarily took a photo of landmark i). The challenge is to strategically select the landmarks to capture photos, ensuring that each photo outshines its predecessor in terms of expected likes.

Your task is to calculate this maximum number of photos for each landmark. Can you rise to the challenge and uncover the most captivating and likable path through this mountain?

Input

The first line contains an integer N ($1 \le N \le 10^5$), representing the number of landmarks in the mountain. The second line contains $N-1$ integers $p_2, p_3, \ldots, p_N$ ($1 \le p_i \le N$), where p_i represents that there is a path between the landmarks i and p_i. The third line contains N integers $v_1, v_2, \ldots, v_N$ ($1 \le v_i \le 10^9$), where v_i represents the expected number of likes a photo of the i-th landmark would get.

Output

Output a single line with $N-1$ integers, where the i-th integer represents the maximum number of photos you can post if you start the hike on landmark 1 and finish on landmark $(i+1)$.

Input example 1	**Output example 1**
5 1 1 3 3 5 7 7 6 8	2 2 2 3

Input example 2	**Output example 2**

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

Problem D
Detour

In the city of Nlogonia, the mayor is taking action on his promises to revitalize the city's road infrastructure. However, the road renewal process renders certain roads temporarily impassable, requiring the establishment of detours to ensure uninterrupted traffic flow.

Each road segment connects two crossroads in the city, has a positive length and can be traversed in both directions. A detour is a designated alternative route that temporarily replaces a road segment under renewal. Specifically, when the road connecting crossroads U and V is impassable, the detour must be a sequence of roads that originates at U and terminates at V, while intentionally avoiding the direct road between U and V. The goal is to find the shortest detour for each road segment to minimize disruptions while road improvements take place.

As the Intern at the Center for Pavement and Cars, your responsibility is to support the mayor's initiative by calculating the length of the shortest detour for each road segment within the city.

Input

The first line contains two integers, N and M ($1 \leq N \leq 300$), representing the number of crossroads in the city and the number of road segments. Each of the following M lines contains three integers, U, V, and L ($1 \leq U \leq N$, $1 \leq V \leq N$, $U \neq V$, $1 \leq L \leq 10^6$), representing a two-way road segment of length L that connects crossroads U and V. No road segment is duplicated.

Output

Output M lines, each line containing an integer. The integer on the i-th line represents the shortest detour length for the i-th road segment or `-1` if there is no possible detour. The answer for each road segment should be given in the same order as road segments are described in the input.

Input example 1	**Output example 1**
4 5 1 2 4 1 3 8 2 3 4 4 1 2 3 4 3	9 5 9 11 10

Input example 2	**Output example 2**
2 1 1 2 1	-1

Problem E
Extracting Pollen

Spring has arrived, ushering in a season of hard work at the Swarm of Bees Company (SBC). With the blooming of N beautiful flowers in the garden, each flower boasts a certain quantity of pollen grains. The SBC enforces strict rules to keep the bees industrious in their pollen collection.

1. The first rule pertains to the amount of pollen grains collected: when a bee visits a flower, it must gather the sum of the digits in its current pollen quantity. For instance, if a bee visits a flower with 123 pollen grains, it must collect $1 + 2 + 3 = 6$ grains, leaving the flower with $123 - 6 = 117$ grains. Similarly, if the flower holds 201 grains, the bee must gather $2+0+1 = 3$ grains, leaving 198 grains remaining.

2. All bees must form a queue at the start of the day; the bee at the front of the queue must collect pollen from one of the flowers with the largest amount pollen. If a bee visits a flower with 0 grains of pollen, it collects zero grains. After collecting pollen from a flower, the bee ends its shift and returns to the hive.

Gertrude finds these rules bewildering and seeks help to determine the pollen amount she must collect when it's her turn. Getrude has amazingly sharp sight and noticed that she is currently the K-th bee in the SBC-defined order.

Input

The first line contains two integers N ($1 \leq N \leq 10^6$) and K ($1 \leq K \leq 10^9$), representing the number of flowers and Gertrude's position in the bee line, respectively. The second line contains N integers, where the i-th integer F_i ($1 \leq F_i \leq 10^6$ for $1 \leq i \leq N$) denotes the initial quantity of pollen grains of the i-th flower.

