A. Success Rate

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

You are an experienced Codeforces user. Today you found out that during your activity on Codeforces you have made y submissions, out of which x have been successful. Thus, your current success rate on Codeforces is equal to x / y.

Your favorite rational number in the $[0;1]$ range is p / q. Now you wonder: what is the smallest number of submissions you have to make if you want your success rate to be p / q?

Input

The first line contains a single integer t ($1 \le t \le 1000$) — the number of test cases.

Each of the next t lines contains four integers x, y, p and q ($0 \le x \le y \le 10^9$; $0 \le p \le q \le 10^9$; $y > 0$; $q > 0$).

It is guaranteed that p / q is an irreducible fraction.

Hacks. For hacks, an additional constraint of $t \le 5$ must be met.

Output

For each test case, output a single integer equal to the smallest number of submissions you have to make if you want your success rate to be equal to your favorite rational number, or `-1` if this is impossible to achieve.

Example

input

```
4
3 10 1 2
7 14 3 8
20 70 2 7
5 6 1 1
```

output

```
4
10
0
-1
```

Note

In the first example, you have to make 4 successful submissions. Your success rate will be equal to $7 / 14$, or $1 / 2$.

In the second example, you have to make 2 successful and 8 unsuccessful submissions. Your success rate will be equal to $9 / 24$, or $3 / 8$.

In the third example, there is no need to make any new submissions. Your success rate is already equal to $20 / 70$, or $2 / 7$.

In the fourth example, the only unsuccessful submission breaks your hopes of having the success rate equal to 1.

B. Dynamic Problem Scoring

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

Vasya and Petya take part in a Codeforces round. The round lasts for two hours and contains five problems.

For this round the dynamic problem scoring is used. If you were lucky not to participate in any Codeforces round with dynamic problem scoring, here is what it means. The maximum point value of the problem depends on the ratio of the number of participants who solved the problem to the total number of round participants. Everyone who made at least one submission is considered to be participating in the round.

Solvers fraction	Maximum point value
$(1/2, 1]$	500
$(1/4, 1/2]$	1000
$(1/8, 1/4]$	1500
$(1/16, 1/8]$	2000
$(1/32, 1/16]$	2500
$[0, 1/32]$	3000

Pay attention to the range bounds. For example, if 40 people are taking part in the round, and 10 of them solve a particular problem, then the solvers fraction is equal to $1 / 4$, and the problem's maximum point value is equal to 1500.

If the problem's maximum point value is equal to x, then for each whole minute passed from the beginning of the contest to the moment of the participant's correct submission, the participant loses $x / 250$ points. For example, if the problem's maximum point value is 2000, and the participant submits a correct solution to it 40 minutes into the round, this participant will be awarded with $2000 \cdot (1 - 40 / 250) = 1680$ points for this problem.

There are n participants in the round, including Vasya and Petya. For each participant and each problem, the number of minutes which passed between the beginning of the contest and the submission of this participant to this problem is known. It's also possible that this participant made no submissions to this problem.

With two seconds until the end of the round, all participants' submissions have passed pretests, and not a single hack attempt has been made. Vasya believes that no more submissions or hack attempts will be made in the remaining two seconds, and every submission will pass the system testing.

Unfortunately, Vasya is a cheater. He has registered $10^9 + 7$ new accounts for the round. Now Vasya can submit any of his solutions from these new accounts in order to change the maximum point values of the problems. Vasya can also submit any wrong solutions to any problems. Note that Vasya can not submit correct solutions to the problems he hasn't solved.

Vasya seeks to score strictly more points than Petya in the current round. Vasya has already prepared the scripts which allow to obfuscate his solutions and submit them into the system from any of the new accounts in just fractions of seconds. However, Vasya doesn't want to make his cheating too obvious, so he wants to achieve his goal while making submissions from the smallest possible number of new accounts.

Find the smallest number of new accounts Vasya needs in order to beat Petya (provided that Vasya's assumptions are correct), or report that Vasya can't achieve his goal.

Input

The first line contains a single integer n $(2 \le n \le 120)$ — the number of round participants, including Vasya and Petya.

Each of the next n lines contains five integers $a_{i,1}, a_{i,2}..., a_{i,5}$ $(-1 \le a_{i,j} \le 119)$ — the number of minutes passed between the beginning of the round and the submission of problem j by participant i, or `-1` if participant i hasn't solved problem j.

It is guaranteed that each participant has made at least one successful submission.

Vasya is listed as participant number 1, Petya is listed as participant number 2, all the other participants are listed in no particular order.

Output

Output a single integer — the number of new accounts Vasya needs to beat Petya, or `-1` if Vasya can't achieve his goal.

