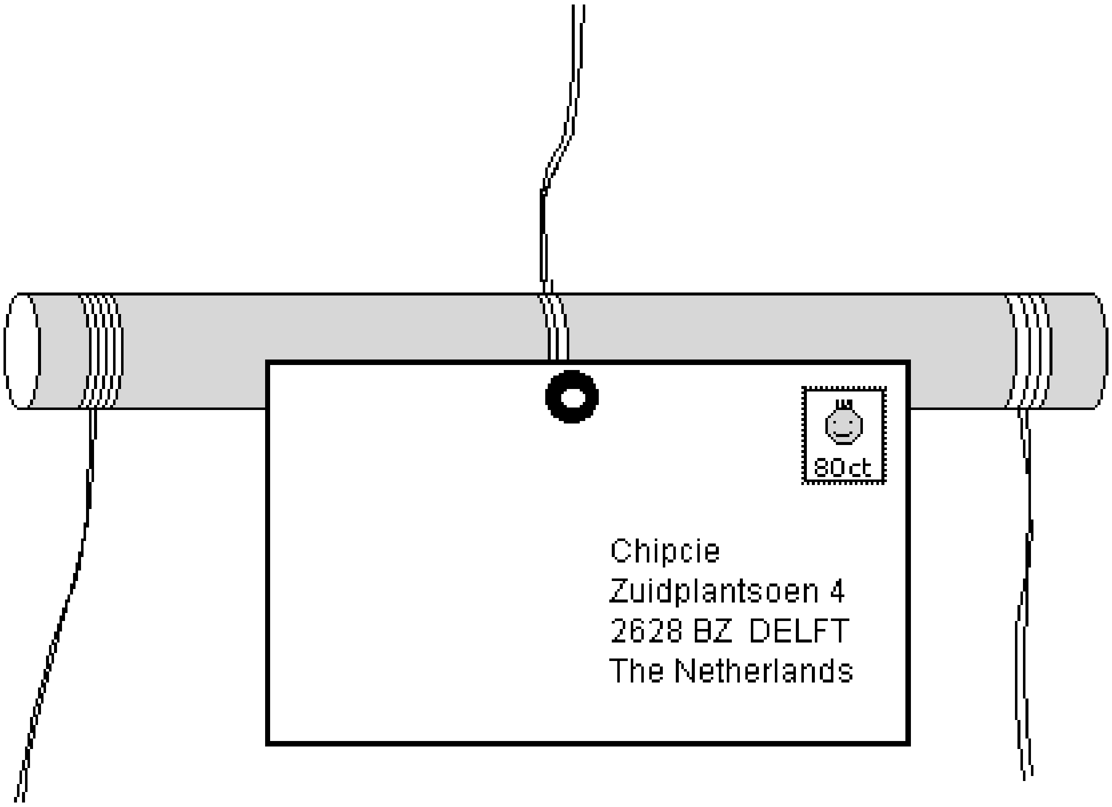
## 564 Gaston

Gaston is an employee of a famous comic magazine. One of his tasks (the one he dislikes most) is archiving post. He has some delay in archiving, so the larger part of his desk is occupied by an enormous pile of unprocessed letters. Sometimes one of the editors needs a certain letter. Then Gaston is faced with the problem to find that letter in the pile.




Gaston realised that he has a retrieval problem. Using sticks and threads he designed a system for better retrieving not archived letters. In the middle of a stick he fixes a letter. At both ends of the stick a thread is fixed where another stick (with letter) can be fastened, etc. He arranges the letters such that all letters pending under the left hand thread of a stick are alphabetically before the letter in the middle. The letters pending under the right hand thread are alphabetically after the letter in the middle. This is true for every stick, so Gaston obtained a sorted tree, in which it is easy to find a letter.

Making a prototype, Gaston finds out that he has a balancing problem. If there is a difference in weight between the ends of a stick, the stick is unbalanced, and the whole system collapses in a mess even bigger than before. Using some extra thread, Gaston invents an asymmetric fix: now the stick is still balanced if the left hand side contains one more letter than the right hand side. If the right hand side contains more letters, the stick is always unbalanced, so the fix is indeed asymmetric.

Opinions about Gaston are different. He believes to be a genius himself, the others don't. The editors point out that when a letter is added there is a fair chance that the tree gets out of balance and has to be rearranged. We want you to find out how often the tree (or a subtree) must be rearranged, given a tree and a sequence of incoming letters.

We have simplified the problem a bit. All letters have the same weight. Every letter has a unique (positive) number. The letters have to be ordered by increasing number. A stick is balanced if the left hand side contains the same number of letters as the right hand side, or at most one more.

A letter is added according to the following procedure. If its number is lower than the number of the letter on the stick, it is added to the left hand side of the balance. If its number is higher, it is added to the right hand side. Following this procedure repeatedly, eventually a free thread is found. Then the letter is fixed on a fresh stick (having two free threads, one at each end) which is tied to the free thread. Then, starting at the top working downwards, all sticks are checked for being balanced. If a stick is unbalanced, a balancing step is executed. Because a balancing step might result in having other sticks being unbalanced again, the whole tree is checked again from top to bottom for unbalanced sticks. This is done until no unbalanced sticks are found.

A balancing step is defined as bringing one stick $s$, being unbalanced, in a balancing state. This is done by removing one letter on the side of the stick which is too heavy and adding one on the other side. This is done according to the following procedure:

- first remove the letter $l_old$ at the stick $s$ itself
- then, find an appropriate letter on the side which is too heavy and fix this one on the stick $s$
- finally add the former letter $l_old$ on the other side.

A balancing step can result in more sticks being unbalanced! Because the new letter $l_new$ at the stick is removed from it's old position, an unbalanced stick can be left behind. Also adding the former letter can result in having a new unbalanced stick. Getting these sticks balanced again is not a part of this balancing step(!), but requires new balancing steps.

The problem is to find out how many balancing steps are needed to keep the tree balanced when new letters are added.

## Notation

At the end of a thread we may find:

- a stick, with a letter, and a thread at both ends, or
- nothing

This leads to the following textual representation of a balance.

- if it is a stick: the number of the letter on it, followed by the representation of what is at the end of its left hand thread, followed by the representation of what is at the end of its right hand thread.
- if it is a loose thread: the number 0.

Within a line numbers are separated with at least one space. When needed the textual representation of a balance may run over several lines. Every line contains at least one number.

## Input

A line with the number $n$ of problems, then $n$ times:

- a description of a balance (see notation)
- on a separate line the number $b$ of letters to be added, then, beginning on the next line
- $b$ times the number of a letter $l$ ($1 \leq l \leq 1000$).

Within a line the numbers of the letters to insert are separated with at least one space. The numbers of the letters may run over several lines. Every line contains at least one number.

## Output

The output consists of $n$ lines. Each line contains the number of balancing steps needed to keep the tree balanced when the letters are added.

## Sample Input

```
2
3 2 0 0
5 0 0
4
1 6 7
9
400 250 0 0 0
2
511 100
```

## Sample Output

```
3
0
```