



Win enough games

Task ID: wingames	Time limit: 0.1 s	Memory limit: 64 MB
-------------------	-------------------	---------------------

Several years ago I was playing Spider Solitaire on my computer all the time. Sometimes I won, sometimes I lost. Then I stopped playing Spider Solitaire and started to take part in programming competitions. After a few years I happened to start the old solitaire program again. I was pleased to discover that with the skill I gained during the years I am now able to win each and every game of Spider Solitaire. However, the program still remembers some of my previous games and thus the statistics don't necessarily reflect my current perfect skills.

The program displays the statistics in the following way:

Games played: p , games won: w (z %)

The number z is the percentage of games I won, rounded **down** to the nearest integer. For example, if $p = 53$ and $w = 47$, then $z = 88$. (In this example, the value w/p is roughly equal to 0.8868, which means that I won roughly 88.68% of the games I played. And 88.68% rounded down to the nearest integer is 88%.)

Find the smallest number of games I have to play and win in order to make z increase.

Input

The input consists of two lines. The first line contains the number p of games I played, the second line contains the number of games w I won ($0 \leq w \leq p$, $1 \leq p$).

There are four batches of test cases, each worth 25 points. Batch #1 has simple test cases with $p \leq 10^4$. Batches #2 to #4 have $p \leq 10^8$, $p \leq 10^{12}$, and $p \leq 10^{16}$, respectively.

Output

Output a single line with a single integer: the smallest positive integer g such that if I win the next g games, the displayed percentage z will be strictly larger than now.

If there is no such g , output -1 instead. (There will be no such test case in batch #1.)

Sample tests

input	output
10 8	1

So far I won 8 games out of 10, and thus the displayed success percentage is 80%. After I win the next game the displayed percentage will increase to 81%.

input	output
100 80	6

input	output
47 47	-1

Nothing left to improve here.



Bipartite subgraph

Task ID: bipartite	Time limit: 1.3 s	Memory limit: 64 MB
--------------------	-------------------	---------------------

An (*undirected*) *graph* is a pair (V, E) , where V is a finite set and E is a set containing some two-element subsets of V . The elements of V are called *vertices* and the elements of E are called *edges*. A graph can be imagined as a set of dots and a set of lines, each connecting a pair of dots.

A graph is *bipartite* if and only if it is possible to split its vertices into two sets V_1 and V_2 such that each edge connects a vertex in V_1 and a vertex in V_2 .

You are given a graph with n vertices and m edges. You want to change this graph into a bipartite graph. Is it possible to do this by erasing at most half of its edges?

Input

The first line of the input will contain the number of vertices n . For convenience, we will assume that $V = \{1, \dots, n\}$.

The second line of the input will contain the number of edges m .

Each of the remaining m lines will contain two space-separated integers: the vertices connected by one of the edges. (Each edge will connect two distinct vertices and each pair of vertices will be connected by at most one edge.)

There will be five batches of test data, each worth 20 points. Constraints:

Batch #1: $n, m \leq 10$. Batch #2: $n \leq 20, m \leq 190$. Batch #3: $n \leq 500, m \leq 3000$.

Batch #4: $n \leq 10\,000, m \leq 50\,000$. Batch #5: $n \leq 100\,000, m \leq 250\,000$.

Warning: For batch #5 it may be better to use fast I/O routines.

Output

If the goal cannot be reached, output a single line with the string “impossible” (quotes for clarity). Otherwise, pick and output any single solution – one possible set of edges that should be erased. The first line of output should contain the number k of erased edges. (Note that k must be at most equal to $m/2$.)

Each of the following k lines must contain two integers: the vertices connected by an edge we want to erase.

Note that it is **not necessary** to minimize k . Any valid solution will do.

Sample tests

input

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

output

```
2
3 1
1 4
```

The example output is not minimal. Erasing the edge 1-3 would have been enough.



