A. As Simple as One and Two

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

You are given a non-empty string $s = s_1 s_2 \dots s_n$, which consists only of lowercase Latin letters. Polycarp does not like a string if it contains at least one string "one" or at least one string "two" (or both at the same time) as a **substring**. In other words, Polycarp does not like the string s if there is an integer j ($1 \le j \le n-2$), that $s_j s_{j+1} s_{j+2} =$"one" or $s_j s_{j+1} s_{j+2} =$"two".

For example:

- Polycarp does not like strings "oneee", "ontwow", "twone" and "oneonetwo" (they all have at least one substring "one" or "two"),
- Polycarp likes strings "oonnee", "twwwo" and "twnoe" (they have no substrings "one" and "two").

Polycarp wants to select a certain set of indices (positions) and remove all letters on these positions. All removals are made at the same time.

For example, if the string looks like $s =$"onetwone", then if Polycarp selects two indices 3 and 6, then "on**e**tw**o**ne" will be selected and the result is "ontwne".

What is the minimum number of indices (positions) that Polycarp needs to select to make the string liked? What should these positions be?

Input

The first line of the input contains an integer t ($1 \le t \le 10^4$) — the number of test cases in the input. Next, the test cases are given.

Each test case consists of one non-empty string s. Its length does not exceed $1.5 \cdot 10^5$. The string s consists only of lowercase Latin letters.

It is guaranteed that the sum of lengths of all lines for all input data in the test does not exceed $1.5 \cdot 10^6$.

Output

Print an answer for each test case in the input in order of their appearance.

The first line of each answer should contain r ($0 \le r \le |s|$) — the required minimum number of positions to be removed, where $|s|$ is the length of the given line. The second line of each answer should contain r different integers — the indices themselves for removal in any order. Indices are numbered from left to right from 1 to the length of the string. If $r = 0$, then the second line can be skipped (or you can print empty). If there are several answers, print any of them.

Examples

input

```
4
onetwone
testme
oneoneone
twotwo
```

output

```
2
6 3
0

3
4 1 7
2
1 4
```

input

```
10
onetwonetwooneooonetwooo
two
one
twooooo
ttttwo
ttwwoo
ooone
onnne
oneeeee
oneeeeeeetwooooo
```

output

```
6
18 11 12 1 6 21
1
1
1
3
1
2
1
6
0

1
4
0

1
1
2
1 11
```

Note

In the first example, answers are:

- "on**e**tw**o**ne",
- "testme" — Polycarp likes it, there is nothing to remove,
- "**o**ne**o**ne**o**ne",
- "**t**wo**t**wo".

In the second example, answers are:

- "**o**netw**o**netw**oo**neooo**n**e**t**wooo",
- "**t**wo",
- "**o**ne",
- "**t**wooooo",
- "tttt**w**o",
- "ttwwoo" — Polycarp likes it, there is nothing to remove,
- "ooo**n**e",
- "onnne" — Polycarp likes it, there is nothing to remove,
- "**o**neeeee",
- "**o**neeeeeee**t**wooooo".

B. Two Fairs

time limit per test: 3 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

There are n cities in Berland and some pairs of them are connected by two-way roads. It is guaranteed that you can pass from any city to any other, moving along the roads. Cities are numerated from 1 to n.

Two fairs are currently taking place in Berland — they are held in two different cities a and b ($1 \le a, b \le n$; $a \ne b$).

Find the number of pairs of cities x and y ($x \ne a, x \ne b, y \ne a, y \ne b$) such that if you go from x to y you will have to go through both fairs (the order of visits doesn't matter). Formally, you need to find the number of pairs of cities x, y such that any path from x to y goes through a and b (in any order).

Print the required number of pairs. The order of two cities in a pair does not matter, that is, the pairs (x, y) and (y, x) must be taken into account only once.

Input

The first line of the input contains an integer t ($1 \le t \le 4 \cdot 10^4$) — the number of test cases in the input. Next, t test cases are specified.

The first line of each test case contains four integers n, m, a and b ($4 \le n \le 2 \cdot 10^5$, $n - 1 \le m \le 5 \cdot 10^5$, $1 \le a, b \le n$, $a \ne b$) — numbers of cities and roads in Berland and numbers of two cities where fairs are held, respectively.

The following m lines contain descriptions of roads between cities. Each of road description contains a pair of integers u_i, v_i ($1 \le u_i, v_i \le n$, $u_i \ne v_i$) — numbers of cities connected by the road.

Each road is bi-directional and connects two different cities. It is guaranteed that from any city you can pass to any other by roads. There can be more than one road between a pair of cities.

The sum of the values of n for all sets of input data in the test does not exceed $2 \cdot 10^5$. The sum of the values of m for all sets of input data in the test does not exceed $5 \cdot 10^5$.

Output

Print t integers — the answers to the given test cases in the order they are written in the input.

Example

input

```
3
7 7 3 5
1 2
2 3
3 4
4 5
5 6
6 7
7 5
4 5 2 3
1 2
2 3
3 4
4 1
4 2
4 3 2 1
1 2
2 3
4 1
```

output

```
4
0
1
```

C. Beautiful Rectangle

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

You are given n integers. You need to choose a subset and put the chosen numbers in a beautiful rectangle (rectangular matrix). Each chosen number should occupy one of its rectangle cells, each cell must be filled with exactly one chosen number. Some of the n numbers may not be chosen.

A rectangle (rectangular matrix) is called beautiful if in each row and in each column all values are different.

What is the largest (by the total number of cells) beautiful rectangle you can construct? Print the rectangle itself.

