A. Eating Soup

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

The three friends, Kuro, Shiro, and Katie, met up again! It's time for a party...

What the cats do when they unite? Right, they have a party. Since they wanted to have as much fun as possible, they invited all their friends. Now n cats are at the party, sitting in a circle and eating soup. The rules are simple: anyone having finished their soup leaves the circle.

Katie suddenly notices that whenever a cat leaves, the place where she was sitting becomes an empty space, which means the circle is divided into smaller continuous groups of cats sitting next to each other. At the moment Katie observes, there are m cats who left the circle. This raises a question for Katie: what is the maximum possible number of groups the circle is divided into at the moment?

Could you help her with this curiosity?

You can see the examples and their descriptions with pictures in the "Note" section.

Input

The only line contains two integers n and m ($2 \leq n \leq 1000$, $0 \leq m \leq n$) — the initial number of cats at the party and the number of cats who left the circle at the moment Katie observes, respectively.

Output

Print a single integer — the maximum number of groups of cats at the moment Katie observes.

Examples

input
```
7 4
```
output
```
3
```

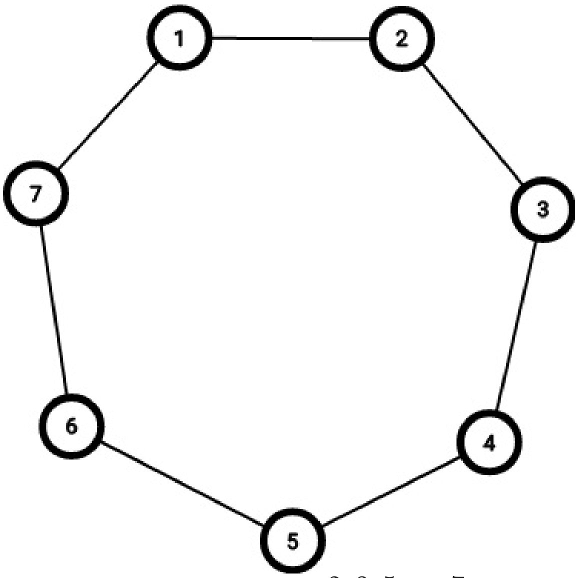
input
```
6 2
```
output
```
2
```

input
```
3 0
```
output
```
1
```

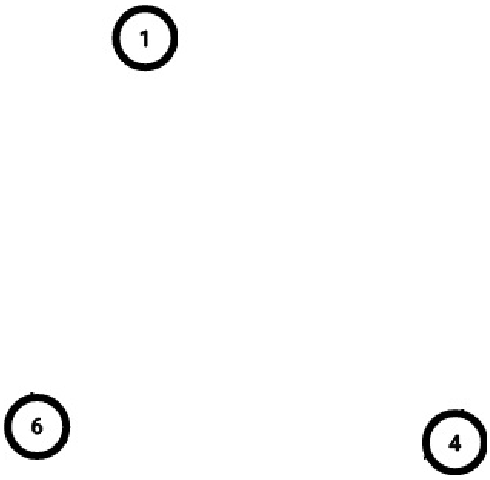
input
```
2 2
```
output
```
0
```

Note

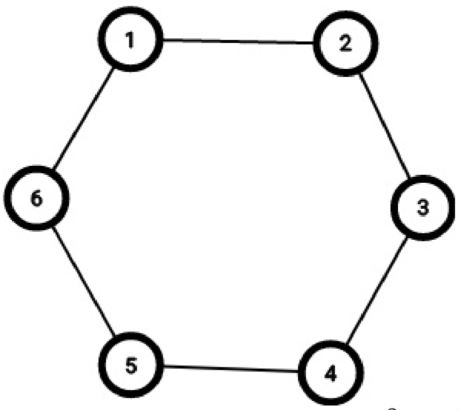
In the first example, originally there are 7 cats sitting as shown below, creating a single group:



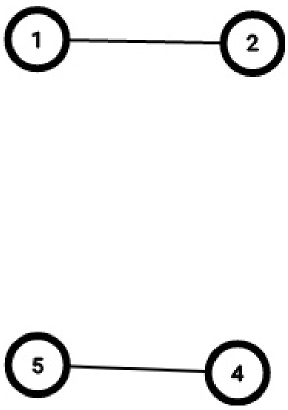
At the observed moment, 4 cats have left the table. Suppose the cats 2, 3, 5 and 7 have left, then there are 3 groups remaining. It is possible to show that it is the maximum possible number of groups remaining.



