[DivC] Contest #1

A. Cutting Carrot

2 seconds, 256 megabytes

Igor the analyst has adopted n little bunnies. As we all know, bunnies love carrots. Thus, Igor has bought a carrot to be shared between his bunnies. Igor wants to treat all the bunnies equally, and thus he wants to cut the carrot into n pieces of equal area.

Formally, the carrot can be viewed as an isosceles triangle with base length equal to 1 and height equal to h. Igor wants to make $n - 1$ cuts **parallel to the base** to cut the carrot into n pieces. He wants to make sure that all n pieces have the same area. Can you help Igor determine where to cut the carrot so that each piece have equal area?

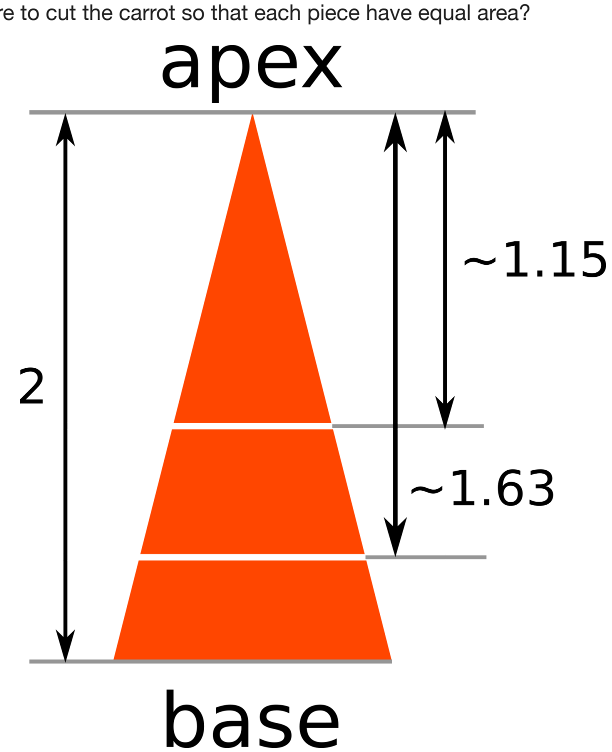


Illustration to the first example.

Input

The first and only line of input contains two space-separated integers, n and h ($2 \le n \le 1000$, $1 \le h \le 10^5$).

Output

The output should contain $n - 1$ real numbers $x_1, x_2, ..., x_{n-1}$. The number x_i denotes that the i-th cut must be made x_i units away from the apex of the carrot. In addition, $0 < x_1 < x_2 < ... < x_{n-1} < h$ must hold.

Your output will be considered correct if absolute or relative error of every number in your output doesn't exceed 10^{-6}.

Formally, let your answer be a, and the jury's answer be b. Your answer is considered correct if $\frac{|a-b|}{\max(1,b)} \le 10^{-6}$.

input
```
3 2
```
output
```
1.154700538379 1.632993161855
```

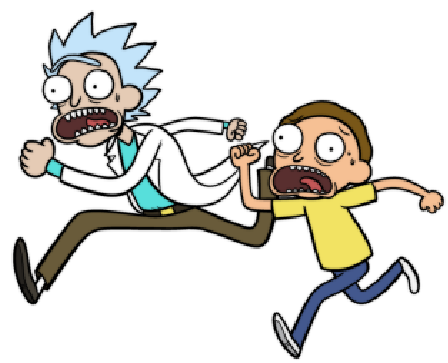
input
```
2 100000
```
output
```
70710.678118654752
```

Definition of isosceles triangle: https://en.wikipedia.org/wiki/Isosceles_triangle.

B. The Monster

1 second, 256 megabytes

A monster is chasing after Rick and Morty on another planet. They're so frightened that sometimes they scream. More accurately, Rick screams at times $b, b+a, b+2a, b+3a, ...$ and Morty screams at times $d, d+c, d+2c, d+3c, ...$.



The Monster will catch them if at any point they scream at the same time, so it wants to know when it will catch them (the first time they scream at the same time) or that they will never scream at the same time.

Input

The first line of input contains two integers a and b ($1 \le a, b \le 100$).

The second line contains two integers c and d ($1 \le c, d \le 100$).

Output

Print the first time Rick and Morty will scream at the same time, or -1 if they will never scream at the same time.

input
```
20 2
9 19
```
output
```
82
```

input
```
2 1
16 12
```
output
```
-1
```

In the first sample testcase, Rick's 5th scream and Morty's 8th time are at time 82.