Output

Output a single integer Q representing the amount of pollen Gertrude will collect.

Input example 1	**Output example 1**
`5 3` `22 15 7 2 1`	`6`

Explanation of sample 1:

The first bee will collect pollen from the first flower, leaving it with $22 - (2 + 2) = 18$ grains remaining. The second bee will also collect from the first flower, leaving it with $18 - (1 + 8) = 9$ grains remaining. Finally, Gertrude, the third bee in line, will collect pollen from the second flower, collecting a total of $1 + 5 = 6$ pollen, which will be the answer for this test case.

Input example 2	**Output example 2**
`3 10` `21 21 21`	`0`

Input example 3	**Output example 3**
`3 9` `21 21 21`	`9`

Problem F
Fatigue-Fighting Vacation

William is planning his upcoming vacations. A recurring problem when he takes vacations is the need to deal with fatigue. Some days he does not enjoy much, as after several activities, the fatigue becomes greater than what he can deal with.

This time, William had an idea. He will estimate the impact on his disposition for each of the tourism activities. He noticed that some of the vacation activities, such as sports and hikes, are tiring and consume his disposition, while other activities, such as theatre plays and musicals, are invigorating and restore his disposition.

More precisely, William starts with D units of disposition and separates his activities into two groups: C tiring activities and R invigorating activities. Each tiring activity requires a certain amount of disposition and consumes that amount of disposition when performed. Each invigorating activity provides him with a certain amount of disposition when performed. Additionally, he arranges the activities in each group according to his preferences, as there are activities he is more willing to perform. It is important to note that activities from both sets can be interleaved, but William will never do an activity from one group without having done all the previous activities from that group, as this would not conform with his preferences.

Throughout his vacation, when deciding which activity to do next, he will choose the first untaken tiring activity, provided he has enough disposition to do it. Otherwise, he will perform the next untaken invigorating activity, if any remains, replenishing a certain amount of disposition. Naturally, if there are no remaining tiring activities at any point, he can simply perform all the remaining invigorating activities.

Considering this process, William has asked for your help to determine how many activities (including both tiring and invigorating ones) he will be able to perform.

Input

The first line of input contains three integer numbers, D, C, and, R, representing, respectively, the initial amount of disposition, the number of tiring activities, and the number of invigorating activities ($1 \leq D \leq 10^5$, $1 \leq C \leq 10^4$, and $1 \leq R \leq 10^4$). Each of the next C lines contains an integer number C_i ($1 \leq C_i \leq 10^5$ for $1 \leq i \leq C$), representing the required amount of disposition for a tiring activity, in order of preference. Finally, each of the next R lines contains an integer number R_i ($1 \leq R_i \leq 10^5$ for $1 \leq i \leq R$), representing the provided amount of disposition for an invigorating activity, in order of preference.

Output

Print a single line with a single integer number, the total number of activities (including both tiring and invigorating ones) that William will be able to perform.

Input example 1	**Output example 1**
40 3 3 30 20 10 5 5 5	5

Input example 2	Output example 2
40 2 2 60 80 5 10	2

Input example 3	Output example 3
100 3 1 60 60 50 10	2

Problem G
Great Treaty of Byteland

The Great War of Byteland is over. The remaining kingdoms are now discussing the Division Treaty, which will split all the land in the world among them. It will refer not only to the known world, but also to any territories yet to be discovered or inhabited, including land or sea. We can assume that the world is an infinite flat plane.

Each kingdom in the continent of Byteland has a single capital, and the Division Treaty will be based on their locations: it declares that each piece of land belongs to the kingdom whose capital is the nearest in a bird's flight (or in a straight line). In other words: wherever you are in the world, if C is the single nearest capital to you, you will be in the territory of C's kingdom. If there is a tie between the distances of two or more capitals, that place will be in the border between their kingdoms.

