[CS 11 25.1] Lab 7f – Lab Draft

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

Submit solution [CS 11 25.1] Lab Exercise 7

My submissions

Points: 250 (partial)
Time limit: 6.0s
Memory limit: 2G

Problem type

Allowed languages
py3

Problem Statement

The Department of Computer Science consists of eight labs (as of this academic year), and students who are planning to take CS 198 and CS 199 have to rank their lab preferences in preparation for the *lab draft*.

For this problem, we will consider the following simplifications:

- Each student has exactly one preferred lab.
- Each lab must take in **exactly** x students.

As the department wants to maximize its research output, you are tasked to choose the number x, and then assign some students to a lab, such that x is maximized. Note that each student, if assigned to a lab, **must** be assigned to their preferred lab.

Task Details

For this problem, you may import the `DCSLab` enum from `oj`, which enumerates the eight DCS labs: `CSL`, `NDSL`, `CVMIL`, `S3L`, `SCL`, `SMSL`, `WSL`, and `ACL`.

Your task is to implement a function named `assign_students`, which should have the following *signature*:

```
def assign_students(preferences):
```

Copy

It has one argument `preferences`. This is a dictionary (`dict`) mapping strings (`str`s) to values of the `DCSLab` enum, denoting the names of the students and their lab preferences.

The function must return a *new* `dict` mapping each name to the lab they are assigned to, or `None` if they are not assigned to any lab. (Do **not** modify the `dict` argument!)

If multiple answers exist, any will be accepted.

Restrictions

- The following symbols are allowed: `iter`, `next`, `map`, `filter`
- The following imports are allowed:
  - `count`, `islice`, `chain`, `takewhile`, `starmap` and `zip_longest` from `itertools`.
  - `cache`, `lru_cache`, `total_ordering`, `partial`, `reduce` and `wraps` from `functools`.
  - `randint`, `randrange and choice` from `random`.
  - `Fraction` from `fractions`.
  - `dataclass` from `dataclasses`.
  - `contextmanager` from `contextlib`.
  - `Enum`, `auto` from `enum`.
  - `DCSLab` from `oj`.
- Anonymous functions are allowed.
- Inner functions are allowed.
- Classes, dataclasses, and enums are allowed.
- Recursion is allowed.
- Loops are allowed.
- Generators and comprehensions are allowed.
- Your source code must have at most 800 bytes.

Examples

Example 1 Function Call

```
assign_students({
    'Bowser': DCSLab.CSL,
    'Climbers': DCSLab.CSL,
    'DrMario': DCSLab.NDSL,
    'Falco': DCSLab.CSL,
    'Falcon': DCSLab.NDSL,
    'Fox': DCSLab.CSL,
    'Ganondorf': DCSLab.NDSL,
    'Jigglypuff': DCSLab.CSL,
    'Kirby': DCSLab.CSL,
    'Kong': DCSLab.NDSL,
    'Link': DCSLab.CSL,
    'Luigi': DCSLab.NDSL,
    'Mario': DCSLab.NDSL,
    'Marth': DCSLab.CSL,
    'Mewtwo': DCSLab.ACL,
    'MrGame': DCSLab.CSL,
    'Ness': DCSLab.WSL,
    'Peach': DCSLab.CSL,
    'Pichu': DCSLab.SMSL,
    'Pikachu': DCSLab.SCL,
    'Roy': DCSLab.SCL,
    'Samus': DCSLab.CSL,
    'Yoshi': DCSLab.S3L,
    'YoungLink': DCSLab.CSL,
    'Zelda': DCSLab.CVMIL,
})
```

Copy

Example 1 Return Value

```
{
    'Bowser': None,
    'Climbers': DCSLab.CSL,
    'DrMario': DCSLab.NDSL,
    'Falco': None,
    'Falcon': None,
    'Fox': None,
    'Ganondorf': None,
    'Jigglypuff': None,
    'Kirby': None,
    'Kong': None,
    'Link': None,
    'Luigi': None,
    'Mario': None,
    'Marth': None,
    'Mewtwo': DCSLab.ACL,
    'MrGame': None,
    'Ness': DCSLab.WSL,
    'Peach': None,
    'Pichu': DCSLab.SMSL,
    'Pikachu': None,
    'Roy': DCSLab.SCL,
    'Samus': None,
    'Yoshi': DCSLab.S3L,
    'YoungLink': None,
    'Zelda': DCSLab.CVMIL,
}
```

Copy

Example 2 Function Call

```
assign_students({
    'Richard': DCSLab.CSL,
})
```

Copy

Example 2 Return Value

```
{
    'Richard': None,
}
```

Copy

Constraints

Let s be the number of students in one call to `assign_students`.

- The function `assign_students` will be called at most $70{,}000$ times.
- $0 \leq s \leq 350{,}000$
- The sum of s across all calls to `assign_students` will be $\leq 350{,}000$.
- Each name is a string of at most 10 English letters.

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 60 ❤️ points if you solve all test cases where:
  - $s \leq 50$
  - The sum of s across all calls to `assign_students` will be ≤ 800.
- You get 30 ❤️ points if you solve all test cases where:
  - $s \leq 400$
  - The sum of s across all calls to `assign_students` will be ≤ 800.
- You get 60 🔴 points if you solve all test cases where:
  - $s \leq 6{,}000$
  - The sum of s across all calls to `assign_students` will be $\leq 12{,}000$.
- You get 100 🔴 points if you solve all test cases.

Clarifications

Report an issue

No clarifications have been made at this time.