Topic Stream Mashup: Number Theory

A. Maximum GCD

1 second, 256 megabytes

Let's consider all integers in the range from 1 to n (inclusive).

Among all pairs of **distinct** integers in this range, find the maximum possible greatest common divisor of integers in pair. Formally, find the maximum value of $\mathrm{gcd}(a, b)$, where $1 \le a < b \le n$.

The greatest common divisor, $\mathrm{gcd}(a, b)$, of two positive integers a and b is the biggest integer that is a divisor of both a and b.

Input

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

The only line of each test case contains a single integer n ($2 \le n \le 10^6$).

Output

For each test case, output the maximum value of $\mathrm{gcd}(a, b)$ among all $1 \le a < b \le n$.

input

```
2
3
5
```

output

```
1
2
```

In the first test case, $\mathrm{gcd}(1, 2) = \mathrm{gcd}(2, 3) = \mathrm{gcd}(1, 3) = 1$.

In the second test case, 2 is the maximum possible value, corresponding to $\mathrm{gcd}(2, 4)$.

B. EhAb AnD gCd

1 second, 256 megabytes

You are given a positive integer x. Find **any** such 2 positive integers a and b such that $GCD(a,b)+LCM(a,b)=x$.

As a reminder, $GCD(a,b)$ is the greatest integer that divides both a and b. Similarly, $LCM(a,b)$ is the smallest integer such that both a and b divide it.

It's guaranteed that the solution always exists. If there are several such pairs (a, b), you can output any of them.

Input

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

Each testcase consists of one line containing a single integer, x $(2 \le x \le 10^9)$.

Output

For each testcase, output a pair of positive integers a and b ($1 \le a, b \le 10^9$) such that $GCD(a,b)+LCM(a,b)=x$. It's guaranteed that the solution always exists. If there are several such pairs (a, b), you can output any of them.

input

```
2
2
14
```

output

```
1 1
6 4
```

In the first testcase of the sample, $GCD(1,1)+LCM(1,1)=1+1=2$.

In the second testcase of the sample, $GCD(6,4)+LCM(6,4)=2+12=14$.

C. Cat Cycle

1 second, 256 megabytes

Suppose you are living with two cats: A and B. There are n napping spots where both cats usually sleep.

Your cats like to sleep and also like all these spots, so they change napping spot each hour cyclically:

- Cat A changes its napping place in order: $n, n-1, n-2, \dots, 3, 2, 1, n, n-1, \dots$ In other words, at the first hour it's on the spot n and then goes in decreasing order cyclically;
- Cat B changes its napping place in order: $1, 2, 3, \dots, n-1, n, 1, 2, \dots$ In other words, at the first hour it's on the spot 1 and then goes in increasing order cyclically.

The cat B is much younger, so they have a strict hierarchy: A and B don't lie together. In other words, if both cats'd like to go in spot x then the A takes this place and B moves to the next place in its order (if $x < n$ then to $x+1$, but if $x = n$ then to 1). Cat B follows his order, so **it won't return to the skipped spot x after A frees it, but will move to the spot $x+2$ and so on**.

Calculate, where cat B will be at hour k?

Input

The first line contains a single integer t $(1 \le t \le 10^4)$ — the number of test cases.

The first and only line of each test case contains two integers n and k ($2 \le n \le 10^9$; $1 \le k \le 10^9$) — the number of spots and hour k.

Output

For each test case, print one integer — the index of the spot where cat B will sleep at hour k.

input

```
7
2 1
2 2
3 1
3 2
3 3
5 5
69 1337
```

output

```
1
2
1
3
2
2
65
```

In the first and second test cases $n = 2$, so:

- at the 1-st hour, A is on spot 2 and B is on 1;
- at the 2-nd hour, A moves to spot 1 and B — to 2.

If $n = 3$ then:

- at the 1-st hour, A is on spot 3 and B is on 1;
- at the 2-nd hour, A moves to spot 2; B'd like to move from 1 to 2, but this spot is occupied, so it moves to 3;
- at the 3-rd hour, A moves to spot 1; B also would like to move from 3 to 1, but this spot is occupied, so it moves to 2.