Under this treaty, some kingdoms may end up enclosed between others, while other kingdoms may end up with unlimited territory. Therefore, some monarchs are contesting the treaty. To inform this discussion, they demand your help. Given the location coordinates of each capital in the continent of Byteland, you must find out which kingdoms would have infinite territories under the Division Treaty.

Input

The first input line contains a single integer N ($2 \leq N \leq 10^5$), the number of kingdoms. Each kingdom is identified by an unique integer between 1 and N. Each of the N following lines contains two integers X and Y ($0 \leq X, Y \leq 10^4$), the 2D coordinates of the location of a kingdom's capital. The capitals are given in increasing order of kingdom identifier, no two capitals have the same location, and you can assume that every capital has negligible size.

Output

Print a single line with a list of space-separated integers in increasing order: the identifiers of the kingdoms that would have infinite territories under the described Division Treaty. It's guaranteed that there is always at least one such kingdom.

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

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

Problem H
Honest Worker

Rafael lives in an ideal egalitarian society: all jobs offer identical payment rates and you only have to work at a job that you find personally fulfilling. Unfortunately, even such a utopia can't account for the fact that Rafael is not very skilled at anything.

To compensate for this, Rafael turns to contractor work, where potential employers lack the time to realize how unqualified for the job he is. Even then, given his lack of qualifications for these contractor positions, he must expend money on acquiring fake cover letters in order to even get hired.

Rafael has a choice of N contractor jobs available to him. The i-th job starts at day ℓ_i, ends at the end of day r_i and pays exactly S gold coins a day. Rafael is a really bad worker, and cannot work two jobs at the same time; moreover, he can only start job i at day ℓ_i but, once hired, he can choose to quit at the end of any day, keeping the money from the days he worked (including the last one), and being able to start another job from the next day on, but not on the same day. Additionally, Rafael knows that it would cost c_i gold coins to buy a fake cover letter for job i. Aware of his inabilities and the need for fake cover letters, Rafael always has enough savings to pay for any number of cover letters he might need, even before working any of the N jobs.

Given the description of the available jobs, what is the maximum profit that Rafael can attain, factoring in the expenses for the required fake cover letters?

Input

The first line contains two integers N ($1 \leq N \leq 10^6$) and S ($1 \leq S \leq 10^9$), representing the number of jobs available and what is their daily pay.

Each of the following N lines contains three integers: ℓ_i, r_i and c_i ($1 \leq \ell_i \leq r_i \leq 10^9$, $1 \leq c_i \leq 10^9$), representing the start date, end date, and cost of obtaining a fake cover letter for job i, respectively.

Output

Output a single line with an integer indicating the amount of money Rafael can make after all the jobs are over.

Input example 1	Output example 1
3 3 1 5 10 2 10 4 5 15 1	37

Explanation of sample 1:

It is optimal to start the second job, then switch to the third one, for a total of $14 \times 3 - 4 - 1 = 37$ gold coins.

Input example 2	Output example 2
3 5 1 1 3 2 3 4 3 3 1	8

Input example 3	**Output example 3**
`1 1000` `1 1 654`	`346`

Explanation of sample 3:

Even though he has to take money from savings to buy the fake cover letter he can still make some profit.

Input example 4	**Output example 4**
`1 5` `1 3 20`	`0`

Explanation of sample 4:

Not worth spending money on the fake cover letter for this job.

Problem I
Investigating Zeroes and Ones

You find yourself in a mysterious binary world, where an array of N binary digits awaits your scrutiny. Each digit is either a zero or a one, creating a unique pattern across the landscape. Your quest is to uncover the hidden patterns of this binary realm by unraveling the significance of subarrays with an odd number of ones.

The array of digits is denoted as $b_1, b_2, \ldots, b_N$. Your task is to embark on a journey to discover the enigmatic subarrays – segments of consecutive digits – and determine the count of subarrays that harbor an odd number of ones.

As you traverse this binary landscape, remember that a subarray is defined by its starting and ending digits. For instance, in the sequence $[b_1, b_2, b_3]$, subarrays include $[b_1]$, $[b_2]$, $[b_3]$, $[b_1, b_2]$, $[b_2, b_3]$, and $[b_1, b_2, b_3]$.

