Codeforces Round #603 (Div. 2)

A. Sweet Problem

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

You have three piles of candies: red, green and blue candies:

- the first pile contains only red candies and there are r candies in it,
- the second pile contains only green candies and there are g candies in it,
- the third pile contains only blue candies and there are b candies in it.

Each day Tanya eats exactly two candies of different colors. She is free to choose the colors of eaten candies: the only restriction that she can't eat two candies of the same color in a day.

Find the maximal number of days Tanya can eat candies? Each day she needs to eat **exactly** two candies.

Input
The first line contains integer t ($1 \le t \le 1000$) — the number of test cases in the input. Then t test cases follow.

Each test case is given as a separate line of the input. It contains three integers r, g and b ($1 \le r, g, b \le 10^8$) — the number of red, green and blue candies, respectively.

Output
Print t integers: the i-th printed integer is the answer on the i-th test case in the input.

Example

input
6 1 1 1 1 2 1 4 1 1 7 4 10 8 1 4 8 2 8
output
1 2 2 10 5 9

Note
In the first example, Tanya can eat candies for one day only. She can eat any pair of candies this day because all of them have different colors.

In the second example, Tanya can eat candies for two days. For example, she can eat red and green candies on the first day, and green and blue candies on the second day.

In the third example, Tanya can eat candies for two days. For example, she can eat red and green candies on the first day, and red and blue candies on the second day. Note, that two red candies will remain uneaten.

B. PIN Codes

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

A PIN code is a string that consists of exactly 4 digits. Examples of possible PIN codes: `7013`, `0000` and `0990`. Please note that the PIN code can begin with any digit, even with `0`.

Polycarp has n ($2 \le n \le 10$) bank cards, the PIN code of the i-th card is p_i.

Polycarp has recently read a recommendation that it is better to set different PIN codes on different cards. Thus he wants to change the minimal number of digits in the PIN codes of his cards so that all n codes would become different.

Formally, in one step, Polycarp picks i-th card ($1 \le i \le n$), then in its PIN code p_i selects one position (from 1 to 4), and changes

the digit in this position to any other. He needs to change the minimum number of digits so that all PIN codes become different.

Polycarp quickly solved this problem. Can you solve it?

Input
The first line contains integer t ($1 \le t \le 100$) — the number of test cases in the input. Then test cases follow.

The first line of each of t test sets contains a single integer n ($2 \le n \le 10$) — the number of Polycarp's bank cards. The next n lines contain the PIN codes $p_1, p_2, \ldots, p_n$ — one per line. The length of each of them is 4. All PIN codes consist of digits only.

Output
Print the answers to t test sets. The answer to each set should consist of a $n + 1$ lines

In the first line print k — the least number of changes to make all PIN codes different. In the next n lines output the changed PIN codes in the order corresponding to their appearance in the input. If there are several optimal answers, print any of them.

Example

input
3 2 1234 0600 2 1337 1337 4 3139 3139 3139 3139
output
0 1234 0600 1 1337 1237 3 3139 3138 3939 6139

C. Everyone is a Winner!

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

On the well-known testing system MathForces, a draw of n rating units is arranged. The rating will be distributed according to the following algorithm: if k participants take part in this event, then the n rating is evenly distributed between them and rounded to the nearest lower integer, At the end of the drawing, an unused rating may remain — it is not given to any of the participants.

For example, if $n = 5$ and $k = 3$, then each participant will recieve an 1 rating unit, and also 2 rating units will remain unused. If $n = 5$, and $k = 6$, then none of the participants will increase their rating.

Vasya participates in this rating draw but does not have information on the total number of participants in this event. Therefore, he wants to know what different values of the rating increment are possible to get as a result of this draw and asks you for help.

For example, if $n = 5$, then the answer is equal to the sequence $0, 1, 2, 5$. Each of the sequence values (and only them) can be obtained as $\lfloor n/k \rfloor$ for some positive integer k (where $\lfloor x \rfloor$ is the value of x rounded down): $0 = \lfloor 5/7 \rfloor$, $1 = \lfloor 5/5 \rfloor$, $2 = \lfloor 5/2 \rfloor$, $5 = \lfloor 5/1 \rfloor$.

Write a program that, for a given n, finds a sequence of all possible rating increments.

Input
The first line contains integer number t ($1 \le t \le 10$) — the number of test cases in the input. Then t test cases follow.

Each line contains an integer n ($1 \le n \le 10^9$) — the total number of the rating units being drawn.

Output
Output the answers for each of t test cases. Each answer should be contained in two lines.

In the first line print a single integer m — the number of different rating increment values that Vasya can get.

In the following line print m integers **in ascending order** — the values of possible rating increments.