The mean, the median and the mode

Task ID: mean	Time limit: 0.15 s	Memory limit: 64 MB
---------------	--------------------	---------------------

- The *mean* of a sequence $a_1, \dots, a_n$ is also called its arithmetic average. It is the sum of the numbers, divided by their count: $(a_1 + \dots + a_n)/n$.
- The *median* of a sequence of n elements, where n is odd, is the element of the sequence that would be exactly in the middle if the sequence were sorted in ascending order. For even n we will define the median as the smaller of the two elements in the middle.
- The *mode* of a sequence is the most frequent element. If there are multiple most frequent elements, the mode will be the smallest one of them.

For example, consider the sequence $(3, 1, 4, 1, 5, 9, 2, 6, 5, 3)$.

- Its mean is $(3 + 1 + 4 + 1 + 5 + 9 + 2 + 6 + 5 + 3)/10 = 3.9$.
- After sorting the sequence becomes $(1, 1, 2, 3, \underline{3}, 4, 5, 5, 6, 9)$, hence its median is 3.
- Three elements occur twice: 1, 3, and 5. The mode is the smallest one of them: 1.

Little Tomi cannot see the difference between these three concepts. Help him by producing a sequence where two of them will differ by as much as possible. More precisely, you will be given an integer n and two of the three concepts. You must produce a sequence of length n , and each element of the sequence must be an integer between 0 and 100, inclusive.

Input

The only line of input contains three space-separated tokens: a positive integer $n \leq 100$ and two different strings from the set `{mean, median, mode}`.

There are 10 batches of tests, each worth 10 points. Batches #1 to #3 test the pair median+mean, batches #4 and #5 test median+mode, and batches #6 to #10 test mean+mode.

Output

Output a single line with a sequence of n space-separated integers: the sequence for which the absolute difference between the two concepts is as large as possible. If there are multiple optimal sequences, you may output any of them.

Sample tests

input	output
1 mean mode	47

For $n = 1$, $a_1 = \text{mean} = \text{mode}$. Any value between 0 and 100 would have been correct.

input	output
3 mode median	0 99 100

The mode is 0, the median is 99, their difference is 99. A larger difference cannot be achieved.

**Sorted subsequences**

Task ID: sortedness	Time limit: 3.7 s	Memory limit: 64 MB
---------------------	-------------------	---------------------

A sequence is *sorted* if and only if it is non-increasing or non-decreasing (or both). That is, the sequences (47, 47, 47), (2, 3, 3, 4, 10), and (5, 4, 3, 1) are sorted, but (1, 2, 4, 3) is not.

Given a sequence S , its subsequence is any sequence that can be obtained by erasing some (possibly all, possibly none) elements of S . For example, the sequences $()$, (2, 3), (1, 1, 2, 4, 3) and (1, 2, 4, 3) are subsequences of $S = (1, 1, 2, 4, 3)$

We can randomly select a subsequence by flipping a fair coin for each element, and keeping the element if and only if the coin comes up heads. The *sortedness* of a sequence is the probability that its randomly selected subsequence is sorted.

Given a sequence, compute its sortedness as an exact fraction.

Input

The first line of the input contains the length of the sequence n . The second line contains the sequence: n space-separated non-negative integers $a_1, \dots, a_n$.

There will be 10 batches of tests, each worth 10 points. In batches #3 and #10 we have $a_i < 2^{60}$, in all other test cases $a_i < 10^6$. The maximum sizes of n in the ten batches of tests will not exceed the following values: (2, 9, 18, 26, 50, 70, 200, 750, 5000, 7500).

Output

Output a single line with the sortedness of the given sequence, shown as a fraction of the form “A/B”. There must be no spaces in the output, and the numbers A and B must be positive and relatively prime.

Sample tests

input	output
5 47 47 47 47 47	1/1 <i>All subsequences are sorted.</i>
5 20 10 40 50 30	9/16 <i>The output fraction must be reduced, the answer 18/32 will be rejected.</i>
10 1 2 3 4 5 6 0 8 9 10	263/512

Out of all 1024 subsequences, 526 are sorted: All 512 sequences that do not contain the 0 are sorted. (This includes the empty sequence.) Additionally, there are 14 sorted sequences that contain the 0: six decreasing and eight increasing ones. The answer is $526/1024 = 263/512$.