Examples

input
```
2
5 15 40 70 115
50 45 40 30 15
```
output
```
2
```

input
```
3
55 80 10 -1 -1
15 -1 79 60 -1
42 -1 13 -1 -1
```

output

```
3
```

input

```
5
119 119 119 119 119
0 0 0 0 -1
20 65 12 73 77
78 112 22 23 11
1 78 60 111 62
```

output

```
27
```

input

```
4
-1 20 40 77 119
30 10 73 50 107
21 29 -1 64 98
117 65 -1 -1 -1
```

output

```
-1
```

Note

In the first example, Vasya's optimal strategy is to submit the solutions to the last three problems from two new accounts. In this case the first two problems will have the maximum point value of 1000, while the last three problems will have the maximum point value of 500. Vasya's score will be equal to $980 + 940 + 420 + 360 + 270 = 2970$ points, while Petya will score just $800 + 820 + 420 + 440 + 470 = 2950$ points.

In the second example, Vasya has to make a single unsuccessful submission to any problem from two new accounts, and a single successful submission to the first problem from the third new account. In this case, the maximum point values of the problems will be equal to 500, 1500, 1000, 1500, 3000. Vasya will score 2370 points, while Petya will score just 2294 points.

In the third example, Vasya can achieve his goal by submitting the solutions to the first four problems from 27 new accounts. The maximum point values of the problems will be equal to 500, 500, 500, 500, 2000. Thanks to the high cost of the fifth problem, Vasya will manage to beat Petya who solved the first four problems very quickly, but couldn't solve the fifth one.

C. Prairie Partition

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

It can be shown that any positive integer x can be uniquely represented as $x = 1 + 2 + 4 + \ldots + 2^{k-1} + r$, where k and r are integers, $k \ge 0$, $0 < r \le 2^k$. Let's call that representation *prairie partition* of x.

For example, the prairie partitions of 12, 17, 7 and 1 are:

$$12 = 1 + 2 + 4 + 5,$$
$$17 = 1 + 2 + 4 + 8 + 2,$$

$$7 = 1 + 2 + 4,$$

$$1 = 1.$$

Alice took a sequence of positive integers (possibly with repeating elements), replaced every element with the sequence of summands in its prairie partition, arranged the resulting numbers in non-decreasing order and gave them to Borys. Now Borys wonders how many elements Alice's original sequence could contain. Find all possible options!

Input

The first line contains a single integer n ($1 \le n \le 10^5$) — the number of numbers given from Alice to Borys.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^{12}$; $a_1 \le a_2 \le \ldots \le a_n$) — the numbers given from Alice to Borys.

Output

Output, **in increasing order**, all possible values of m such that there exists a sequence of positive integers of length m such that if you replace every element with the summands in its prairie partition and arrange the resulting numbers in non-decreasing order, you will get the sequence given in the input.

If there are no such values of m, output a single integer `-1`.

Examples

input
```
8
1 1 2 2 3 4 5 8
```
output
```
2
```

input
```
6
1 1 1 2 2 2
```
output
```
2 3
```

input
```
5
1 2 4 4 4
```
output
```
-1
```

Note

In the first example, Alice could get the input sequence from $[6, 20]$ as the original sequence.

In the second example, Alice's original sequence could be either $[4, 5]$ or $[3, 3, 3]$.

D. Perishable Roads

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

In the country of Never, there are n cities and a well-developed road system. There is exactly one bidirectional road between every pair of cities, thus, there are as many as $\frac{n(n-1)}{2}$ roads! No two roads intersect, and no road passes through intermediate cities. The art of building tunnels and bridges has been mastered by Neverians.

An independent committee has evaluated each road of Never with a positive integer called the *perishability* of the road. The lower the road's perishability is, the more pleasant it is to drive through this road.

It's the year of transport in Never. It has been decided to build a museum of transport in one of the cities, and to set a single signpost directing to some city (not necessarily the one with the museum) in each of the other cities. The signposts must satisfy the following important condition: if any Neverian living in a city without the museum starts travelling from that city following the directions of the signposts, then this person will eventually arrive in the city with the museum.

Neverians are incredibly positive-minded. If a Neverian travels by a route consisting of several roads, he considers the *perishability of the route* to be equal to the smallest perishability of all the roads in this route.

The government of Never has not yet decided where to build the museum, so they consider all n possible options. The most important is the sum of perishabilities of the routes to the museum city from all the other cities of Never, if the travelers strictly follow the directions of the signposts. The government of Never cares about their citizens, so they want to set the signposts in a way which minimizes this sum. Help them determine the minimum possible sum for all n possible options of the city where the museum can be built.

Input

The first line contains a single integer n ($2 \le n \le 2000$) — the number of cities in Never.

The following $n - 1$ lines contain the description of the road network. The i-th of these lines contains $n - i$ integers. The j-th integer in the i-th line denotes the perishability of the road between cities i and $i + j$.

All road perishabilities are between 1 and 10^9, inclusive.

Output

For each city in order from 1 to n, output the minimum possible sum of perishabilities of the routes to this city from all the other cities of Never if the signposts are set in a way which minimizes this sum.