Example

input

4 5 11 1 3

output
4 0 1 2 5 6 0 1 2 3 5 11 2 0 1 3 0 1 3

D. Secret Passwords

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

One unknown hacker wants to get the admin's password of AtForces testing system, to get problems from the next contest. To achieve that, he sneaked into the administrator's office and stole a piece of paper with a list of n passwords — strings, consists of small Latin letters.

Hacker went home and started preparing to hack AtForces. He found that the system contains only passwords from the stolen list and that the system determines the equivalence of the passwords a and b as follows:

- two passwords a and b are equivalent if there is a letter, that exists in both a and b;
- two passwords a and b are equivalent if there is a password c from the list, which is equivalent to both a and b.

If a password is set in the system and an equivalent one is applied to access the system, then the user is accessed into the system.

For example, if the list contain passwords "a", "b", "ab", "d", then passwords "a", "b", "ab" are equivalent to each other, but the password "d" is not equivalent to any other password from list. In other words, if:

- admin's password is "b", then you can access to system by using any of this passwords: "a", "b", "ab";
- admin's password is "d", then you can access to system by using only "d".

Only one password from the list is the admin's password from the testing system. Help hacker to calculate the minimal number of passwords, required to **guaranteed** access to the system. Keep in mind that the hacker does not know which password is set in the system.

Input

The first line contain integer n ($1 \le n \le 2 \cdot 10^5$) — number of passwords in the list. Next n lines contains passwords from the list – non-empty strings s_i, with length at most 50 letters. Some of the passwords may be equal.

It is guaranteed that the total length of all passwords does not exceed 10^6 letters. All of them consist only of lowercase Latin letters.

Output

In a single line print the minimal number of passwords, the use of which will allow **guaranteed** to access the system.

Examples

input
4 a b ab d

output
2

input
3 ab bc abc

output
1

input
1 codeforces

output

```
1
```

Note
In the second example hacker need to use any of the passwords to access the system.

E. Editor

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

The development of a text editor is a hard problem. You need to implement an extra module for brackets coloring in text.

Your editor consists of a line with infinite length and cursor, which points to the current character. Please note that it points to only one of the characters (and not between a pair of characters). Thus, it points to an index character. The user can move the cursor left or right one position. If the cursor is already at the first (leftmost) position, then it does not move left.

Initially, the cursor is in the first (leftmost) character.

Also, the user can write a letter or brackets (either `(`, or `)`) to the position that the cursor is currently pointing at. A new character always overwrites the old value at that position.

Your editor must check, whether the current line is the `correct text`. Text is correct if the brackets in them form the *correct bracket sequence*.

Formally, correct text (CT) must satisfy the following rules:

- any line without brackets is CT (the line can contain whitespaces);
- If the first character of the string — is `(`, the last — is `)`, and all the rest form a CT, then the whole line is a CT;
- two consecutively written CT is also CT.

Examples of correct texts: `hello(codeforces)`, `round`, `((i)(write))edi(tor)s`, `( me)`. Examples of incorrect texts: `hello)oops(`, `round)`, `((me)`.

The user uses special commands to work with your editor. Each command has its symbol, which must be written to execute this command.

The correspondence of commands and characters is as follows:

- L — move the cursor one character to the left (remains in place if it already points to the first character);
- R — move the cursor one character to the right;
- any lowercase Latin letter or bracket (`(` or `)`) — write the entered character to the position where the cursor is now.

For a complete understanding, take a look at the first example and its illustrations in the note below.

You are given a string containing the characters that the user entered. For the brackets coloring module's work, after each command you need to:

- check if the current text in the editor is a correct text;
- if it is, print the least number of colors that required, to color all brackets.

If two pairs of brackets are nested (the first in the second or vice versa), then these pairs of brackets should be painted in *different* colors. If two pairs of brackets are not nested, then they can be painted in different or the same colors. For example, for the bracket sequence `()(())()()` the least number of colors is 2, and for the bracket sequence `(()(()())())(())` — is 3.

Write a program that prints the minimal number of colors after processing each command.

Input
The first line contains an integer n ($1 \le n \le 10^6$) — the number of commands.

The second line contains s — a sequence of commands. The string s consists of n characters. It is guaranteed that all characters in a string are valid commands.

Output
In a single line print n integers, where the i-th number is:

- -1 if the line received after processing the first i commands is not valid text,
- the minimal number of colors in the case of the correct text.

