

**Odd one out**

Task ID: oddoneout	Time limit: 0.5 s	Memory limit: 64 MB
--------------------	-------------------	---------------------

Given is a sequence consisting of n integers, with n odd. Clearly, there has to be at least one integer that occurs in the sequence an odd number of times. Find one.

Input

The first line of the input contains the length of the sequence: a positive odd integer n .

The second line contains the sequence: n nonnegative space-separated integers $a_1, \dots, a_n$.

There are four batches of inputs, each worth 25 points.

- In batch 1 you may assume that $n < 1\,000$ and $\forall i : a_i < 2^{20} = 1\,048\,576$.
- In batch 2 you may assume that $n < 100\,000$ and $\forall i : a_i < 2^{20}$.
- In batch 3 you may assume that $n < 100\,000$ and $\forall i : a_i < 2^{30}$.
- In batch 4 you may assume that $n < 200\,000$ and $\forall i : a_i < 2^{60}$.

Output

Output a single line with a single integer. The integer must occur an odd number of times in the sequence given in the input. If there are multiple possibilities, pick any of them.

Sample tests

input	output
11 1 1 2 2 3 4 5 6 6 5 4	3
input	output
5 47 47 47 47 47	47
input	output
5 10 20 10 30 10	10

For this input both 20 and 30 are correct outputs as well.



Altering the distance

Task ID: distance	Time limit: 0.4 s	Memory limit: 64 MB
-------------------	-------------------	---------------------

The omniscient magic fairy Amálka is taking a trip on an infinite plane. The plane is divided into a grid of empty unit squares. Amálka is currently on the square (a_x, a_y) and plans to reach the square (b_x, b_y) . Amálka moves in steps. Each step takes one second, and it moves her to one of the four adjacent squares.

The evil wizard Zemčo plans to capture Amálka. He set up a trap at her destination. In order for the trap to work, Amálka must reach the destination at exactly the right moment.

Using an ancient spell, Zemčo can instantly grow large trees on any squares other than (a_x, a_y) and (b_x, b_y) . Once he summons the trees, Amálka will not be able to enter the affected squares. Zemčo must summon all trees before Amálka starts her journey.

Zemčo is certain that Amálka will always use a shortest path to reach her destination. Help him decide which squares have to be blocked. (And don't worry about the moral implications. Amálka is the hero in this story, and in the end she'll always outsmart Zemčo somehow.)

Input

The first line of the input contains two space-separated integers a_x and a_y .

The second line of the input contains two space-separated integers b_x and b_y .

You may assume that $-10^9 \leq a_x, a_y, b_x, b_y \leq 10^9$ and that $(a_x, a_y) \neq (b_x, b_y)$. Also, you may assume that without the trees Amálka can reach her destination in at most 2000 seconds.

In test cases worth 50 points, $0 \leq a_x, a_y, b_x, b_y \leq 100$.

The third line of the input contains a single positive integer s . Zemčo needs Amálka to reach her destination in exactly s seconds. The number s will not exceed 2000.

Output

If there is no way to satisfy Zemčo's goal, output a single line with the string "impossible".

If there are multiple solutions, pick and output any solution that uses at most 10 000 trees.

The first line of the output should contain the number t of trees to plant ($0 \leq t \leq 10\,000$). Each of the next t lines should contain two space-separated integers: the coordinates of a cell that should contain a tree. All coordinates must be between $-2 \cdot 10^9$ and $2 \cdot 10^9$, inclusive.

Sample tests

input

10 10
20 20
14

output

impossible
<i>Without trees, Amálka will reach her destination in 20 seconds. You have no way to make her reach the destination faster.</i>

input

10 10
10 50
42

output

2
3 5
10 47



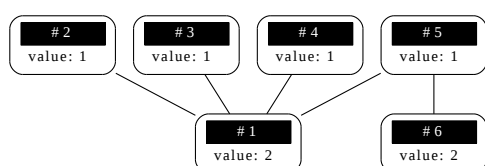
Testdata for Gems

Task ID: <code>gemtest</code>	Time limit: 0.5 s	Memory limit: 64 MB
-------------------------------	-------------------	---------------------

The following task, called Gems, was used at the Baltic Olympiad in Informatics 2003:

Given is a tree with $n \leq 10\,000$ vertices, labeled 1 through n . (A tree is a connected graph without cycles. A tree with n vertices has exactly $n - 1$ edges. See the figure below.)

