[CS 11] Prac 7c – Increasing Counts

Problem Statement

Given a sequence of integers give the same sequence but with the kth element appearing k times.

Task Details

Your task is to implement a function called `increasing_appearances`. This function has a single parameter: an iterable of `int`s.

The function must return a *generator* that generates `int`s, as described in the problem statement.

Note that your generator must be **as lazy as possible**. It should yield each resulting next element as soon as it has enough information, and it should produce these results while advancing the input generators for as little as possible.

Restrictions

(See 7a for more restrictions)

For this problem:

- Loops and lists are allowed.
- Up to 8 function definitions are allowed.
- Recursion is **disallowed**. (The recursion limit has been greatly reduced.)
- Sets and dictionaries are allowed.
- Generators and comprehensions are allowed.
- The source code limit is 300.

Example Calls

Example 1 Function Call

```
[*increasing_appearances((3, 1, 4))]
```
Copy

Example 1 Return Value

```
[3, 1, 1, 4, 4, 4]
```
Copy

Example 2 Function Call

```
print(*increasing_appearances(iter((3, 1, 4))))
```
Copy

Example 2 Output

```
3 1 1 4 4 4
```
Copy

Constraints

When your program is run:

- The function `increasing_appearances` will be called at most 200 times.
- At most 500 elements will be consumed from the returned generator.
- Each element of the input sequence is a positive integer at most 10^{10}.

Scoring

- You get 120 ♥ points if you solve all test cases.

Clarifications

Report an issue

No clarifications have been made at this time.

Submit solution
[CS 11] Practice 7 ♥
My submissions ♥

Points: 120 (partial)
Time limit: 6.0s
Memory limit: 1G

Author: kvatienza (Kevin Atienza)

Problem type

Allowed languages
~~NONE~~, py3