In the sixth test case:

- A's spots at each hour are $[5, 4, 3, 2, 1]$;
- B's spots at each hour are $[1, 2, 4, 5, 2]$.

D. New Year and the Sphere Transmission

1 second, 256 megabytes

There are n people sitting in a circle, numbered from 1 to n in the order in which they are seated. That is, for all i from 1 to $n-1$, the people with id i and $i+1$ are adjacent. People with id n and 1 are adjacent as well.

The person with id 1 initially has a ball. He picks a positive integer k at most n, and passes the ball to his k-th neighbour in the direction of increasing ids, that person passes the ball to his k-th neighbour in the same direction, and so on until the person with the id 1 gets the ball back. When he gets it back, people do not pass the ball any more.

For instance, if $n = 6$ and $k = 4$, the ball is passed in order $[1, 5, 3, 1]$.

Consider the set of all people that touched the ball. The **fun value** of the game is the sum of the ids of people that touched it. In the above example, the *fun value* would be $1 + 5 + 3 = 9$.

Find and report the set of possible *fun values* for all choices of positive integer k. It can be shown that under the constraints of the problem, the ball always gets back to the 1-st player after finitely many steps, and there are no more than 10^5 possible *fun values* for given n.

Input

The only line consists of a single integer n ($2 \le n \le 10^9$) — the number of people playing with the ball.

Output

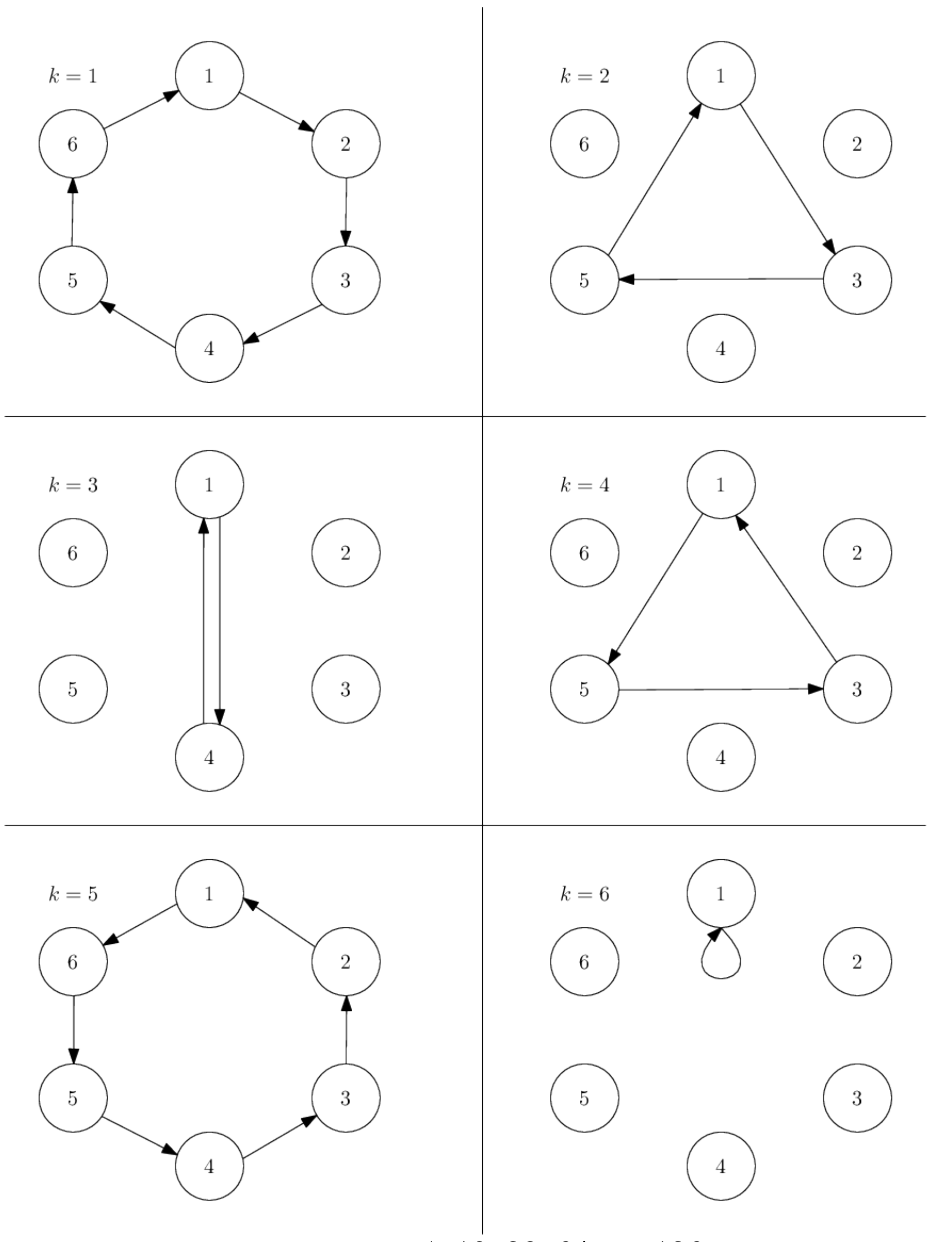
Suppose the set of all *fun values* is $f_1, f_2, \dots, f_m$.