Your task is to assign a positive integer **value** to each of its vertices. There are two conditions: First, for each edge the vertices it connects must have different values. Second, the **sum** of all n values you'll use must be as small as possible. (Note that we do not care about the largest value used, only the total value of the entire tree matters.)



The image shows a tree and one optimal way to assign the values to the vertices of that tree.

You will notice that the solution for the above tree only uses the values 1 and 2. In fact, all test data used at BOI 2003 could be solved optimally using only the values 1, 2, and 3.

Show that you are smarter than the problemsetters of this task. You will be given a positive integer k . Your task is to design any valid test case for the task Gems with the following two properties:

- It must have an optimal solution that only uses values 1 through k .
- It must not have an optimal solution that only uses values 1 through $k - 1$.

Input

The only line of the input contains the integer k , with $1 \leq k \leq 8$.

The values $k = 4$ and $k = 7$ are worth 20 points each. Each other value of k is worth 10 points. (There will be ten test cases, each worth 10 points. The values 4 and 7 will occur in two test cases each.)

Output

The first line of your output must contain a positive integer n : the number of vertices in the tree. Each of the remaining $n - 1$ lines must contain two integers, each between 1 and n , inclusive: a pair of vertices connected by an edge.

Sample tests

input

2

The sample output describes the tree shown in the picture above.

output

6
1 2
1 3
1 4
1 5
5 6



Doll nesting

Task ID: nesting	Time limit: 0.5 s	Memory limit: 64 MB
------------------	-------------------	---------------------

Most of you are probably familiar with *matryoshka dolls* – the famous Russian nesting dolls. A set of matryoshka dolls consists of several hollow wooden dolls of various sizes. The dolls can be opened and placed one inside another. In the assembled state, each doll, except for the smallest one, directly contains exactly one other doll.

During the last Christmas, little Vladko was staying at his grandparents' place. At the attic he discovered a suitcase full of old toys. Among the toys were many matryoshka dolls.

Sadly, the dolls Vladko discovered come from multiple old sets, and many of them have been lost. Still, some of the dolls fit into other dolls, and so it is possible to pack them somehow.

Your task is to find the best way of nesting the dolls – the number of visible dolls must be as small as possible.

Input

The first line of the input contains the number of dolls n . The dolls are numbered 1 through n . The i -th of the next n lines contains a string of n zeroes and ones. The j -th character in this string is 1 if doll i fits inside doll j , and 0 otherwise.

Obviously, a doll will never fit inside itself. Moreover, you may assume that *fitting inside* will always be *antisymmetric* (if j fits inside i , then i does not fit inside j) and *transitive* (if k fits inside j and j fits inside i , then k fits directly inside i).

The dolls have various shapes and sizes, therefore it is possible that for some pairs of dolls neither fits inside the other.

There will be five batches of test data, each worth 20 points. In these batches, n will be at most 8, 16, 50, 200, and 500, respectively.

Output

The first line of the output should contain the smallest positive integer k such that it is possible to nest the dolls in such a way that only k of them are visible at the end.

The remaining $n - k$ lines must contain one possible recipe how to do the nesting, starting with all dolls visible. The i -th of these lines must contain two integers x_i and y_i , meaning: “take the doll x_i and place it inside the doll y_i ”. At this moment, both dolls x_i and y_i must be visible and y_i must be empty.

Sample tests

input

3
000
000
000

output

3

No pair of dolls fits into one another, so there is nothing to do.



input

```
3
011
000
010
```

output

```
1
1 3
3 2
```

Doll 1 fits into each of the other dolls. The better solution is to put it inside doll 3, and then to put doll 3 inside doll 2.

input

```
4
0011
0011
0000
0000
```

output

```
2
1 4
2 3
```

Dolls 1 and 2 are “small” and dolls 3 and 4 are “large”. The best solution is to put each of the small dolls inside one of the large dolls.