Codeforces Round #120 (Div. 2)

A. Vasya and the Bus

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

One day Vasya heard a story: "In the city of High Bertown a bus number 62 left from the bus station. It had n grown-ups and m kids..."

The latter events happen to be of no importance to us. Vasya is an accountant and he loves counting money. So he wondered what maximum and minimum sum of money these passengers could have paid for the ride.

The bus fare equals one berland ruble in High Bertown. However, not everything is that easy — **no more than one** child can ride for free with each grown-up passenger. That means that a grown-up passenger who rides with his k $(k > 0)$ children, pays overall k rubles: a ticket for himself and $(k - 1)$ tickets for his children. Also, a grown-up can ride without children, in this case he only pays one ruble.

We know that in High Bertown children can't ride in a bus unaccompanied by grown-ups.

Help Vasya count the minimum and the maximum sum in Berland rubles, that all passengers of this bus could have paid in total.

Input

The input file consists of a single line containing two space-separated numbers n and m $(0 \le n, m \le 10^5)$ — the number of the grown-ups and the number of the children in the bus, correspondingly.

Output

If n grown-ups and m children could have ridden in the bus, then print on a single line two space-separated integers — the minimum and the maximum possible total bus fare, correspondingly.

Otherwise, print "`Impossible`" (without the quotes).

Sample test(s)

input
1 2
output
2 2

input
0 5
output
Impossible

input
2 2
output
2 3

Note

In the first sample a grown-up rides with two children and pays two rubles.

In the second sample there are only children in the bus, so the situation is impossible.

In the third sample there are two cases:

- Each of the two grown-ups rides with one children and pays one ruble for the tickets. In this case the passengers pay two rubles in total.
- One of the grown-ups ride with two children's and pays two rubles, the another one rides alone and pays one ruble for himself. So, they pay three rubles in total.

B. Surrounded

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

So, the Berland is at war with its eternal enemy Flatland again, and Vasya, an accountant, was assigned to fulfil his duty to the nation.

Right now the situation in Berland is dismal — their both cities are surrounded! The armies of flatlanders stand on the borders of circles, the circles' centers are in the surrounded cities. At any moment all points of the flatland ring can begin to move quickly in the direction of the city — that's the strategy the flatlanders usually follow when they besiege cities.

The berlanders are sure that they can repel the enemy's attack if they learn the exact time the attack starts. For that they need to construct a radar that would register any movement at the distance of at most r from it. Thus, we can install a radar at such point, that at least one point of the enemy ring will be in its detecting range (that is, at a distance of at most r). Then the radar can immediately inform about the enemy's attack.

Due to the newest technologies, we can place a radar at any point without any problems. But the problem is that the berlanders have the time to make only one radar. Besides, the larger the detection radius (r) is, the more the radar costs.

That's why Vasya's task (that is, your task) is to find the minimum possible detection radius for the radar. In other words, your task is to find the minimum radius r $(r \geq 0)$ such, that a radar with radius r can be installed at some point and it can register **the start of the movements** of both flatland rings from that point.

In this problem you can consider the cities as material points, the attacking enemy rings - as circles with centers in the cities, the radar's detection range — as a disk (including the border) with the center at the point where the radar is placed.

Input

The input files consist of two lines. Each line represents the city and the flatland ring that surrounds it as three space-separated integers x_i, y_i, r_i $(|x_i|, |y_i| \leq 10^4; 1 \leq r_i \leq 10^4)$ — the city's coordinates and the distance from the city to the flatlanders, correspondingly.

It is guaranteed that the cities are located at different points.

Output

Print a single real number — the minimum detection radius of the described radar. The answer is considered correct if the absolute or relative error does not exceed 10^{-6}.