In the second example, there are 6 cats sitting as shown below:



At the observed moment, 2 cats have left the table. Suppose the cats numbered 3 and 6 left, then there will be 2 groups remaining ($\{1,2\}$ and $\{4,5\}$). It is impossible to have more than 2 groups of cats remaining.



In the third example, no cats have left, so there is 1 group consisting of all cats.

In the fourth example, all cats have left the circle, so there are 0 groups.

B1. Cat Party (Easy Edition)

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

This problem is same as the next one, but has smaller constraints.

Shiro's just moved to the new house. She wants to invite all friends of her to the house so they can play monopoly. However, her house is too small, so she can only invite one friend at a time.

For each of the n days since the day Shiro moved to the new house, there will be exactly one cat coming to the Shiro's house. The cat coming in the i-th day has a ribbon with color u_i. Shiro wants to know the largest number x, such that if we consider the streak of the first x days, it is possible to remove **exactly one** day from this streak so that every ribbon color *that has appeared* among the remaining $x - 1$ will have the same number of occurrences.

For example, consider the following sequence of u_i: $[2, 2, 1, 1, 5, 4, 4, 5]$. Then $x = 7$ makes a streak, since if we remove the leftmost $u_i = 5$, each ribbon color will appear exactly twice in the prefix of $x - 1$ days. Note that $x = 8$ doesn't form a streak, since you must remove exactly one day.

Since Shiro is just a cat, she is not very good at counting and needs your help finding the longest streak.

Input

The first line contains a single integer n ($1 \leq n \leq 10^5$) — the total number of days.

The second line contains n integers $u_1, u_2, \ldots, u_n$ ($1 \leq u_i \leq 10$) — the colors of the ribbons the cats wear.

Output

Print a single integer x — the largest possible streak of days.

Examples

input

```
13
1 1 1 2 2 2 3 3 3 4 4 4 5
```

output

```
13
```

input

```
5
10 2 5 4 1
```

output

```
5
```

input

```
1
10
```

output

```
1
```

input

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

output

```
6
```

input

```
6
1 1 1 2 2 2
```

output

```
5
```

Note

In the first example, we can choose the longest streak of 13 days, since upon removing the last day out of the streak, all of the remaining colors 1, 2, 3, and 4 will have the same number of occurrences of 3. Note that the streak can also be 10 days (by removing the 10-th day from this streak) but we are interested in the longest streak.

In the fourth example, if we take the streak of the first 6 days, we can remove the third day from this streak then all of the remaining colors 1, 2, 3, 4 and 5 will occur exactly once.

B2. Cat Party (Hard Edition)

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

This problem is same as the previous one, but has larger constraints.

Shiro's just moved to the new house. She wants to invite all friends of her to the house so they can play monopoly. However, her house is too small, so she can only invite one friend at a time.

For each of the n days since the day Shiro moved to the new house, there will be exactly one cat coming to the Shiro's house. The cat coming in the i-th day has a ribbon with color u_i. Shiro wants to know the largest number x, such that if we consider the streak of the first x days, it is possible to remove **exactly one** day from this streak so that every ribbon color *that has appeared* among the remaining $x-1$ will have the same number of occurrences.

For example, consider the following sequence of u_i: $[2, 2, 1, 1, 5, 4, 4, 5]$. Then $x = 7$ makes a streak, since if we remove the leftmost $u_i = 5$, each ribbon color will appear exactly twice in the prefix of $x-1$ days. Note that $x = 8$ doesn't form a streak, since you must remove exactly one day.

Since Shiro is just a cat, she is not very good at counting and needs your help finding the longest streak.

Input

The first line contains a single integer n ($1 \le n \le 10^5$) — the total number of days.

The second line contains n integers $u_1, u_2, \ldots, u_n$ ($1 \le u_i \le 10^5$) — the colors of the ribbons the cats wear.

Output

Print a single integer x — the largest possible streak of days.

Examples

input
```
13
1 1 1 2 2 2 3 3 3 4 4 4 5
```
output
```
13
```

input
```
5
10 100 20 200 1
```
output
```
5
```

input
```
1
100000
```
output
```
1
```

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

input
```
6
1 1 1 2 2 2
```
output
```
5
```

Note

In the first example, we can choose the longest streak of 13 days, since upon removing the last day out of the streak, all of the remaining colors 1, 2, 3, and 4 will have the same number of occurrences of 3. Note that the streak can also be 10 days (by removing the 10-th day from this streak) but we are interested in the longest streak.