Output a single line containing m space separated integers f_1 through f_m in **increasing** order.

input
```
6
```
output
```
1 5 9 21
```

input
```
16
```
output
```
1 10 28 64 136
```

In the first sample, we've already shown that picking $k = 4$ yields *fun value* 9, as does $k = 2$. Picking $k = 6$ results in *fun value* of 1. For $k = 3$ we get *fun value* 5 and with $k = 1$ or $k = 5$ we get 21.



In the second sample, the values 1, 10, 28, 64 and 136 are achieved for instance for $k = 16$, 8, 4, 10 and 11, respectively.

E. Math

1 second, 256 megabytes

JATC's math teacher always gives the class some interesting math problems so that they don't get bored. Today the problem is as follows. Given an integer n, you can perform the following operations zero or more times:

- `mul` x: multiplies n by x (where x is an arbitrary positive integer).
- `sqrt`: replaces n with $\sqrt{n}$ (to apply this operation, $\sqrt{n}$ must be an integer).

You can perform these operations as many times as you like. What is the minimum value of n, that can be achieved and what is the minimum number of operations, to achieve that minimum value?

Apparently, no one in the class knows the answer to this problem, maybe you can help them?

Input

The only line of the input contains a single integer n ($1 \le n \le 10^6$) — the initial number.

Output

Print two integers: the minimum integer n that can be achieved using the described operations and the minimum number of operations required.

input
```
20
```
output
```
10 2
```

input
```
5184
```
output
```
6 4
```

In the first example, you can apply the operation `mul` 5 to get 100 and then `sqrt` to get 10.

In the second example, you can first apply `sqrt` to get 72, then `mul` 18 to get 1296 and finally two more `sqrt` and you get 6.

Note, that even if the initial value of n is less or equal 10^6, it can still become greater than 10^6 after applying one or more operations.

F. Meaningless Operations

1 second, 256 megabytes

Can the greatest common divisor and bitwise operations have anything in common? It is time to answer this question.

Suppose you are given a positive integer a. You want to choose some integer b from 1 to $a - 1$ inclusive in such a way that the greatest common divisor (GCD) of integers $a \oplus b$ and $a \,\&\, b$ is as large as possible. In other words, you'd like to compute the following function:

$$f(a) = \max_{0<b<a} gcd(a \oplus b, a \,\&\, b).$$

Here $\oplus$ denotes the bitwise XOR operation, and $\&$ denotes the bitwise AND operation.

The greatest common divisor of two integers x and y is the largest integer g such that both x and y are divided by g without remainder.

You are given q integers $a_1, a_2, \dots, a_q$. For each of these integers compute the largest possible value of the greatest common divisor (when b is chosen optimally).

Input

The first line contains an integer q ($1 \le q \le 10^3$) — the number of integers you need to compute the answer for.

After that q integers are given, one per line: $a_1, a_2, \dots, a_q$ ($2 \le a_i \le 2^{25} - 1$) — the integers you need to compute the answer for.