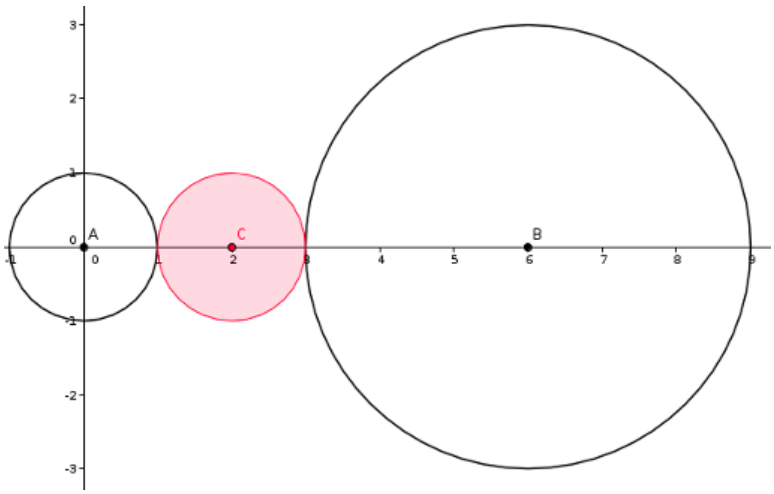
Sample test(s)

input
0 0 1 6 0 3
output
1.000000000000000

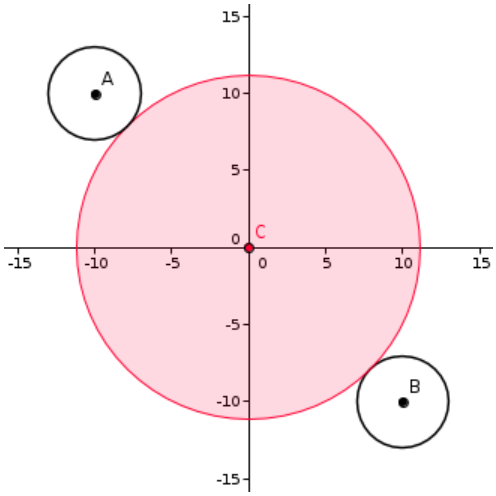
input
-10 10 3 10 -10 3
output
11.142135623730951

Note

The figure below shows the answer to the first sample. In this sample the best decision is to put the radar at point with coordinates $(2, 0)$.



The figure below shows the answer for the second sample. In this sample the best decision is to put the radar at point with coordinates $(0, 0)$.


A
B
C
15
10
5
0
-5
-10
-15
-15
-10
-5
0
5
10
15

C. STL

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

Vasya used to be an accountant before the war began and he is one of the few who knows how to operate a computer, so he was assigned as the programmer.

We all know that programs often store sets of integers. For example, if we have a problem about a weighted directed graph, its edge can be represented by three integers: the number of the starting vertex, the number of the final vertex and the edge's weight. So, as Vasya was trying to represent characteristics of a recently invented robot in his program, he faced the following problem.

Vasya is not a programmer, so he asked his friend Gena, what the convenient way to store n integers is. Gena used to code in language X-- and so he can use only the types that occur in this language. Let's define, what a "type" is in language X--:

- First, a type is a string "`int`".
- Second, a type is a string that starts with "`pair`", then followed by angle brackets listing **exactly two** comma-separated other types of language X--. This record contains no spaces.
- No other strings can be regarded as types.

More formally: `type` := `int` | `pair<type,type>`. For example, Gena uses the following type for graph edges: `pair<int,pair<int,int>>`.

Gena was pleased to help Vasya, he dictated to Vasya a type of language X--, that stores n integers. Unfortunately, Gena was in a hurry, so he omitted the punctuation. Now Gena has already left and Vasya can't find the correct punctuation, resulting in a type of language X--, however hard he tries.

Help Vasya and add the punctuation marks so as to receive the valid type of language X--. Otherwise say that the task is impossible to perform.

Input

The first line contains a single integer n $(1 \le n \le 10^5)$, showing how many numbers the type dictated by Gena contains.

The second line contains space-separated words, said by Gena. Each of them is either "`pair`" or "`int`" (without the quotes).

It is guaranteed that the total number of words does not exceed 10^5 and that among all the words that Gena said, there are exactly n words "`int`".

Output

If it is possible to add the punctuation marks so as to get a correct type of language X-- as a result, print a single line that represents the resulting type. Otherwise, print "`Error occurred`" (without the quotes). **Inside the record of a type should not be any extra spaces and other characters**.

It is guaranteed that if such type exists, then it is unique.

Note that you should print the type dictated by Gena (if such type exists) and not any type that can contain n values.

Sample test(s)

input
3 pair pair int int int
output
pair<pair<int,int>,int>

input
1 pair int
output
Error occurred

D. Non-Secret Cypher

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

Berland starts to seize the initiative on the war with Flatland. To drive the enemy from their native land, the berlanders need to know exactly how many more flatland soldiers are left in the enemy's reserve. Fortunately, the scouts captured an enemy in the morning, who had a secret encrypted message with the information the berlanders needed so much.

