3188 Pathological Paths

Professor Pathfinder is a distinguished authority on the structure of hyperlinks in the World Wide Web. For establishing his hypotheses, he has been developing software agents, which automatically traverse hyperlinks and analyze the structure of the Web. Today, he has gotten an intriguing idea to improve his software agents. However, he is very busy and requires help from good programmers. You are now being asked to be involved in his development team and to create a small but critical software module of his new type of software agents.

Upon traversal of hyperlinks, Pathfinder's software agents incrementally generate a map of visited portions of the Web. So the agents should maintain the list of traversed hyperlinks and visited web pages. One problem in keeping track of such information is that two or more different URLs can point to the same web page. For instance, by typing any one of the following five URLs, your favorite browsers probably bring you to the same web page, which as you may have visited is the home page of the ACM ICPC Ehime contest.

```
http://www.ehime-u.ac.jp/ICPC/
http://www.ehime-u.ac.jp/ICPC
http://www.ehime-u.ac.jp/ICPC/../ICPC/
http://www.ehime-u.ac.jp/ICPC/./
http://www.ehime-u.ac.jp/ICPC/index.html
```

Your program should reveal such aliases for Pathfinder's experiments.

Well, . . . but it were a real challenge and to be perfect you might have to embed rather complicated logic into your program. We are afraid that even excellent programmers like you could not complete it in five hours. So, we make the problem a little simpler and subtly unrealistic. You should focus on the path parts (i.e. `/ICPC/`, `/ICPC`, `/ICPC/../ICPC/`, `/ICPC/./`, and `/ICPC/index.html` in the above example) of URLs and ignore the scheme parts (e.g. `http://`), the server parts (e.g. `www.ehime-u.ac.jp`), and other optional parts. You should carefully read the rules described in the sequel since some of them may not be based on the reality of today's Web and URLs.

Each path part in this problem is an absolute pathname, which specifies a path from the root directory to some web page in a hierarchical (tree-shaped) directory structure. A pathname always starts with a slash (/), representing the root directory, followed by path segments delimited by a slash. For instance, `/ICPC/index.html` is a pathname with two path segments `ICPC` and `index.html`.
All those path segments but the last should be directory names and the last one the name of an ordinary file where a web page is stored. However, we have one exceptional rule: an ordinary file name `index.html` at the end of a pathname may be omitted. For instance, a pathname `/ICPC/index.html` can be shortened to `/ICPC/`, if `index.html` is an existing ordinary file name. More precisely, if `ICPC` is the name of an existing directory just under the root and index.html is the name of an existing ordinary file just under the `/ICPC` directory, `/ICPC/index.html` and `/ICPC/` refer to the same web page. Furthermore, the last slash following the last path segment can also be omitted. That is, for instance, `/ICPC/` can be further shortened to `/ICPC`. However, `/index.html` can only be abbreviated to `/` (a single slash).

You should pay special attention to path segments consisting of a single period (`.`) or a double period (`..`), both of which are always regarded as directory names. The former represents the directory itself and the latter represents its parent directory. Therefore, if `/ICPC/` refers to some web page, both

`/ICPC/./` and `/ICPC/../ICPC/` refer to the same page. Also `/ICPC2/../ICPC/` refers to the same page if `ICPC2` is the name of an existing directory just under the root; otherwise it does not refer to any web page. Note that the root directory does not have any parent directory and thus such pathnames as `/../` and `/ICPC/../../index.html` cannot point to any web page.

Your job in this problem is to write a program that checks whether two given pathnames refer to existing web pages and, if so, examines whether they are the same.

Input

The input consists of multiple datasets. The first line of each dataset contains two positive integers N and M, both of which are less than or equal to 100 and are separated by a single space character.

The rest of the dataset consists of $N + 2M$ lines, each of which contains a syntactically correct pathname of at most 100 characters. You may assume that each path segment enclosed by two slashes is of length at least one. In other words, two consecutive slashes cannot occur in any pathname. Each path segment does not include anything other than alphanumerical characters (i.e. '`a`'-'`z`', '`A`'-'`Z`', and '`0`'-'`9`') and periods ('`.`').

The first N pathnames enumerate all the web pages (ordinary files). Every existing directory name occurs at least once in these pathnames. You can assume that these pathnames do not include any path segments consisting solely of single or double periods and that the last path segments are ordinary file names. Therefore, you do not have to worry about special rules for `index.html` and single/double periods. You can also assume that no two of the N pathnames point to the same page.

Each of the following M pairs of pathnames is a question: do the two pathnames point to the same web page? These pathnames may include single or double periods and may be terminated by a slash. They may include names that do not correspond to existing directories or ordinary files.

Two zeros in a line indicate the end of the input.

Output

For each dataset, your program should output the M answers to the M questions, each in a separate line. Each answer should be '`yes`' if both point to the same web page, '`not found`' if at least one of the pathnames does not point to any one of the first N web pages listed in the input, or '`no`' otherwise.

Sample Input

```
5 6
/home/ACM/index.html
/ICPC/index.html
/ICPC/general.html
/ICPC/japanese/index.html
/ICPC/secret/confidential/2005/index.html
/home/ACM/
/home/ICPC/../ACM/
/ICPC/secret/
/ICPC/secret/index.html
/ICPC
/ICPC/../ICPC/index.html
/ICPC
/ICPC/general.html
/ICPC/japanese/.././
/ICPC/japanese/./../
/home/ACM/index.html
```

```
/home/ACM/index.html/
1 4
/index.html/index.html
/
/index.html/index.html
/index.html
/index.html/index.html
/..
/index.html/../..
/index.html/
/index.html/index.html/..
0 0
```

Sample Output

```
not found
not found
yes
no
yes
not found
not found
yes
not found
not found
```