In the fourth example, if we take the streak of the first 6 days, we can remove the third day from this streak then all of the remaining colors 1, 2, 3, 4 and 5 will occur exactly once.

C1. Power Transmission (Easy Edition)

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

This problem is same as the next one, but has smaller constraints.

It was a Sunday morning when the three friends Selena, Shiro and Katie decided to have a trip to the nearby power station *(do not try this at home)*. After arriving at the power station, the cats got impressed with a large power transmission system consisting of many chimneys, electric poles, and wires. Since they are cats, they found those things gigantic.

At the entrance of the station, there is a map describing the complicated wiring system. Selena is the best at math among three friends. He decided to draw the map on the Cartesian plane. Each pole is now a point at some coordinates (x_i, y_i). Since every pole is different, all of the points representing these poles are distinct. Also, every two poles are connected with each other by wires. A wire is a straight line on the plane **infinite in both directions**. If there are more than two poles lying on the same line, they are connected by a single common wire.

Selena thinks, that whenever two different electric wires intersect, they may interfere with each other and cause damage. So he wonders, how many pairs are intersecting? Could you help him with this problem?

Input

The first line contains a single integer n ($2 \le n \le 50$) — the number of electric poles.

Each of the following n lines contains two integers x_i, y_i ($-10^4 \le x_i, y_i \le 10^4$) — the coordinates of the poles.

It is guaranteed that all of these n points are distinct.

Output

Print a single integer — the number of pairs of wires that are intersecting.

Examples

input

```
4
0 0
1 1
0 3
1 2
```

output

```
14
```

input

```
4
0 0
0 2
0 4
2 0
```

output

```
6
```

input

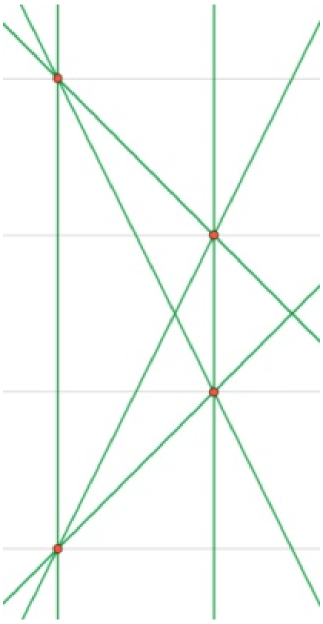
```
3
-1 -1
1 0
3 1
```

output

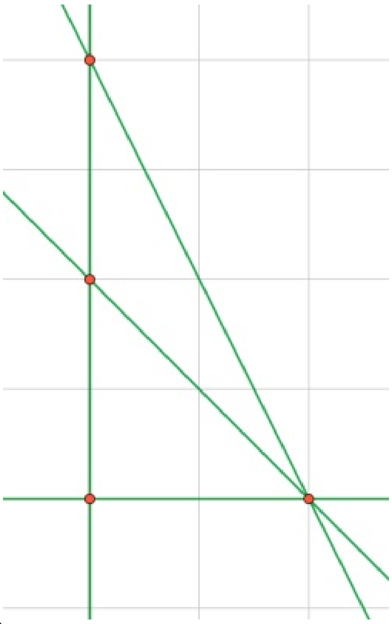
```
0
```

Note

In the first example:

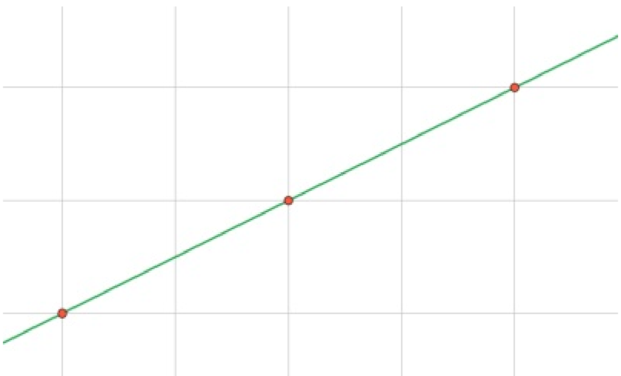


In the second example:



Note that the three poles $(0,0)$, $(0,2)$ and $(0,4)$ are connected by a single wire.

