ACM-ICPC Live Archive

7554 git

Farzi Coder has been assigned an important coding assignment by his boss *Programmer Bhai*. Now, *Farzi Coder* has done coding his project and has to submit it to a central git repository. In git, before committing we have to add the files which we want to commit to the stage. We can add (stage) or remove (unstage) files to/from the stage. Initially the stage is empty. *Programmer Bhai* is very particular about what goes into the repository and what doesn't. He has given a list of files alongwith whether he wants the file in the stage or not. e.g.

```
stage /code/lib/acpp
stage /code/lib/bcpp
unstage /code/lib/aout
```

This means we want `/code/lib/acpp` and `/code/lib/bcpp` to be in the stage while `/code/lib/aout` should not be in the stage. There are 2 git commands to control staging:

`git add` $< path >$
`git remove` $< path >$

The path here can be a path to a file or a path to a directory. If the path is to a directory, the command applies to all the files and subdirectories inside that directory recursively. *Farzi Coder* wants to commit the code *asap*. What is the minimum number of git commands needed to add all the required files to stage ***while making sure none of the unstage files are in the stage.***

Input

The first line of the input has the number T denoting the number of test cases.

The first line of each test case has the number N denoting the number of files in this case.

Each of the next N lines contains a word denoting whether the following file should be on the stage or not followed by the path to the file.

i.e. the 2 types of input will be '`stage` $< path >$' and '`unstage` $< path >$'.

Output

For each test case, output the number of minimum git commands needed, on a line by itself.

Constraints:

- $1 \leq T \leq 10$
- $1 \leq N \leq 100$
- **The length of each path doesn't exceed 100**
- **Each path is a valid path to a file. Each path input follows these rules:**
  - It doesn't end with a forward slash (`/`)
  - Each path starts from the root directory (`/`)
  - There are no two consecutive forward slashes
  - Each file or directory name consists only of lowercase alphabets
  - All paths in a test case are distinct

– There will be no file — directory conflict. e.g. consider `/a/b/c` and `/a/b`. Here '`b`' is both a file and a directory.

Explanation:

- **Case 1:** The following 2 commands are enough:

  ```
  git add /
  git remove /code/lib/aout
  ```

  The first command adds the entire project to the stage while the second command removes aout from the stage as required.

- **Case 2:** As initially there are no files in the stage, no commands are necassary.
- **Case 3:** We add all the files to stage using '`git add /`' or '`git add /code/lib/`'

Sample Input

```
3
3
stage /code/lib/acpp
stage /code/lib/bcpp
unstage /code/lib/aout
2
unstage /code/lib/acpp
unstage /code/lib/bcpp
3
stage /code/lib/acpp
stage /code/lib/bcpp
stage /code/lib/aout
```

Sample Output

```
2
0
1
```