In the second sample testcase, all Rick's screams will be at odd times and Morty's will be at even times, so they will never scream at the same time.

C. Prime Graph

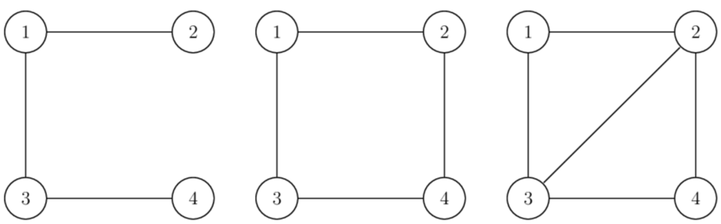
1 second, 256 megabytes

Every person likes prime numbers. Alice is a person, thus she also shares the love for them. Bob wanted to give her an affectionate gift but couldn't think of anything inventive. Hence, he will be giving her a graph. How original, Bob! Alice will surely be *thrilled*!

When building the graph, he needs four conditions to be satisfied:

- It must be a simple undirected graph, i.e. without multiple (parallel) edges and self-loops.
- The number of vertices must be exactly n — a number he selected. This number is not necessarily prime.
- The total number of edges must be prime.
- The degree (i.e. the number of edges connected to the vertex) of each vertex must be prime.

Below is an example for $n = 4$. The first graph (left one) is invalid as the degree of vertex 2 (and 4) equals to 1, which is not prime. The second graph (middle one) is invalid as the total number of edges is 4, which is not a prime number. The third graph (right one) is a valid answer for $n = 4$.



Note that the graph can be disconnected.

Please help Bob to find any such graph!

Input

The input consists of a single integer n $(3 \le n \le 1\,000)$ — the number of vertices.

Output

If there is no graph satisfying the conditions, print a single line containing the integer -1.

Otherwise, first print a line containing a prime number m ($2 \le m \le \frac{n(n-1)}{2}$) — the number of edges in the graph. Then, print m lines, the i-th of which containing two integers u_i, v_i $(1 \le u_i, v_i \le n)$ — meaning that there is an edge between vertices u_i and v_i. The degree of each vertex must be prime. There must be no multiple (parallel) edges or self-loops.

If there are multiple solutions, you may print any of them.

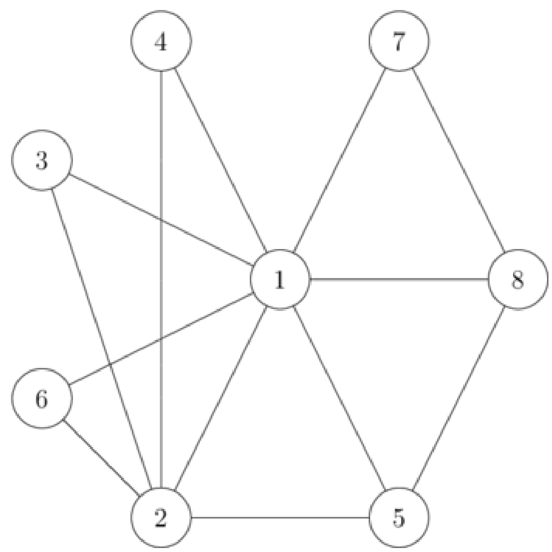
Note that the graph can be disconnected.

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

input
```
8
```
output
```
13
1 2
1 3
2 3
1 4
2 4
1 5
2 5
1 6
2 6
1 7
1 8
5 8
7 8
```

The first example was described in the statement.

In the second example, the degrees of vertices are $[7, 5, 2, 2, 3, 2, 2, 3]$. Each of these numbers is prime. Additionally, the number of edges, 13, is also a prime number, hence both conditions are satisfied.



D. 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, $GCD(6,4) + LCM(6,4) = 2 + 12 = 14$.

E. K-th Missing Digit

0.5 seconds, 32 megabytes

You're given two positive integers A and B, and a string P, representing their product but with a missing digit indicated with an $*$. Find the missing digit.

It's guaranteed that the missing digit isn't 0.

Input

The first line contains three integers a, b and p $(1 \le a, b < 10^6$, $1 \le p < 2 * 10^6)$ – the number of digits in A, B and P respectively.

The second line contains positive number A.

The third line contains positive number B.

The fourth line contains string P – the product of A and B with a missing digit.

