[CS 11 25.1] Mock HOPE 2e – Felipe and Files

Cheatsheet is available here: https://oj.dcs.upd.edu.ph/cs11cheatsheet/

Points: 160 (partial)
Time limit: 4.0s
Memory limit: 1G

Problem type
Allowed languages
py3

Problem Statement

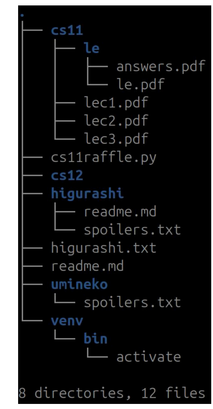
Felipe needs to enumerate the files in his thumb drive!

He has a `dict` representing the whole structure of the thumb drive. Each key of this `dict` is a `str` representing the name of either a file or a folder. The corresponding value for key `k` is as follows:

- If `k` is a file, then the value is `None`
- If `k` is a folder, then the value is another `dict` detailing the contents of that folder, in the same format recursively.

Thus, the `dict` can be arbitrarily nested, representing nested folders.

For example, suppose the thumb drive has the following structure (as outputted by the bash command `tree`, which you can try running yourself):



The ones in blue are folders, and the remaining ones are files. Note that the `cs12` folder is empty.

This folder can be represented by the following `dict`:

```
{
    "cs11": {
        "le": {
            "answers.pdf": None,
            "le.pdf": None,
        },
        "lec1.pdf": None,
        "lec2.pdf": None,
        "lec3.pdf": None,
    },
    "cs11raffle.py": None,
    "cs12": {},
    "higurashi": {
        "readme.md": None,
        "spoilers.txt": None,
    },
    "higurashi.txt": None,
    "readme.md": None,
    "umineko": {
        "spoilers.txt": None,
    },
    "venv": {
        "bin": {
            "activate": None,
        }
    }
}
```

Felipe needs to enumerate the complete paths of all the files in the folder. This is most easily done with the bash command `find -type f` (which you can try yourself). However, Felipe needs to enumerate them in a particular order:

- For each folder, the bare files appear before the files inside subfolders.
- For each folder, the bare files are sorted alphabetically.
- For each folder, the subfolder names are sorted alphabetically.

Thus, for the example above, the expected order of the files is:

```
cs11raffle.py
higurashi.txt
readme.md
cs11/lec1.pdf
cs11/lec2.pdf
cs11/lec3.pdf
cs11/le/answers.pdf
cs11/le/le.pdf
higurashi/readme.md
higurashi/spoilers.txt
umineko/spoilers.txt
venv/bin/activate
```

Task Details

Your task is to implement a function called `enumerate_files_in_order`. It has a single argument, a `dict` in the format described in the problem statement.

It must return a `list` of `str`s denoting the complete paths in the specified order. Use forward slash (`/`) as the separator character.

Restrictions

Note that some names are banned.

For this problem:

- Loops and lists are allowed.
- Sets and dictionaries are allowed.
- Generators and comprehensions are allowed.
- Recursion is allowed.
- Up to 6 function definitions are allowed.
- The source code limit is 1500.

Example Calls

Example 1 Function Call

```
enumerate_files_in_order({
    "cs11": {
        "le": {
            "answers.pdf": None,
            "le.pdf": None,
        },
        "lec1.pdf": None,
        "lec2.pdf": None,
        "lec3.pdf": None,
    },
    "cs11raffle.py": None,
    "cs12": {},
    "higurashi": {
        "readme.md": None,
        "spoilers.txt": None,
    },
    "higurashi.txt": None,
    "readme.md": None,
    "umineko": {
        "spoilers.txt": None,
    },
    "venv": {
        "bin": {
            "activate": None,
        },
    },
})
```

Example 1 Return Value

```
[
    "cs11raffle.py",
    "higurashi.txt",
    "readme.md",
    "cs11/lec1.pdf",
    "cs11/lec2.pdf",
    "cs11/lec3.pdf",
    "cs11/le/answers.pdf",
    "cs11/le/le.pdf",
    "higurashi/readme.md",
    "higurashi/spoilers.txt",
    "umineko/spoilers.txt",
    "venv/bin/activate",
]
```

Constraints

- The function `enumerate_files_in_order` will be called at most 20 times.
- The total number of files and folders across all calls will be $\leq$ 100.
- Each file and folder name consists of lowercase letters and dots and has length between 1 and 16.

Scoring

Note: New tests may be added and all submissions may be rejudged at a later time. (All future tests will satisfy the constraints.)

- You get 160 🔴 points if you solve all test cases.

Clarifications

No clarifications have been made at this time.

Report an issue