Your mission is to design an algorithm that determines the total number of subarrays containing an odd number of ones within this binary sequence. Please don't forget that the answer might not fit in a 32-bits integer.

Input

The first line contains an integer N ($1 \leq N \leq 10^5$) representing the length of the binary sequence.

The second line contains N binary digits $b_1, b_2, \ldots, b_N$ ($b_i \in \{0, 1\}$) representing the elements of the sequence.

Output

Output a single line with an integer representing the count of subarrays in the sequence that hold an odd number of ones.

Input example 1	**Output example 1**
3 0 1 0	4

Input example 2	**Output example 2**
10 1 0 0 1 1 0 1 1 1 0	30

Problem J
Jumping to Victory

The stage is set for the Olympic Games super volleyball finals, and tensions are high! Ricardão, the coach of a competing team, is meticulously positioning his players for a strategic advantage. However, a lingering concern plagues him: the possibility that some areas of the court might be inadequately covered by his players. This could lead to exhausting lateral jumps to reach the ball.

In this high-stakes match, your task is to assist Coach Ricardão by analyzing the court layout and the positions of the players. Calculate the maximum jump distance required by any player to intercept the ball, ensuring that no point on the court remains vulnerable. Professional super volleyball teams are very good, and can easily determine which is the closest player to where the ball will fall; only this player will try to reach it, all other will remain still.

The court is defined by an axis-aligned rectangle $\mathcal{R}$ defined by its four vertices and the players modelled as a set of N points inside this rectangle. Your task is to determine the smallest distance d such that any point on the court can be reached by at least one player, if a player can jump a distance of up to d in any direction. Remember: the border of the court is also part of the court and must be covered by the players!

Input

The first four lines contains the four vertices of $\mathcal{R}$; that is, the i-th line contains two integers x_i and y_i ($-10^5 \leq x_i, y_i \leq 10^5$), representing the coordinates of the i-th vertex of $\mathcal{R}$. The fifth line contains a single integer N ($1 \leq N \leq 10^5$), representing the number of players. Each of the following N lines contains the players coordinates, where the i-th line contains two integers x_i and y_i ($-10^5 \leq x_i, y_i \leq 10^5$), representing the coordinates of the i-th player. It's guaranteed that every player is inside the court.

Output

Output a single line with the least d that is enough to guard the whole court. The output will be considered correct if it is within an absolute or relative error of 10^{-5} of the correct answer.

Input example 1

```
-1 -1
1 -1
1 1
-1 1
1
0 0
```

Output example 1

```
1.41421356237
```

Input example 2

```
1 -1
-1 3
1 3
-1 -1
3
0 0
1 3
-1 3
```

Output example 2

```
1.666666666667
```

Problem K
K for More, K for Less

The life of those who study computer science is not always as easy as it seems. Some days you might be implementing a revolutionary algorithm, and other days you find yourself reading the same book for the tenth time. But at all times we are looking for the same thing: to optimize and automate tasks. In this case, a teacher needs your help to guide his students to the next exam. In the professor's opinion, it is not easy to decide how much time students should spend studying theoretical topics and how much time they should spend implementing algorithms.

This is not the first time that the professor has taught this subject, so the amount of data available is so large that he was able to create two polynomials to describe the final performance of each student. If the student spends x units of his time studying theory, his grade will increase by $t(x)$. If the student spends x units of his time implementing algorithms, his grade will increase by $p(x)$. In such a way that the student who spends the same amount x of time in each of the areas will have a final grade of $t(x) + p(x)$.

Recently one of the students has been standing out unpredictably. He does not hide his technique from anyone: "I study theory a lot more than practice!". The professor believes this is a lie and, to confirm his suspicion, he decided to estimate students grades if they always studied more theory than practice (or more practice than theory). Can you compute the polynomial $q(x) = t(x+K)+p(x-K)$? It will be able to describe every student grade if they change their study strategy.