Output

For each integer, print the answer in the same order as the integers are given in input.

input

```
3
2
3
5
```

output

```
3
1
7
```

For the first integer the optimal choice is $b = 1$, then $a \oplus b = 3$, $a \mathbin{\&} b = 0$, and the greatest common divisor of 3 and 0 is 3.

For the second integer one optimal choice is $b = 2$, then $a \oplus b = 1$, $a \mathbin{\&} b = 2$, and the greatest common divisor of 1 and 2 is 1.

For the third integer the optimal choice is $b = 2$, then $a \oplus b = 7$, $a \mathbin{\&} b = 0$, and the greatest common divisor of 7 and 0 is 7.

G. Floor and Mod

2 seconds, 256 megabytes

A pair of positive integers (a, b) is called **special** if $\lfloor \frac{a}{b} \rfloor = a \bmod b$. Here, $\lfloor \frac{a}{b} \rfloor$ is the result of the integer division between a and b, while $a \bmod b$ is its remainder.

You are given two integers x and y. Find the number of special pairs (a, b) such that $1 \leq a \leq x$ and $1 \leq b \leq y$.

Input

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

The only line of the description of each test case contains two integers x, y $(1 \leq x, y \leq 10^9)$.

Output

For each test case print the answer on a single line.

input

```
9
3 4
2 100
4 3
50 3
12 4
69 420
12345 6789
123456 789
12345678 9
```

output

```
1
0
2
3
5
141
53384
160909
36
```

In the first test case, the only special pair is $(3, 2)$.

In the second test case, there are no special pairs.

In the third test case, there are two special pairs: $(3, 2)$ and $(4, 3)$.

H. Enlarge GCD

1 second, 256 megabytes

Mr. F has n positive integers, $a_1, a_2, \ldots, a_n$.

He thinks the greatest common divisor of these integers is too small. So he wants to enlarge it by removing some of the integers.

But this problem is too simple for him, so he does not want to do it by himself. If you help him, he will give you some scores in reward.

Your task is to calculate the minimum number of integers you need to remove so that the greatest common divisor of the remaining integers is bigger than that of all integers.

Input

The first line contains an integer n $(2 \leq n \leq 3 \cdot 10^5)$ — the number of integers Mr. F has.

The second line contains n integers, $a_1, a_2, \ldots, a_n$ ($1 \leq a_i \leq 1.5 \cdot 10^7$).

Output

Print an integer — the minimum number of integers you need to remove so that the greatest common divisor of the remaining integers is bigger than that of all integers.

You should not remove all of the integers.

If there is no solution, print «`-1`» (without quotes).

input

```
3
1 2 4
```

output

```
1
```

input

```
4
6 9 15 30
```

output

```
2
```

input

```
3
1 1 1
```

output

```
-1
```

In the first example, the greatest common divisor is 1 in the beginning. You can remove 1 so that the greatest common divisor is enlarged to 2. The answer is 1.

In the second example, the greatest common divisor is 3 in the beginning. You can remove 6 and 9 so that the greatest common divisor is enlarged to 15. There is no solution which removes only one integer. So the answer is 2.

In the third example, there is no solution to enlarge the greatest common divisor. So the answer is -1.

I. Remainders Game

1 second, 256 megabytes

Today Pari and Arya are playing a game called Remainders.

Pari chooses two positive integer x and k, and tells Arya k but not x. Arya have to find the value $x \bmod k$. There are n ancient numbers $c_1, c_2, ..., c_n$ and Pari has to tell Arya $x \bmod c_i$ if Arya wants. Given k and the ancient values, tell us if Arya has a winning strategy independent of value of x or not. Formally, is it true that Arya can understand the value $x \bmod k$ for any positive integer x?

Note, that $x \bmod y$ means the remainder of x after dividing it by y.

Input

The first line of the input contains two integers n and k $(1 \leq n,\ k \leq 1\,000\,000)$ — the number of ancient integers and value k that is chosen by Pari.

The second line contains n integers $c_1, c_2, ..., c_n$ $(1 \leq c_i \leq 1\,000\,000)$.