In the third example:



C2. Power Transmission (Hard Edition)

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

This problem is same as the previous one, but has larger constraints.

It was a Sunday morning when the three friends Selena, Shiro and Katie decided to have a trip to the nearby power station *(do not try this at home)*. After arriving at the power station, the cats got impressed with a large power transmission system consisting of many chimneys, electric poles, and wires. Since they are cats, they found those things gigantic.

At the entrance of the station, there is a map describing the complicated wiring system. Selena is the best at math among three friends. He decided to draw the map on the Cartesian plane. Each pole is now a point at some coordinates (x_i, y_i). Since every pole is different, all of the points representing these poles are distinct. Also, every two poles are connected with each other by wires. A wire is a straight line on the plane **infinite in both directions**. If there are more than two poles lying on the same line, they are connected by a single common wire.

Selena thinks, that whenever two different electric wires intersect, they may interfere with each other and cause damage. So he wonders, how many pairs are intersecting? Could you help him with this problem?

Input

The first line contains a single integer n ($2 \le n \le 1000$) — the number of electric poles.

Each of the following n lines contains two integers x_i, y_i ($-10^4 \le x_i, y_i \le 10^4$) — the coordinates of the poles.

It is guaranteed that all of these n points are distinct.

Output

Print a single integer — the number of pairs of wires that are intersecting.

Examples

input

```
4
0 0
1 1
0 3
1 2
```

output

```
14
```

input

```
4
0 0
0 2
0 4
2 0
```

output

```
6
```

input

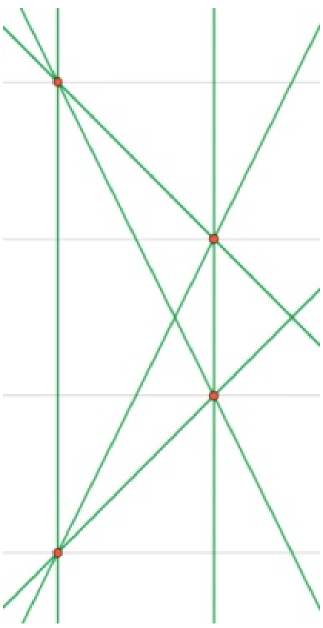
```
3
-1 -1
1 0
3 1
```

output

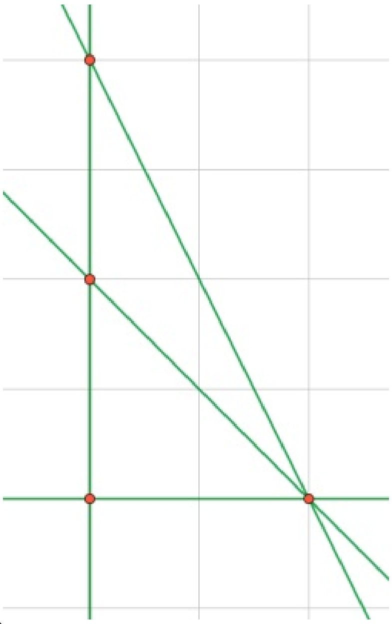
```
0
```

Note

In the first example:

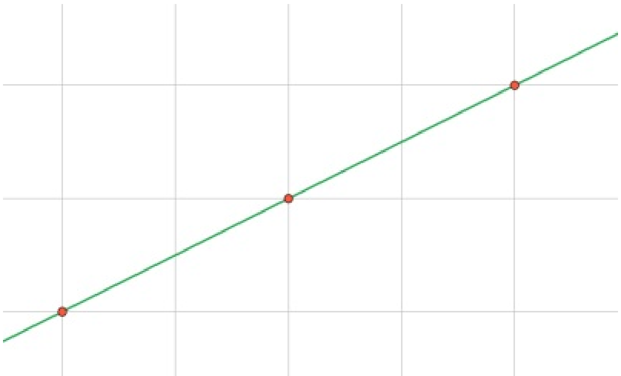


In the second example:



Note that the three poles $(0,0)$, $(0,2)$ and $(0,4)$ are connected by a single wire.

In the third example:



D. Mysterious Code

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

During a normal walk in the forest, Katie has stumbled upon a mysterious code! However, the mysterious code had some characters unreadable. She has written down this code as a string c consisting of lowercase English characters and asterisks ("*"), where each of the asterisks denotes an unreadable character. Excited with her discovery, Katie has decided to recover the unreadable characters by replacing each asterisk with arbitrary lowercase English letter (different asterisks might be replaced with different letters).

