[CS 11 25.1] Lab 4f – Year Standing

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

Submit solution [CS 11 25.1] Lab Exercise 4

My submissions

Points: 190 (partial)
Time limit: 4.0s
Memory limit: 2G

Problem type

Allowed languages
py3

Problem Statement

At the Department of Classification and Sorting, students are grouped into several *standings*, depending on their student number.

Suppose it is currently year y.

- A student whose student number starts with y is called a **freshman**.
- A student whose student number starts with $y - 1$ is called a **sophomore**.
- A student whose student number starts with $y - 2$ is called a **junior**.
- A student whose student number starts with $y - 3$ is called a **senior**.
- Any other student is called an **immortal**.

Given a list of students and their student numbers, classify them by standing.

Task Details

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

```
def classify_by_standing(y, students):
```

Copy

The above says that it has two arguments y and `students`.

- y is an integer (`int`) denoting the current year.
- `students` is a `tuple` of pairs (tuples of length 2), where each pair contains two strings (`str`s) denoting a student's name and student number.

The function must return a dictionary (`dict`) with five keys `freshman`, `sophomore`, `junior`, `senior`, and `immortal`. The value corresponding to each standing should be a `set` of the names of students with that standing.

Restrictions

- The following symbols can now be used: `list`, `set`, `dict`, `enumerate`, `append`, `pop`, `extend`, `remove`, `sort`, `sorted`, `insert`, `clear`, `reverse`, `reversed`.
- Loops are allowed.
- Recursion is *disallowed*.
- Comprehensions are *disallowed*.
- Your source code must have at most $1{,}000$ bytes.

Examples

Example 1 Function Call

```
classify_by_standing(2025, (
    ("Alpha", "2022-99999"),
    ("Beta", "2021-11111"),
    ("Charlie", "2025-12345"),
    ("Delta", "2024-31415"),
    ("Echo", "2023-27182"),
    ("Foxtrot", "2000-00000"),
))
```

Copy

Example 1 Return Value

```
{
    "freshman": {"Charlie"},
    "sophomore": {"Delta"},
    "junior": {"Echo"},
    "senior": {"Alpha"},
    "immortal": {"Beta", "Foxtrot"},
}
```

Copy

Example 2 Function Call

```
classify_by_standing(2999, ())
```

Copy

Example 2 Return Value

```
{
    "freshman": set(),
    "sophomore": set(),
    "junior": set(),
    "senior": set(),
    "immortal": set(),
}
```

Copy

Constraints

- The function `classify_by_standing` will be called at most 100 times.
- $2{,}000 \le y \le 3{,}000$
- Each student name is a string of (uppercase and lowercase) English letters with length at most 10.
- Each student number is a string of the form `WXYZ-ABCDE`, where each letter is some digit from `0` to `9`.
- The length of `students` is at most 100.

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 90 ❤️ points if you solve all test cases where:
  - There are no `immortal` students.
- You get 100 🔴 points if you solve all test cases.

Clarifications

Report an issue

No clarifications have been made at this time.