Instructions

In various cases, it is useful to determine if the current directory or some parent of it contains a particular file. For example, a repository using the Git version control system has a `.git` directory at its root, and sometimes we want to know if we are in a Git repository (and, if so, where the root it is).

Similarly, a project using the Gradle build tool usually has a `gradlew` shell script at its root, and to build we need to find and run that shell script.

For this drill, write a shell script called `find-up.sh` that takes file names to look for as its arguments, and does the following:

1. If no arguments are provided (`"$1"` is empty), exit with an error message on stderr and an exit code of 2.
2. Print the relative path (using `./..`) from the current directory for the first occurrence of any file on the command line and exit with code 0.
3. If no parent of the current directory contains any of the files, then exit with code 1 and produce no output.

So, if the user runs `find-up.sh .git` and the current directory contains a file called `.git`, it should print `./.git`. If the parent directory contains `.git`, it should print `../.git`, and `../../.git` for the grandparent directory, and so on.

If multiple files are passed, then the first directory to contain any file is selected; for `./find-up.sh .git .hg`, if the parent directory contains `.hg` and the grandparent `.git`, it should print `../.hg`.

Use the `-e` operator of `test` or `[` to test if a file exists.

To see if you have found the root directory, use `-ef`; if you have built up a path in the variable `dir`, `test "$dir" -ef /` will test if "$dir" refers to the root directory. If it does, search that directory but no farther.

The attached tests will help you ensure your script's correctness.