Input

The first line contains n ($1 \le n \le 4 \cdot 10^5$). The second line contains n integers ($1 \le a_i \le 10^9$).

Output

In the first line print x ($1 \le x \le n$) — the total number of cells of the required maximum beautiful rectangle. In the second line print p and q ($p \cdot q = x$): its sizes. In the next p lines print the required rectangle itself. If there are several answers, print any.

Examples

input

```
12
3 1 4 1 5 9 2 6 5 3 5 8
```

output

```
12
3 4
1 2 3 5
3 1 5 4
5 6 8 9
```

input

```
5
1 1 1 1 1
```

output

```
1
1 1
1
```

D. Tree Elimination

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Vasya has a tree with n vertices numbered from 1 to n, and $n-1$ edges numbered from 1 to $n-1$. Initially each vertex contains a token with the number of the vertex written on it.

Vasya plays a game. He considers all edges of the tree by increasing of their indices. For every edge he acts as follows:

- If both endpoints of the edge contain a token, remove a token from one of the endpoints and write down its number.
- Otherwise, do nothing.

The result of the game is the sequence of numbers Vasya has written down. Note that there may be many possible resulting sequences depending on the choice of endpoints when tokens are removed.

Vasya has played for such a long time that he thinks he exhausted all possible resulting sequences he can obtain. He wants you to verify him by computing the number of distinct sequences modulo $998\,244\,353$.

Input

The first line contains a single integer n ($2 \le n \le 2 \cdot 10^5$) — the number of vertices of the tree.

The next $n-1$ lines describe edges of the tree. The i-th of these lines contains two integers u_i, v_i ($1 \le u_i, v_i \le n$) — endpoints of the edge with index i. It is guaranteed that the given graph is indeed a tree.

Output

Print a single integer — the number of distinct sequences modulo $998\,244\,353$.

Examples

input

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

output

```
5
```

input

```
7
7 2
7 6
1 2
7 5
4 7
3 5
```

output

```
10
```

Note

In the first sample case the distinct sequences are $(1), (2,1), (2,3,1), (2,3,4,1), (2,3,4,5)$.

Int the second sample case the distinct sequences are $(2,6,5,3), (2,6,5,7), (2,6,7,2), (2,6,7,5), (2,7,3), (2,7,5), (7,1,3), (7,1,5), (7,2,3), (7,2,5)$.

E. Four Stones

time limit per test: 1 second
memory limit per test: 512 megabytes
input: standard input
output: standard output

There are four stones on an infinite line in integer coordinates a_1, a_2, a_3, a_4. The goal is to have the stones in coordinates b_1, b_2, b_3, b_4. The order of the stones does not matter, that is, a stone from any position a_i can end up in at any position b_j, provided there is a required number of stones in each position (that is, if a coordinate x appears k times among numbers $b_1, \ldots, b_4$, there should be exactly k stones at x in the end).

We are allowed to move stones with the following operation: choose two stones at **distinct** positions x and y with at least one stone each, and move one stone from x to $2y-x$. In other words, the operation moves a stone to a symmetric position relative to some other stone. At any moment it is allowed to have any number of stones at the same position.

Find any sequence of operations that achieves the goal, or determine that it is impossible. The sequence does not have to be shortest, but it may contain at most 1000 operations.

Input

The first line contains four integers $a_1, \ldots, a_4$ ($-10^9 \le a_i \le 10^9$) — initial coordinates of the stones. There may be multiple stones sharing the same coordinate.

The second line contains four integers $b_1, \ldots, b_4$ ($-10^9 \le b_i \le 10^9$) — target coordinates of the stones. There may be multiple targets sharing the same coordinate.

Output

If there is no sequence of operations that achieves the goal, print a single integer -1. Otherwise, on the first line print a single integer k ($0 \le k \le 1000$) — the number of operations in your sequence. On the next k lines, describe the operations. The i-th of these lines should contain two integers x_i and y_i ($x_i \ne y_i$) — coordinates of the moved stone and the center of symmetry stone for the i-th operation.

For each operation i, there should at least one stone in each of the coordinates x_i and y_i, and the resulting coordinate $2y_i - x_i$ must not exceed 10^{18} by absolute value.

If there are multiple suitable sequences, print any of them. It is guaranteed that if there is a suitable sequence of operations, then there is also a suitable sequence that satisfies all the additional requirement.

Examples

input

```
0 1 2 3
3 5 6 8
```

output

```
3
1 3
2 5
0 3
```

input

```
0 0 0 0
1 1 1 1
```

output

```
-1
```

input

```
0 0 0 1
0 1 0 1
```

output

```
-1
```

F. Asterisk Substrings

time limit per test: 5 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Consider a string s of n lowercase English letters. Let t_i be the string obtained by replacing the i-th character of s with an asterisk character *. For example, when $s = \texttt{abc}$, we have $t_1 = *\texttt{bc}$, $t_2 = \texttt{a} * \texttt{c}$, and $t_3 = \texttt{ab}*$.

Given a string s, count the number of distinct strings of lowercase English letters and asterisks that occur as a substring of at least one string in the set $\{s, t_1, \ldots, t_n\}$. The empty string should be counted.

Note that *'s are just characters and do not play any special role as in, for example, regex matching.

Input

The only line contains a string s of n lowercase English letters ($1 \le n \le 10^5$).

Output

Print a single integer — the number of distinct strings of $s, t_1, \ldots, t_n$.

Example

input

```
abc
```

output

```
15
```

Note

For the sample case, the distinct substrings are (empty string), a, b, c, *, ab, a*, bc, b*, *b, *c, abc, ab*, a*c, *bc.