Output

Print one integer representing the missing digit in P.

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

input
```
2 2 3
10
10
*00
```
output
```
1
```

F. Solution for Cube

2 seconds, 256 megabytes

During the breaks between competitions, top-model Izabella tries to develop herself and not to be bored. For example, now she tries to solve Rubik's cube 2x2x2.

It's too hard to learn to solve Rubik's cube instantly, so she learns to understand if it's possible to solve the cube in some state using 90-degrees rotation of one face of the cube in any direction.

To check her answers she wants to use a program which will for some state of cube tell if it's possible to solve it using one rotation, described above.

Cube is called solved if for each face of cube all squares on it has the same color.

https://en.wikipedia.org/wiki/Rubik's_Cube

Input

In first line given a sequence of 24 integers a_i $(1 \le a_i \le 6)$, where a_i denotes color of i-th square. There are exactly 4 occurrences of all colors in this sequence.

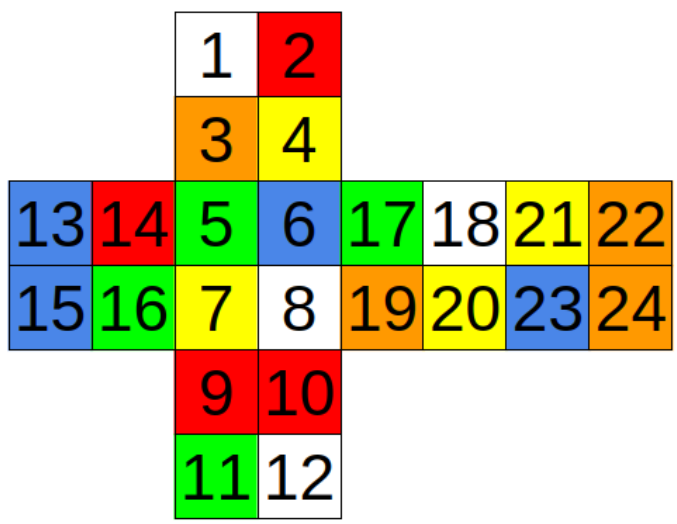
Output

Print «YES» (without quotes) if it's possible to solve cube using one rotation and «NO» (without quotes) otherwise.

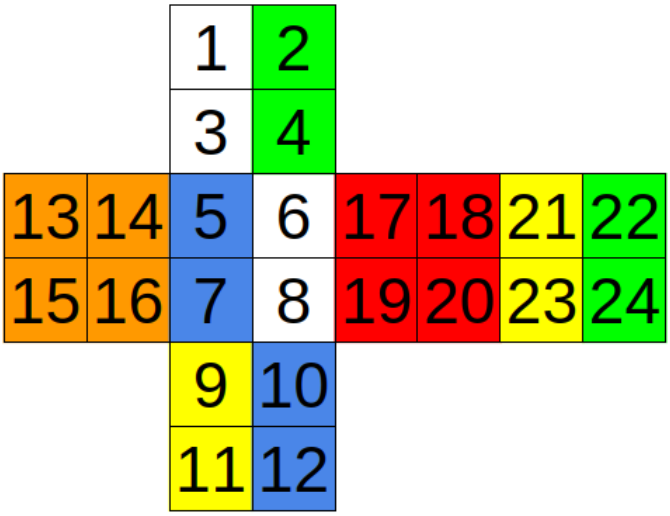
input
```
2 5 4 6 1 3 6 2 5 5 1 2 3 5 3 1 1 2 4 6 6 4 3 4
```
output
```
NO
```

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

In first test case cube looks like this:



In second test case cube looks like this:



It's possible to solve cube by rotating face with squares with numbers 13, 14, 15, 16.