Output

Print "`Yes`" (without quotes) if Arya has a winning strategy independent of value of x, or "`No`" (without quotes) otherwise.

input

```
4 5
2 3 5 12
```

output

```
Yes
```

input

```
2 7
2 3
```

output

```
No
```

In the first sample, Arya can understand $x \mod 5$ because 5 is one of the ancient numbers.

In the second sample, Arya can't be sure what $x \mod 7$ is. For example 1 and 7 have the same remainders after dividing by 2 and 3, but they differ in remainders after dividing by 7.

J. Coprime Subsequences Redux

2 seconds, 256 megabytes

You have an array a with n integers, all between 1 and m. For all x between 1 and m, find the number of subsequences where the GCD of all elements in the subsequence is equal to x. Output your answer(s) modulo $10^9 + 7$.

Input

The first line contains two integers n $(1 \le n \le 10^5)$ and m $(1 \le m \le 10^5)$.

The second line contains n integers, the array a $(1 \le a_i \le m)$.

Output

Output m integers, where the i-th integer is the answer for $x = i$, mod $10^9 + 7$.

input
```
3 3
1 2 3
```
output
```
5 1 1
```

input
```
4 10
1 1 1 1
```
output
```
15 0 0 0 0 0 0 0 0 0
```

input
```
7 30
1 2 3 6 2 30 15
```
output
```
100 12 10 0 0 2 0 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1
```

K. Two Divisors

2 seconds, 256 megabytes

You are given n integers $a_1, a_2, \dots, a_n$.

For each a_i find its **two divisors** $d_1 > 1$ and $d_2 > 1$ such that $\gcd(d_1 + d_2, a_i) = 1$ (where $\gcd(a, b)$ is the greatest common divisor of a and b) or say that there is no such pair.

Input

The first line contains single integer n $(1 \le n \le 5 \cdot 10^5)$ — the size of the array a.

The second line contains n integers $a_1, a_2, \dots, a_n$ $(2 \le a_i \le 10^7)$ — the array a.

Output

To speed up the output, print two lines with n integers in each line.

The i-th integers in the first and second lines should be corresponding divisors $d_1 > 1$ and $d_2 > 1$ such that $\gcd(d_1 + d_2, a_i) = 1$ or -1 and -1 if there is no such pair. If there are multiple answers, print any of them.

input
```
10
2 3 4 5 6 7 8 9 10 24
```
output
```
-1 -1 -1 -1 3 -1 -1 -1 2 2
-1 -1 -1 -1 2 -1 -1 -1 5 3
```

Let's look at $a_7 = 8$. It has 3 divisors greater than 1: $2, 4, 8$. As you can see, the sum of any pair of divisors is divisible by 2 as well as a_7.

There are other valid pairs of d_1 and d_2 for $a_{10} = 24$, like $(3, 4)$ or $(8, 3)$. You can print any of them.

L. Congruence Equation

3 seconds, 256 megabytes

Given an integer x. Your task is to find out how many positive integers n ($1 \le n \le x$) satisfy

$$n \cdot a^n \equiv b \pmod p,$$

where a, b, p are all known constants.

Input

The only line contains four integers a, b, p, x ($2 \le p \le 10^6 + 3$, $1 \le a, b < p$, $1 \le x \le 10^{12}$). It is guaranteed that p is a prime.

Output

Print a single integer: the number of possible answers n.

input
```
2 3 5 8
```
output
```
2
```

input
```
4 6 7 13
```
output
```
1
```

input
```
233 233 10007 1
```
output
```
1
```

In the first sample, we can see that $n = 2$ and $n = 8$ are possible answers.

M. Mocha and Stars

2 seconds, 256 megabytes

Mocha wants to be an astrologer. There are n stars which can be seen in Zhijiang, and the brightness of the i-th star is a_i.

Mocha considers that these n stars form a constellation, and she uses $(a_1, a_2, \dots, a_n)$ to show its state. A state is called *mathematical* if all of the following three conditions are satisfied:

- For all i $(1 \le i \le n)$, a_i is an integer in the range $[l_i, r_i]$.
- $\sum\limits_{i=1}^{n} a_i \le m$.
- $\gcd(a_1, a_2, \dots, a_n) = 1$.

Here, $\gcd(a_1, a_2, \dots, a_n)$ denotes the greatest common divisor (GCD) of integers $a_1, a_2, \dots, a_n$.