Input

The input consists of three lines. The first line contains two integers: N, representing the degree of the polynomials t and p ($1 \leq N \leq 10^5$) and K ($-10^5 \leq K \leq 10^5$). The second line contains the $N + 1$ coefficients of t, and the third line contains the $N + 1$ coefficients of p. The coefficients are given in increasing order of degree, with the last coefficient in the row corresponding to the term with degree N, all of which are non-negative with a maximum value of 10^6.

Output

Your program should print a line with $N + 1$ integers, the coefficients of the polynomial $q(x)$ in increasing order of degree modulo 998244353.

Input example 1	Output example 1
1 2 1 2 0 1	3 3

Input example 2	Output example 2
2 0 1 2 3 4 5 6	5 7 9

Input example 3	Output example 3
2 -1 3 3 3 1 0 0	4 998244350 3

Problem L
Lexicographical Challenge

In the picturesque village of Lexicoville there lived two friends, Lily and Ethan. One day, a mysterious letter arrived at their house, sealed with a bunch of beatiful insignia full of sparkle and enchantment. Inside the envelope, they found a riddle too complex, even for the wisest minds of their village.

In this riddle, they were given an integer K and a string S containing only lower-case letters, which could be changed according to a curious rule. At each moment, the villagers have the freedom to choose an index i; then, magically, the characters S_i and S_{i+K} will be swapped! The riddle will be solved when the lexicographically minimum string, using only the allowed operation, is found.

The village was filled with curiosity and excitement with this riddle, but none more so than Lily and Ethan. The friends, always craving adventure, decided to dive headfirst into this challenge. However, as they observed the string, they noticed that the road to success was paved with countless possible swaps.

With the riddle's words echoing in their minds, they wondered: How could they navigate this web of possibilities to unveil the lexicographically minimum string? Every swap was like turning a page in a magical book, revealing new secrets and mysteries.

A long time has passed, but neither Lily nor Ethan have been able to solve the riddle. Can you help them?

Input

The first line contains the string S ($1 \leq |S| \leq 10^5$). The seconds line contains the integer K ($1 \leq K < |S|$).

Output

Output a single line with the lexicographically earliest string that can be obtained.

Input example 1	**Output example 1**
`zaaab` `4`	`baaaz`

Input example 2	**Output example 2**
`njoab` `2`	`banjo`

Problem M
Maximizing Flight Efficiency

In the kingdom of Quadradonia, the king wants to review the prices of flights. For this purpose, he asked his accountant for a table with proposals for new prices.

However, the king studied at the Institute of Computing and Programming of Chapecó (ICPC) and has sufficient knowledge to demand coherence in the table. The table is considered *coherent* if no multi-stop route is cheaper than a direct flight.

Once the coherence of the table has been verified, the king would like to decrease the number of direct flights, without increasing the costs of the trips.

Your problem is to verify the coherence of the table and, if it is coherent, inform the king how many direct flights can be eliminated without increasing the cost of any trip.

Input

The first line of input contains an integer N ($1 \leq N \leq 100$), the number of cities in Quadradonia served by flights. Following there are N more lines, $L_1, L_2, \ldots, L_N$. Line L_i contains N integers, $C_{i1}, C_{i2}, \ldots, C_{iN}$, where C_{ij} is the cost of the direct flight between cities i and j.

The cost for an outbound and a return flight between two cities is always the same, meaning $C_{ij} = C_{ji}$ for all pairs i, j where $1 \leq i \leq N$ and $1 \leq j \leq N$. When $i = j$, $C_{ij} = 0$. When $i \neq j$, $1 \leq C_{ij} \leq 10^3$.

Output

Print a single line containing an integer. If the table is incoherent, the integer should be `-1`. If the table is coherent, the integer should be equal to the maximum number of direct flights that can be removed without increasing the costs of the trips for the passengers.

Input example 1	**Output example 1**
3 0 1 2 1 0 1 2 1 0	1

Input example 2	**Output example 2**
3 0 2 2 2 0 2 2 2 0	0

Input example 3	**Output example 3**
3 0 2 9 2 0 2 9 2 0	-1