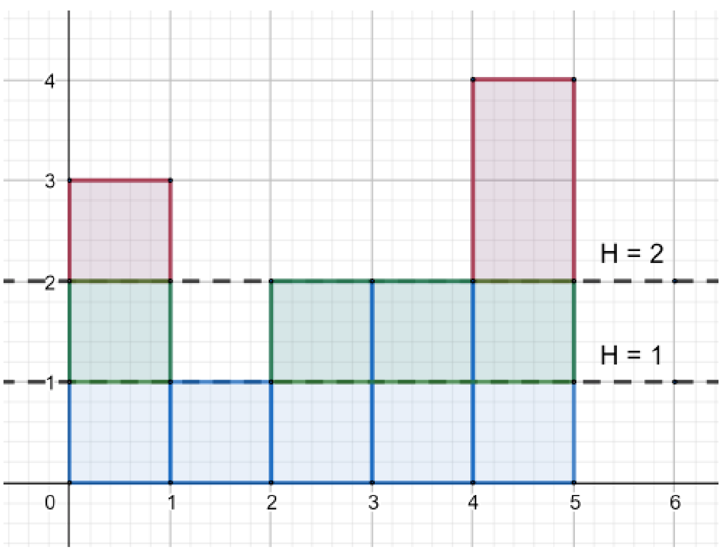
G. Make It Equal

2 seconds, 256 megabytes

There is a toy building consisting of n towers. Each tower consists of several cubes standing on each other. The i-th tower consists of h_i cubes, so it has height h_i.

Let's define operation *slice* on some height H as following: for each tower i, if its height is greater than H, then remove some top cubes to make tower's height equal to H. Cost of one "slice" equals to the total number of removed cubes from all towers.

Let's name slice as *good* one if its cost is lower or equal to k ($k \ge n$).



Calculate the minimum number of good slices you have to do to make all towers have the same height. Of course, it is always possible to make it so.

Input

The first line contains two integers n and k ($1 \le n \le 2 \cdot 10^5$, $n \le k \le 10^9$) — the number of towers and the restriction on slices, respectively.

The second line contains n space separated integers $h_1, h_2, \dots, h_n$ ($1 \le h_i \le 2 \cdot 10^5$) — the initial heights of towers.

Output

Print one integer — the minimum number of good slices you have to do to make all towers have the same heigth.

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

input
```
4 5
2 3 4 5
```
output
```
2
```

In the first example it's optimal to make 2 slices. The first slice is on height 2 (its cost is 3), and the second one is on height 1 (its cost is 4).

H. Medium Design

3 seconds, 256 megabytes

The array $a_1, a_2, \dots, a_m$ is initially filled with zeroes. You are given n pairwise distinct segments $1 \le l_i \le r_i \le m$. You have to select an arbitrary subset of these segments (in particular, you may select an empty set). Next, you do the following:

- For each $i = 1, 2, \dots, n$, if the segment (l_i, r_i) has been selected to the subset, then for each index $l_i \le j \le r_i$ you increase a_j by 1 (i. e. a_j is replaced by $a_j + 1$). If the segment (l_i, r_i) has not been selected, the array does not change.
- Next (after processing all values of $i = 1, 2, \dots, n$), you compute $\max(a)$ as the maximum value among all elements of a. Analogously, compute $\min(a)$ as the minimum value.
- Finally, the cost of the selected subset of segments is declared as $\max(a) - \min(a)$.

Please, find the maximum cost among all subsets of segments.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 10^4$). The description of the test cases follows.

The first line of each test case contains two integers n and m ($1 \le n \le 10^5$, $1 \le m \le 10^9$) — the number of segments and the length of the array.

The following n lines of each test case describe the segments. The i-th of these lines contains two integers l_i and r_i ($1 \le l_i \le r_i \le m$). It is guaranteed that the segments are pairwise distinct.

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 the maximum cost among all subsets of the given set of segments.

input

```
6
1 3
2 2
3 8
2 4
3 5
4 6
6 3
1 1
1 2
1 3
2 2
2 3
3 3
7 6
2 2
1 6
1 2
5 6
1 5
4 4
3 6
6 27
6 26
5 17
2 3
20 21
1 22
12 24
4 1000000000
2 999999999
3 1000000000
123456789 987654321
9274 123456789
```

output

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

In the first test case, there is only one segment available. If we do not select it, then the array will be $a = [0, 0, 0]$, and the cost of such (empty) subset of segments will be 0. If, however, we select the only segment, the array will be $a = [0, 1, 0]$, and the cost will be $1 - 0 = 1$.

In the second test case, we can select all the segments: the array will be $a = [0, 1, 2, 3, 2, 1, 0, 0]$ in this case. The cost will be $3 - 0 = 3$.

I. k-th divisor

2 seconds, 256 megabytes

You are given two integers n and k. Find k-th smallest divisor of n, or report that it doesn't exist.

Divisor of n is any such natural number, that n can be divided by it without remainder.

Input

The first line contains two integers n and k ($1 \le n \le 10^{15}$, $1 \le k \le 10^9$).