Examples

input
11 (RaRbR)L)L(
output
-1 -1 -1 -1 -1 -1 1 1 -1 -1 2

input
11 (R)R(R)Ra)c
output
-1 -1 1 1 -1 -1 1 1 1 -1 1

Note

In the first example, the text in the editor will take the following form:

1. (
   ^
2. (
    ^
3. (a
    ^
4. (a
     ^
5. (ab
     ^
6. (ab
      ^
7. (ab)
      ^
8. (ab)
      ^
9. (a))
      ^
10. (a))
     ^
11. (())
     ^

F. Economic Difficulties

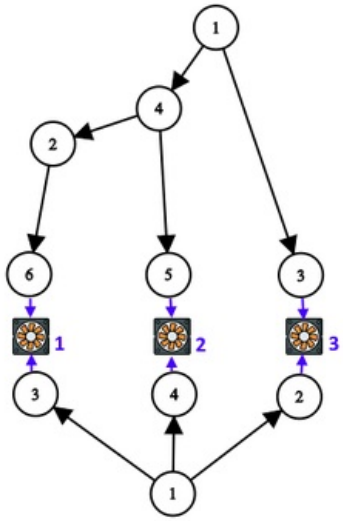
time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

An electrical grid in Berland palaces consists of 2 grids: main and reserve. Wires in palaces are made of expensive material, so selling some of them would be a good idea!

Each grid (main and reserve) has a head node (its number is 1). Every other node gets electricity from the head node. Each node can be reached from the head node by a unique path. Also, both grids have exactly n nodes, which do not spread electricity further.

In other words, every grid is a rooted directed tree on n leaves with a root in the node, which number is 1. Each tree has independent enumeration and nodes from one grid are not connected with nodes of another grid.

Also, the palace has n electrical devices. Each device is connected with one node of the main grid and with one node of the reserve grid. Devices connect only with nodes, from which electricity is not spread further (these nodes are the tree's leaves). Each grid's leaf is connected with exactly one device.



In this example the main grid contains 6 nodes (the top tree) and the reserve grid contains 4 nodes (the lower tree). There are 3 devices with numbers colored in blue.

It is guaranteed that the whole grid (two grids and n devices) can be shown in this way (like in the picture above):

- main grid is a top tree, whose wires are directed 'from the top to the down',
- reserve grid is a lower tree, whose wires are directed 'from the down to the top',
- devices — horizontal row between two grids, which are numbered from 1 to n from the left to the right,
- wires between nodes do not intersect.

Formally, for each tree exists a depth-first search from the node with number 1, that visits leaves in order of connection to devices $1, 2, \ldots, n$ (firstly, the node, that is connected to the device 1, then the node, that is connected to the device 2, etc.).

Businessman wants to sell (remove) **maximal** amount of wires so that each device will be powered from at least one grid (main or reserve). In other words, for each device should exist at least one path to the head node (in the main grid or the reserve grid), which contains only nodes from one grid.

Input

The first line contains an integer n ($1 \le n \le 1000$) — the number of devices in the palace.

The next line contains an integer a ($1 + n \le a \le 1000 + n$) — the amount of nodes in the main grid.

Next line contains $a - 1$ integers p_i ($1 \le p_i \le a$). Each integer p_i means that the main grid contains a wire from p_i-th node to $(i + 1)$-th.

The next line contains n integers x_i ($1 \le x_i \le a$) — the number of a node in the main grid that is connected to the i-th device.

The next line contains an integer b ($1 + n \le b \le 1000 + n$) — the amount of nodes in the reserve grid.

Next line contains $b - 1$ integers q_i ($1 \le q_i \le b$). Each integer q_i means that the reserve grid contains a wire from q_i-th node to $(i + 1)$-th.

The next line contains n integers y_i ($1 \le y_i \le b$) — the number of a node in the reserve grid that is connected to the i-th device.

It is guaranteed that each grid is a tree, which has exactly n leaves and each leaf is connected with one device. Also, it is guaranteed, that for each tree exists a depth-first search from the node 1, that visits leaves in order of connection to devices.

Output

Print a single integer — the maximal amount of wires that can be cut so that each device is powered.

Examples

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

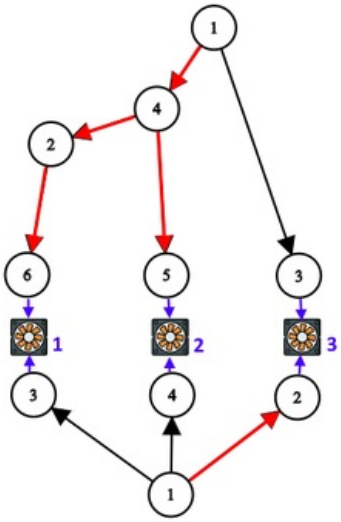
input
4 6 4 4 1 1 1 3 2 6 5 6 6 6 1 1 1 5 4 3 2

output
6

input
5 14 1 1 11 2 14 14 13 7 12 2 5 6 1 9 8 3 10 4 16 1 1 9 9 2 5 10 1 14 3 7 11 6 12 2 8 16 13 4 15
output
17

Note

For the first example, the picture below shows one of the possible solutions (wires that can be removed are marked in red):



The second and the third examples can be seen below: