

## Git Status (git)

Today William was looking for a simple solution to backup his entire hard drive. Basically, he wants to make a full copy of the entire file system structure, starting from the root / and copying every subdirectory, recursively. To do this, normally you would use something like *rsync*, however he decided that *git* was the best software for the job<sup>1</sup>.

The way *git* works is that it monitors which files are changed, and at any point in time you can choose to *commit* the latest changes and finally *push* them to a remote repository: backup done!

To make his life easier, William wants to write a program that will show which files were changed since the last commit.

Yes, there is a *git* command for it (*git status*) but it's not really optimal: if a folder **only** contains files that were modified, the command will still list each individual file, like this:

```
/some-directory/some-file
/some-directory/some-other-file
```

In such a case, William prefers to only list the directory which contains only-modified files:

```
/some-directory
```



Formally, you will be given a list of  $N$  files, each with its full path (starting with the / character) and an indication of whether that file was modified or not. William wants a program that, from this list, will produce a new “compressed” list containing only the paths that changed, and, if a folder contains **only** files that changed, then the list should only mention the folder, and not the files inside of it.

**Note:** when we say that a folder “contains” files we are referring also to the files that exist in subdirectories. For example, if you have these files:

```
/dir-A/this-file-was-not-changed
/dir-A/this-file-was-changed
/dir-B/sub-dir/this-file-was-changed
/dir-B/sub-dir/this-file-was-also-changed
```

Then the “compressed” list of changed files would simply be:

```
/dir-A/this-file-was-changed
/dir-B
```

Since “/dir-B” only contains modified files (even if they are not direct children) we can safely omit the “/sub-dir” part. Notice, also, that if the “/dir-A/this-file-was-not-changed” file were to be modified too, then we would simply output “/” as the compressed list: neat!

Among the attachments of this task you may find a template file `git.*` with a sample incomplete implementation.

<sup>1</sup>It's worth mentioning that he has lots of heavy files, like music and movies, but Edoardo reassured him that *git* is absolutely the right tool for big non-text files.

## Input

The first line contains the only integer  $N$ . Each of the following  $N$  lines is formed by a number  $M_i$  (equal to 1 if the file was modified, 0 if it was not) and a string  $P_i$  (the full path of the file).

## Output





You need to write one line for each entry of the “compressed” list of modified files, as shown in the examples. The lines should be sorted lexicographically (i.e. in the same relative order as the input).

## Constraints

- The paths  $P_i$  in input are provided in lexicographical order.
- $1 \leq N \leq 10\,000$ , and the length of each  $P_i$  is between 2 and 100 000 characters.
- $M_i$  is either 0 or 1.
- $P_i$  only contains lowercase letters, dashes (-), and slashes (/). There are no spaces.
- $P_i$  is a valid path: it starts with a slash, and it does not contain two consecutive slashes.
- $P_i$  always identifies a *file*, i.e. there are no empty folders.

## Scoring

Your program will be tested against several test cases grouped in subtasks. In order to obtain the score of a subtask, your program needs to correctly solve all of its test cases.

- **Subtask 1** (0 points)      Examples.  

- **Subtask 2** (10 points)      Each path has **only one** slash (there are no subdirectories).  

- **Subtask 3** (30 points)      Each path has **up to two** slashes (max nesting level is 1).  

- **Subtask 4** (60 points)      No additional limitations.  


## Examples

input	output
4 1 /dirA/this-was-changed 0 /dirA/this-was-not-changed 1 /dirB/sub/this-was-changed 1 /dirB/sub/this-was-changed-too	/dirA/this-was-changed /dirB
4 1 /dirA/this-was-changed 1 /dirA/this-was-now-changed 1 /dirB/sub/this-was-changed 1 /dirB/sub/this-was-changed-too	/

## Explanation

The two examples are equivalent to the ones presented in the problem statement.