Katie has a favorite string s and a not-so-favorite string t and she would love to recover the mysterious code so that it has as many occurrences of s as possible and as little occurrences of t as possible. Formally, let's denote $f(x, y)$ as the number of occurrences of y in x (for example, $f(aababa, ab) = 2$). Katie wants to recover the code c' conforming to the original c, such that $f(c', s) - f(c', t)$ is largest possible. However, Katie is not very good at recovering codes in general, so she would like you to help her out.

Input

The first line contains string c ($1 \leq |c| \leq 1000$) — the mysterious code . It is guaranteed that c consists of lowercase English characters and asterisks "*" only.

The second and third line contain strings s and t respectively ($1 \leq |s|, |t| \leq 50$, $s \neq t$). It is guaranteed that s and t consist of lowercase English characters only.

Output

Print a single integer — the largest possible value of $f(c', s) - f(c', t)$ of the recovered code.

Examples

input

```
*****
katie
shiro
```

output

```
1
```

input

```
caat
caat
a
```

output

```
-1
```

input

```
*a*
bba
b
```

output

```
0
```

input

```
***
cc
z
```

output

```
2
```

Note

In the first example, for c' equal to "katie" $f(c', s) = 1$ and $f(c', t) = 0$, which makes $f(c', s) - f(c', t) = 1$ which is the largest possible.

In the second example, the only c' conforming to the given c is "caat". The corresponding $f(c', s) - f(c', t) = 1 - 2 = -1$.

In the third example, there are multiple ways to recover the code such that $f(c', s) - f(c', t)$ is largest possible, for example "aaa", "aac", or even "zaz". The value of $f(c', s) - f(c', t) = 0$ for all of these recovered codes.

In the fourth example, the optimal recovered code c' would be "ccc". The corresponding $f(c', s) - f(c', t) = 2$.

E. Magical Permutation

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

Kuro has just learned about permutations and he is really excited to create a new permutation type. He has chosen n distinct positive integers and put all of them in a set S. Now he defines a magical permutation to be:

- A permutation of integers from 0 to $2^x - 1$, where x is a non-negative integer.
- The bitwise xor of any two consecutive elements in the permutation is an element in S.

Since Kuro is really excited about magical permutations, he wants to create the longest magical permutation possible. In other words, he wants to find the largest non-negative integer x such that there is a magical permutation of integers from 0 to $2^x - 1$. Since he is a newbie in the subject, he wants you to help him find this value of x and also the magical permutation for that x.

Input

The first line contains the integer n ($1 \leq n \leq 2 \cdot 10^5$) — the number of elements in the set S.

The next line contains n distinct integers $S_1, S_2, \ldots, S_n$ ($1 \leq S_i \leq 2 \cdot 10^5$) — the elements in the set S.

Output

In the first line print the largest non-negative integer x, such that there is a magical permutation of integers from 0 to $2^x - 1$.

Then print 2^x integers describing a magical permutation of integers from 0 to $2^x - 1$. If there are multiple such magical permutations, print any of them.

Examples

input

```
3
1 2 3
```

output

```
2
0 1 3 2
```

input

```
2
2 3
```

output

```
2
0 2 1 3
```

input

```
4
1 2 3 4
```

output

```
3
0 1 3 2 6 7 5 4
```

input

```
2
2 4
```

output

```
0
0
```

input

```
1
20
```

output

```
0
0
```

input

```
1
1
```

output

```
1
0 1
```

Note

In the first example, $0, 1, 3, 2$ is a magical permutation since:

- $0 \oplus 1 = 1 \in S$
- $1 \oplus 3 = 2 \in S$
- $3 \oplus 2 = 1 \in S$

Where $\oplus$ denotes bitwise xor operation.

F. Indecisive Taxi Fee

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

In the city of Capypaland where Kuro and Shiro resides, there are n towns numbered from 1 to n and there are m bidirectional roads numbered from 1 to m connecting them. The i-th road connects towns u_i and v_i. Since traveling between the towns is quite difficult, the taxi industry is really popular here. To survive the harsh competition, each taxi company has to find a distinctive trait for their customers.

