Problem 265: Permissions Check

Difficulty: Medium

Author: Adrienne McKee, Huntsville, Alabama, United States

Originally Published: Code Quest 2025

Problem Background

File permissions are an essential piece to the security model used by Linux systems. They determine who can access files and folders (or directories) on a system and how. On the Linux system files and folders the owner, group, and others are granted access based on the permission. We want to make sure that the permissions follow the Least Privilege principle, meaning that everyone has the access they need, but nothing more than that.

As part of your internship with Lockheed Martin's Corporate Information Security division, you have been tasked with writing a program to determine if all of the files or directories in a given folder contain the correct permissions, meaning they follow the Least Privilege principle. The input to your program will be a cleaned-up listing of a directory using the `ls -lAR` command which will recursively find all files or directories and print out their permissions.

Problem Description

When the command `ls -l` is executed from a Linux command line, the output is a group of metadata that includes the permissions on each file.

```
-rw-rw-r--. 1 user group 10 Feb 26 07:08 file1
-rw-rw-r--. 1 user group 10 Feb 26 2017 file2
```

Here are the components of the listing, separated by spaces:

- File Attributes:
  - File type: 1 character describing the type: - (file), or d (directory)
  - Permission settings: 9 characters in groups of three; each group contains "rwx" in that order, although each letter can be replaced with a dash (-). "r" stands for read, "w" for write, and "x" for execute; a "-" means that permission is denied.
    - Characters 1-3 are the owner's permissions.
    - Characters 4-6 are the group's permissions.
    - Characters 7-9 are the others' (not the owner or member of the group) permissions.
  - Extended attributes: For this example, will always be a dot (.)
- Number of Hard Links
- User owner: user
- Group owner: group

- File size, usually in MB
- The date and time at which the file was created/modified
- Name of the file

You'll need to inspect each file and directory you've been asked to check and compare their permissions to what we want set. The permissions must match exactly; if a user or group has either more or fewer permissions than they should, it's incorrect and must be fixed.

Sample Input

The first line of your program's input, received from the standard input channel, will contain a positive integer representing the number of test cases. Each test case will include the following values separated by spaces:

A single line containing the following information, separated by spaces:

- An uppercase letter, indicating whether to evaluate only Files (F) or Directories (D).
- Three characters, representing the desired owner Permissions (a lowercase "r" or a dash, then a lowercase "w" or a dash, then a lowercase "x" or a dash).
- Three characters, representing the desired group Permissions (a lowercase "r" or a dash, then a lowercase "w" or a dash, then a lowercase "x" or a dash).
- Three characters, representing the desired other user Permissions (a lowercase "r" or a dash, then a lowercase "w" or a dash, then a lowercase "x" or a dash).
- A positive integer, N, indicating the number of lines of input that follow.

The next N lines contain the details for each directory you need to review. Each directory will be identified by its relative path, followed by a colon (:); all relative paths will start with "./". Following that will be one or more lines containing the `ls -l` output from that directory, as outlined above.

```
2
F r-- r-- r-- 5
./lookHere:
drwxrwxrwx. 1 user group 10 Feb 26 2017 directory1
-r-xr-xr--. 1 user group 10 Feb 26 2017 file1
./lookHere/directory1:
-r--r--r--. 1 user group 10 Feb 26 2017 file2
D r-- r-- r-- 5
./lookHere:
drwxrwxrwx. 1 user group 10 Feb 26 2017 directory1
-r-xr-xr--. 1 user group 10 Feb 26 2017 file1
./lookHere/directory1:
-r--r--r--. 1 user group 10 Feb 26 2017 file2
```

Sample Output

For each test case, and for each directory being searched in the order provided, your program must print out a report as follows:

- If the permissions for all files/directories in a directory are correct (or there are no files/directories to review), print a single line containing the relative path to the directory, a colon (:), a space, and the words ALL GOOD.
- If any permissions are incorrect, print a line containing the relative path to the directory, followed by a colon. Then print a separate line for each file with incorrect permissions, in the order they were originally listed, in the following format:
    - The name of the file
    - The text ": WRONG PERM: "
    - The user categories with incorrect permissions (one or more of Owner, Group, and/or Other, in that order) separated by a comma and a space.

In nested directory structures, only consider those immediate children of a directory when deciding to print "ALL GOOD" or not; a child directory with a file with bad permissions doesn't reflect against the parent directory.

```
./lookHere:
file1: WRONG PERM: Owner, Group
./lookHere/directory1: ALL GOOD
./lookHere:
directory1: WRONG PERM: Owner, Group, Other
./lookHere/directory1: ALL GOOD
```