[CS 11 25.1] Lab 7e – Pista ng Pasta

Sub[CS 11 25.1] Lab Exercise 7

Points: 290 (partial)
Time limit: 4.0s
Memory limit: 1G

Problem type

Allowed languages
py3

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

Problem Statement

It is currently Pista ng Pasta, and there is a buffet of pasta!

The buffet consists of n stations in a row, numbered 0 to $n-1$ from left to right, each serving one of the following three types of pasta:

- **Spaghetti alla puttanesca**, a pasta dish invented in Naples in the mid-20th century and made typically with tomatoes, olives, capers, anchovies, garlic, peperoncino, extra virgin olive oil, and salt.
- **Anelletti al forno** or timballo di anelletti, a type of baked pasta called pasta al forno typical of Palermo and its province but also widespread in the rest of Sicily.
- **Fagottini**, a filled pasta usually filled with vegetables, typically steamed carrots and green beans, ricotta, onion and olive oil. Fagottini are made by cutting sheets of pasta dough into squares, placing the filling on the square, and folding the corners to meet in a point.

Each person can only get one serving of pasta from each station. Furthermore, each person likes to eat only one of these pasta types, so they will only get servings from stations serving the pasta type they like.

Also, each station's pasta is cooked by a different chef, which means the quality varies a lot. When a person eats a serving from station i, their happiness increases by h_i.

Suppose a person starts at station i and exits after checking out station j, with $0 \le i \le j < n$. Given the type of pasta they like, what is the total happiness they gain after eating a single serving from each station from i to j serving the pasta they like?

You must process up to q queries of the previous kind.

Task Details

In this problem, you will be provided a module named `pasta` whose contents are exactly as follows:

```
from dataclasses import dataclass
from enum import Enum, auto

class Pasta(Enum):
    PUTTANESCA = auto()
    ANELLETTI = auto()
    FAGOTTINI = auto()

@dataclass
class Station:
    pasta_type: Pasta
    h: int
```

Import it using

```
from pasta import Pasta, Station
```

Your task is to implement a function called `process_stations`. It takes a single argument, a `tuple` or `list` of n `Station`s representing the pasta stations from left to right.

- The `pasta_type` represents the type of pasta served at this station.
- The `.h` attribute represents the happiness increase for each serving eaten from this station.

The function must return a function. Calling this returned function represents a query. It takes in three arguments:

- first, the `int` i.
- second, the `int` j.
- third, a `Pasta` denoting the type of pasta that the person likes.

It must return an `int` denoting the total happiness the person gains after eating a single serving from each station from i to j serving the pasta they like.

Restrictions

Note that some names are banned. Here are a few of them: `input`, `type`. This is not an exhaustive list. (If you accidentally use a variable name that turns out to be banned, please rename it.)

The following names are allowed: `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`.

(Read the docs to learn what they do!)

Anonymous functions are allowed.

Inner functions are allowed.

Classes, dataclasses and enums are allowed.

For this problem in particular:

- The following imports are allowed: `Pasta`, `Station` from `pasta`.
- The source code limit is 3000.

Example Testing

Note: This assumes that your submission has filename `hope3d1.py`. Write this testing code in a separate file, say `test_hope3d1.py`, and run it to test your code.

Example 1 Testing

```
from pasta import Pasta, Station

from hope3d1 import process_stations

def verify(condition: bool, message: str = "verification
failed"):
    assert condition, message

get_happiness = process_stations((
    Station(pasta_type=Pasta.PUTTANESCA, h=32),
    Station(pasta_type=Pasta.PUTTANESCA, h=11),
    Station(pasta_type=Pasta.ANELLETTI, h=2),
    Station(pasta_type=Pasta.ANELLETTI, h=3),
    Station(pasta_type=Pasta.FAGOTTINI, h=5),
    Station(pasta_type=Pasta.PUTTANESCA, h=199),
    Station(pasta_type=Pasta.FAGOTTINI, h=5),
))

verify(get_happiness(1, 3, Pasta.PUTTANESCA) == 11)
verify(get_happiness(1, 3, Pasta.ANELLETTI) == 5)
verify(get_happiness(1, 3, Pasta.FAGOTTINI) == 0)
verify(get_happiness(0, 6, Pasta.ANELLETTI) == 5)
verify(get_happiness(0, 6, Pasta.PUTTANESCA) == 242)
```

Testing

To test your program locally, you should create a file called `pasta.py` and save the code above to it. Note that this `pasta.py` is **not** to be submitted! The judge has its own version of `pasta.py`. The `pasta.py` you create is only for your own testing.

Constraints

- The function `process_stations` will be called at most $60{,}000$ times.
- The sum of ns across all calls will be $\le 200{,}000$.
- The sum of qs across all calls will be $\le 200{,}000$.
- $1 \le n \le 200{,}000$
- $0 \le i \le j < n$
- $0 \le h_i \le 10^{20}$

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 70 ❤️ points if you solve all test cases where:
  - $n \le 50$
  - The sum of the ns across all calls will be 500.
  - The sum of the qs across all calls will be 500.
- You get 100 🔴 points if you solve all test cases where:
  - $n \le 4{,}000$
  - The sum of the ns across all calls will be $8{,}000$.
  - The sum of the qs across all calls will be $8{,}000$.
- You get 120 🔴 points if you solve all test cases.

Clarifications

Report an issue

No clarifications have been made at this time.