Kuro is the owner of a taxi company. He has decided to introduce a new fee model for his taxi brand, where the fee for each ride is not calculated based on the trip length, but on the sum of the prices of the roads traveled. The price for each of the m roads has been decided by Kuro himself.

As of now, the price for the road i is w_i and hence the fee for a taxi ride traveling through roads $e_1, e_2, \ldots, e_k$ is $\sum_{i=1}^{k} w_{e_i}$.

However, Kuro himself is an indecisive person, so he has drafted q plans to change the road price. Each of the plans will be based on the original prices w_i, except for a single road t_j, the price of which is changed to x_j. Note, that the plans are independent of each other.

Shiro is a regular customer of the Kuro's taxi brand since she uses the taxi to travel from town 1 to town n every day. Since she's so a regular customer, Kuro decided to show her all his q plans before publishing them to the public. Now, Shiro wants to know the lowest fee she must pay to travel from the town 1 to the town n for each Kuro's plan.

Input

The first line contains three integers n, m and q ($2 \le n \le 2 \cdot 10^5$, $1 \le m, q \le 2 \cdot 10^5$) — the number of towns, the number of roads, and the number of plans that Kuro has drafted respectively.

The i-th of the next m contains three integers u_i, v_i and w_i ($1 \le u_i, v_i \le n, 1 \le w_i \le 10^9$, $u_i \ne v_i$) — two endpoints and the original price of the i-th road.

It is guaranteed, that there is at least one way to travel from town 1 to town n using these m bidirectional roads.

Each of the next q lines contains two integers t_j and x_j ($1 \le t_j \le m, 1 \le x_j \le 10^9$) — the index of the road Kuro has planned to change and its new price respectively.

Output

Print q integers — the lowest fee Shiro must pay to get from town 1 to town n in each of those q plans.

Examples

input

```
4 5 6
1 2 2
2 4 3
1 4 7
1 3 1
3 4 5
3 4
5 1
3 8
1 4
2 1
3 1
```

output

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

input

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

output

```
1
2
2
3
```

input

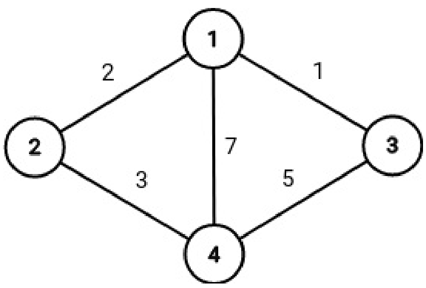
```
2 1 1
1 2 1
1 3
```

output

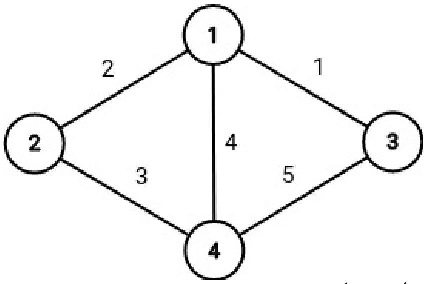
```
3
```

Note

In the first example, the original overview of Capypaland looks like this, where the number next to each road denotes the original prices of the roads,

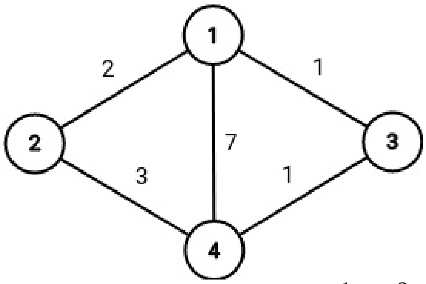


The overview of the first plan,



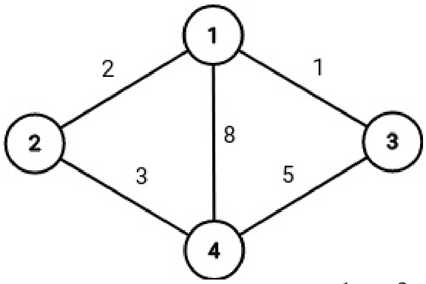
The lowest fee Shiro must pay in this plan is 4, which corresponds to the path $1 \to 4$.

The overview of the second plan,



The lowest fee Shiro must pay in this plan is 2, which corresponds to the path $1 \to 3 \to 4$.

The overview of the third plan,



The lowest fee Shiro must pay in this plan is 5, which corresponds to the path $1 \to 2 \to 4$.