Examples

input

```
3
1 2
3
```

output

```
2
2
3
```

input

```
6
2 9 9 6 6
7 1 9 10
9 2 5
4 10
8
```

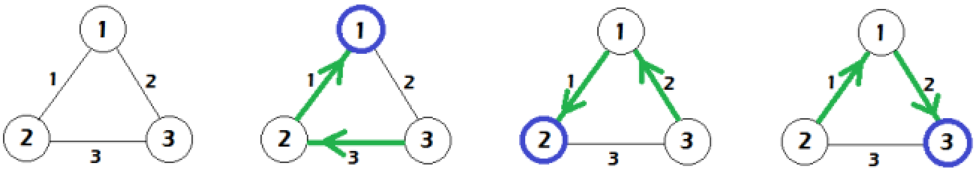
output

```
6
5
7
5
7
11
```

Note

The first example is explained by the picture below. From left to right, there is the initial road network and the optimal directions of the signposts in case the museum is built in city 1, 2 and 3, respectively. The museum city is represented by a blue circle, the directions of the signposts are represented by green arrows.

For instance, if the museum is built in city 3, then the signpost in city 1 must be directed to city 3, while the signpost in city 2 must be directed to city 1. Then the route from city 1 to city 3 will have perishability 2, while the route from city 2 to city 3 will have perishability 1. The sum of perishabilities of these routes is 3.


1
2
3
1
2
3
1
2
3
1
2
3
1
2
3
1
2
3
1
2
3
1
2
3

E. Blog Post Rating

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

It's well-known that blog posts are an important part of Codeforces platform. Every blog post has a global characteristic changing over time — its *community rating*. A newly created blog post's community rating is 0. Codeforces users may visit the blog post page and rate it, changing its community rating by +1 or -1.

Consider the following model of Codeforces users' behavior. The i-th user has his own *estimated blog post rating* denoted by an integer a_i. When a user visits a blog post page, he compares his estimated blog post rating to its community rating. If his estimated rating is higher, he rates the blog post with +1 (thus, the blog post's community rating increases by 1). If his estimated rating is lower, he rates the blog post with -1 (decreasing its community rating by 1). If the estimated rating and the community rating are equal, user doesn't rate the blog post at all (in this case we'll say that user rates the blog post for 0). In any case, after this procedure user closes the blog post page and never opens it again.

Consider a newly created blog post with the initial community rating of 0. For each of n Codeforces users, numbered from 1 to n, his estimated blog post rating a_i is known.

For each k from 1 to n, inclusive, the following question is asked. Let users with indices from 1 to k, **in some order**, visit the blog post page, rate the blog post and close the page. Each user opens the blog post only after the previous user closes it. What could be the maximum possible community rating of the blog post after these k visits?

Input

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

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($-5 \cdot 10^5 \leq a_i \leq 5 \cdot 10^5$) — estimated blog post ratings for users in order from 1 to n.

Output

For each k from 1 to n, output a single integer equal to the maximum possible community rating of the blog post after users with indices from 1 to k, in some order, visit the blog post page, rate the blog post, and close the page.

Examples

input

```
4
2 0 2 2
```

output

```
1
1
2
2
```

input

```
7
2 -3 -2 5 0 -3 1
```

output

```
1
0
-1
0
1
1
2
```

F. Test Data Generation

time limit per test: 5 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Test data generation is not an easy task! Often, generating big random test cases is not enough to ensure thorough testing of solutions for correctness.

For example, consider a problem from an old Codeforces round. Its input format looks roughly as follows:

The first line contains a single integer n ($1 \le n \le max_n$) — the size of the set. The second line contains n distinct integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le max_a$) — the elements of the set ***in increasing order****.*

If you don't pay attention to the problem solution, it looks fairly easy to generate a good test case for this problem. Let $n = max_n$, take random distinct a_i from 1 to max_a, sort them... Soon you understand that it's not that easy.

Here is the actual problem solution. Let g be the greatest common divisor of $a_1, a_2, \ldots, a_n$. Let $x = a_n / g - n$. Then the correct solution outputs "`Alice`" if x is odd, and "`Bob`" if x is even.

Consider two wrong solutions to this problem which differ from the correct one only in the formula for calculating x.

The first wrong solution calculates x as $x = a_n / g$ (without subtracting n).

The second wrong solution calculates x as $x = a_n - n$ (without dividing by g).

A test case is interesting if it makes **both** wrong solutions output an incorrect answer.

Given max_n, max_a and q, find the number of interesting test cases satisfying the constraints, and output it modulo q.

Input

The only line contains three integers max_n, max_a and q ($1 \le max_n \le 30\,000$; $max_n \le max_a \le 10^9$; $10^4 \le q \le 10^5 + 129$).

Output

Output a single integer — the number of test cases which satisfy the constraints and make both wrong solutions output an incorrect answer, modulo q.

Examples

input
```
3 6 100000
```
output
```
4
```

input
```
6 21 100129
```
output
```
154
```

input
```
58 787788 50216
```
output
```
46009
```

Note

In the first example, interesting test cases look as follows:

1 2	1 4	1 6	3 2 4 6