Mocha is wondering how many different mathematical states of this constellation exist. Because the answer may be large, you must find it modulo $998\,244\,353$.

Two states $(a_1, a_2, \dots, a_n)$ and $(b_1, b_2, \dots, b_n)$ are considered different if there exists i $(1 \le i \le n)$ such that $a_i \ne b_i$.

Input

The first line contains two integers n and m ($2 \le n \le 50$, $1 \le m \le 10^5$) — the number of stars and the upper bound of the sum of the brightness of stars.

Each of the next n lines contains two integers l_i and r_i ($1 \le l_i \le r_i \le m$) — the range of the brightness of the i-th star.

Output

Print a single integer — the number of different mathematical states of this constellation, modulo $998\,244\,353$.

input

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

output

```
4
```

input

```
5 10
1 10
1 10
1 10
1 10
1 10
```

output

```
251
```

input

```
5 100
1 94
1 96
1 91
4 96
6 97
```

output

```
47464146
```

In the first example, there are 4 different mathematical states of this constellation:

- $a_1 = 1, a_2 = 1$.
- $a_1 = 1, a_2 = 2$.
- $a_1 = 2, a_2 = 1$.
- $a_1 = 3, a_2 = 1$.

N. Prime Gift

3.5 seconds, 256 megabytes

Opposite to Grisha's nice behavior, Oleg, though he has an entire year at his disposal, didn't manage to learn how to solve number theory problems in the past year. That's why instead of Ded Moroz he was visited by his teammate Andrew, who solemnly presented him with a set of n **distinct prime** numbers alongside with a simple task: Oleg is to find the k-th smallest integer, such that **all** its prime divisors are in this set.

Input

The first line contains a single integer n $(1 \le n \le 16)$.

The next line lists n distinct prime numbers $p_1, p_2, ..., p_n$ $(2 \le p_i \le 100)$ in ascending order.

The last line gives a single integer k $(1 \le k)$. It is guaranteed that the k-th smallest integer such that all its prime divisors are in this set does not exceed 10^{18}.

Output

Print a single line featuring the k-th smallest integer. It's guaranteed that the answer doesn't exceed 10^{18}.

input

```
3
2 3 5
7
```

output

```
8
```

input

```
5
3 7 11 13 31
17
```

output

```
93
```

The list of numbers with all prime divisors inside $\{2, 3, 5\}$ begins as follows:

$$(1, 2, 3, 4, 5, 6, 8, ...)$$

The seventh number in this list (1-indexed) is eight.

O. Xum

2 seconds, 256 megabytes

You have a blackboard and initially only an **odd** number x is written on it. Your goal is to write the number 1 on the blackboard.

You may write new numbers on the blackboard with the following two operations.

- You may take two numbers (not necessarily distinct) already on the blackboard and write their sum on the blackboard. The two numbers you have chosen remain on the blackboard.
- You may take two numbers (not necessarily distinct) already on the blackboard and write their bitwise XOR on the blackboard. The two numbers you have chosen remain on the blackboard.

Perform a sequence of operations such that at the end the number 1 is on the blackboard.

Input

The single line of the input contains the odd integer x ($3 \le x \le 999,999$).

Output

Print on the first line the number q of operations you perform. Then q lines should follow, each describing one operation.

- The "sum" operation is described by the line "a + b", where a, b must be integers already present on the blackboard.
- The "xor" operation is described by the line "a ^ b", where a, b must be integers already present on the blackboard.

The operation symbol (+ or ^) must be separated from a, b by a whitespace.

You can perform at most $100,000$ operations (that is, $q \le 100,000$) and all numbers written on the blackboard must be in the range $[0, 5 \cdot 10^{18}]$. It can be proven that under such restrictions the required sequence of operations exists. You can output any suitable sequence of operations.

input

```
3
```

output

```
5
3 + 3
3 ^ 6
3 + 5
3 + 6
8 ^ 9
```

input

```
123
```

output

```
10
123 + 123
123 ^ 246
141 + 123
246 + 123
264 ^ 369
121 + 246
367 ^ 369
30 + 30
60 + 60
120 ^ 121
```