Output

If n has less than k divisors, output `-1`.

Otherwise, output the k-th smallest divisor of n.

input

```
4 2
```

output

```
2
```

input

```
5 3
```

output

```
-1
```

input

```
12 5
```

output

```
6
```

In the first example, number 4 has three divisors: 1, 2 and 4. The second one is 2.

In the second example, number 5 has only two divisors: 1 and 5. The third divisor doesn't exist, so the answer is `-1`.

J. Plasticine zebra

1 second, 256 megabytes

Is there anything better than going to the zoo after a tiresome week at work? No wonder Grisha feels the same while spending the entire weekend accompanied by pretty striped zebras.

Inspired by this adventure and an accidentally found plasticine pack (represented as a sequence of black and white stripes), Grisha now wants to select several consequent (contiguous) pieces of alternating colors to create a zebra. Let's call the number of selected pieces the length of the zebra.

Before assembling the zebra Grisha can make the following operation 0 or more times. He splits the sequence in some place into two parts, then reverses each of them and sticks them together again. For example, if Grisha has pieces in the order "`bwbbw`" (here '`b`' denotes a black strip, and '`w`' denotes a white strip), then he can split the sequence as `bw|bbw` (here the vertical bar represents the cut), reverse both parts and obtain "`wbwbb`".

Determine the maximum possible length of the zebra that Grisha can produce.

Input

The only line contains a string s ($1 \le |s| \le 10^5$, where $|s|$ denotes the length of the string s) comprised of lowercase English letters '`b`' and '`w`' only, where '`w`' denotes a white piece and '`b`' denotes a black piece.

Output

Print a single integer — the maximum possible zebra length.

input

```
bwwwbwwbw
```

output

```
5
```

input

```
bwwbwwb
```

output

```
3
```

In the first example one of the possible sequence of operations is `bwwwbww|bw` → `w|wbwwwbwb` → **`wbwbw`**`wwbw`, that gives the answer equal to 5.

In the second example no operation can increase the answer.

K. Coin Games

1 second, 256 megabytes

There are n coins on the table forming a circle, and each coin is either facing up or facing down. Alice and Bob take turns to play the following game, and Alice goes first.

In each operation, the player chooses a facing-up coin, removes the coin, and flips the two coins that are adjacent to it. If (before the operation) there are only two coins left, then one will be removed and the other won't be flipped (as it would be flipped twice). If (before the operation) there is only one coin left, no coins will be flipped. If (before the operation) there are no facing-up coins, the player loses.

Decide who will win the game if they both play optimally. It can be proved that the game will end in a finite number of operations, and one of them will win.

Input

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

The first line of each test case contains only one positive integer n ($1 \leq n \leq 100$), representing the number of the coins.

A string s of length n follows on the second line of each test case, containing only "`U`" and "`D`", representing that each coin is facing up or facing down.

Output

For each test case, print "`YES`" if Alice will win the game, and "`NO`" otherwise.

You can output the answer in any case (upper or lower). For example, the strings "`yEs`", "`yes`", "`Yes`", and "`YES`" will be recognized as positive responses.

input
```
3
5
UUDUD
5
UDDUD
2
UU
```
output
```
YES
NO
NO
```

In the first test case, the game may go as follows.

- Alice chooses the first coin and s becomes "`DDUU`".
- Bob chooses the last coin and s becomes "`UDD`".
- Alice chooses the first coin and s becomes "`UU`".
- Bob chooses the first coin and s becomes "`U`".
- Alice chooses the only coin and s becomes empty.
- Bob can't choose any coin now, and he loses the game.

It can be proved that Bob will always lose if they both play optimally.

L. Mike and palindrome

2 seconds, 256 megabytes

Mike has a string s consisting of only lowercase English letters. He wants to **change exactly one** character from the string so that the resulting one is a palindrome.

A palindrome is a string that reads the same backward as forward, for example strings "`z`", "`aaa`", "`aba`", "`abccba`" are palindromes, but strings "`codeforces`", "`reality`", "`ab`" are not.

Input

The first and single line contains string s ($1 \leq |s| \leq 15$).

Output

Print "`YES`" (without quotes) if Mike can change **exactly** one character so that the resulting string is palindrome or "`NO`" (without quotes) otherwise.

input
```
abccaa
```
output
```
YES
```

input
```
abbcca
```
output
```
NO
```

input
```
abcda
```
output
```
YES
```