DMOPC '16 Contest 2 P6 - Console Simulator 2017

Time limit: 1.2s **Memory limit:** 128M

Gigel is back at it with a new challenge. He wants to create a new game for his friends to play. However, this kind of project is hard for a single person to handle, so he asks you for help on the backend for the console system. He gives you his work paper regarding the story part of this game:

> Name: Console Simulator 2017
> Genre: Simulator, Action, Horror
> Description: You are trapped in a room, where the only working thing is a computer. As you start poking around, you find out that the console you are sitting at is controlling the whole building, and also your life. Prepare yourself for a run for your life while writing in the console.

The system that you must design has to simulate a console. It has to emulate a console similar to a UNIX one. Here is the list of commands that you have to emulate, and the details for each one:

- `ls` - Lists the subdirectories and the files contained by the current folder. If it has the `-r` argument, it will list the subdirectories and files recursively.
- `cd` - Changes the directory to the specified one. If the path starts with `~/`, you will have an absolute path.
- `grep` - Searches through a list and makes a result list. The search query is a regex subset. It will only use the `^`, `$` and `.` special characters. The `^` and `$` are position markers, which can be put at the beginning or the end of the query. The `.` character is considered a wildcard, and can be replaced by any other character. See the sample for more details.
- `mkdir` - Creates a new directory in the current folder.
- `touch` - Creates a new file in the current folder.
- `pwd` - Prints the current path.
- `exit` - Exits the console.

Input Specification

Firstly, you will receive the initial folder structure.
On the first line, you will have a number N.
On the next N lines, you will find the current depth and the name of the file/folder.
After this, you will find commands, and you will stop reading when you reach the `exit` command.

Output Specification

The outputs from the commands are always separated by one empty line. After the last command, be sure to also have an additional blank line.

Constraints and Notes

- $0 \leq N \leq 105\,000$

- Each folder will have at most 1000 subdirectories and 1000 files.
- The maximum depth will be 100.
- In a folder there will not be two subdirectories or files with the same name.
- A file always has an extension. (ends with `.<ext>`)
- The output of each command must be printed sorted lexicographical.
- Always print `\n\n` after the output of a command, even if the command does not print anything.
- If you don't print `\n\n` after the output from every command that gives output, you will get `WA`.
- Be sure to ask in the comments if you don't find a piece of information.

Sample Input

```
35
0 usr
1 home
2 homespace
3 desktop
3 helper
2 derringer
3 games
4 steam
5 magico
6 map
7 part1.map
7 part2.map
6 exe
7 steam_api.dll
7 magico.exe
5 europauniversalis4
6 history
7 data.txt
6 map
7 map.dat
6 common
3 nothere
3 yarp
0 system
1 util
2 lsgrepandcd
3 nofiles
2 gcc
2 g++
1 xgraphicssystem
2 source
2 binariesbin
3 server.exe
2 gource
2 orrel
cd usr
cd home
cd derringer
ls
ls -r
ls | grep a
ls -r | grep a
cd games
ls
pwd
```

```
cd ~/
ls -r | grep \.exe$
ls -r | grep ^g...
ls -r | grep ^g..$
exit
```

Sample Output

```
games
nothere
yarp

common
data.txt
europauniversalis4
exe
games
history
magico
magico.exe
map
map
map.dat
nothere
part1.map
part2.map
steam
steam_api.dll
yarp

games
yarp

data.txt
europauniversalis4
games
magico
magico.exe
map
map
map.dat
part1.map
part2.map
steam
steam_api.dll
yarp

steam

~/usr/home/derringer/games/

magico.exe
server.exe
```

```
games
gource

g++
gcc
```