The captured enemy had an array of positive integers. Berland intelligence have long been aware of the flatland code: to convey the message, which contained a number m, the enemies use an array of integers a. The number of its subarrays, in which there are at least k equal numbers, equals m. The number k has long been known in the Berland army so General Touristov has once again asked Corporal Vasya to perform a simple task: to decipher the flatlanders' message.

Help Vasya, given an array of integers a and number k, find the number of subarrays of the array of numbers a, which has at least k equal numbers.

Subarray $a[i... j]$ $(1 \le i \le j \le n)$ of array $a = (a_1, a_2, ..., a_n)$ is an array, made from its consecutive elements, starting from the i-th one and ending with the j-th one: $a[i... j] = (a_i, a_{i+1}, ..., a_j)$.

Input

The first line contains two space-separated integers n, k $(1 \le k \le n \le 4 \cdot 10^5)$, showing how many numbers an array has and how many equal numbers the subarrays are required to have, correspondingly.

The second line contains n space-separated integers a_i $(1 \le a_i \le 10^9)$ — elements of the array.

Output

Print the single number — the number of such subarrays of array a, that they have at least k equal integers.

Please do not use the `%lld` specifier to read or write 64-bit integers in C++. In is preferred to use the `cin`, `cout` streams or the `%I64d` specifier.

Sample test(s)

input
4 2 1 2 1 2
output
3

input
5 3 1 2 1 1 3
output
2

input
3 1 1 1 1
output
6

Note

In the first sample are three subarrays, containing at least two equal numbers: (1,2,1), (2,1,2) and (1,2,1,2).

In the second sample are two subarrays, containing three equal numbers: (1,2,1,1,3) and (1,2,1,1).

In the third sample any subarray contains at least one 1 number. Overall they are 6: (1), (1), (1), (1,1), (1,1) and (1,1,1).

E. Counter Attack

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

Berland has managed to repel the flatlanders' attack and is now starting the counter attack.

Flatland has n cities, numbered from 1 to n, and some pairs of them are connected by bidirectional roads. The Flatlandian maps show roads between cities if and only if there is in fact no road between this pair of cities (we do not know whether is it a clever spy-proof strategy or just saving ink). In other words, if two cities are connected by a road on a flatland map, then there is in fact no road between them. The opposite situation is also true: if two cities are not connected by a road on a flatland map, then in fact, there is a road between them.

The berlanders got hold of a flatland map. Now Vasya the Corporal is commissioned by General Touristov to find all such groups of flatland cities, that in each group of cities you can get from any city to any other one, moving along the **actual** roads. Also the cities from different groups are unreachable from each other, moving along the **actual** roads. Indeed, destroying such groups one by one is much easier than surrounding all Flatland at once!

Help the corporal complete this task and finally become a sergeant! Don't forget that a flatland map shows a road between cities if and only if there is in fact no road between them.

Input

The first line contains two space-separated integers n and m $(1 \le n \le 5 \cdot 10^5, 0 \le m \le 10^6)$ — the number of cities and the number of roads marked on the flatland map, correspondingly.

Next m lines contain descriptions of the cities on the map. The i-th line contains two integers a_i and b_i $(1 \le a_i, b_i \le n, a_i \ne b_i)$ — the numbers of cities that are connected by the i-th road on the flatland map.

It is guaranteed that each pair of cities occurs in the input no more than once.

Output

On the first line print number k — the number of groups of cities in Flatland, such that in each group you can get from any city to any other one by flatland roads. At the same time, the cities from different groups should be unreachable by flatland roads.

On each of the following k lines first print t_i $(1 \le t_i \le n)$ — the number of vertexes in the i-th group. Then print space-separated numbers of cities in the i-th group.

The order of printing groups and the order of printing numbers in the groups does not matter. The total sum t_i for all k groups must equal n.

Sample test(s)

input
4 4 1 2 1 3 4 2 4 3

output
2 2 1 4 2 2 3

input
3 1 1 2

output
1 3 1 2 3

Note

In the first sample there are roads only between pairs of cities 1-4 and 2-3.

In the second sample there is no road between cities 1 and 2, but still you can get from one city to the other one through city number 3.