[CS 11] Prac 7p – Continued Fraction I

Problem Statement

Given an integer k, consider the following infinite sequence of rational numbers:

$$k, k+\frac{1}{k+1}, k+\frac{1}{(k+1)+\frac{1}{k+2}}, k+\frac{1}{(k+1)+\frac{1}{(k+2)+\frac{1}{k+3}}}, \ldots$$

For example, for $k = 1$, we get

$$1, 1+\frac{1}{2}, 1+\frac{1}{2+\frac{1}{3}}, 1+\frac{1}{2+\frac{1}{3+\frac{1}{4}}}, \ldots$$

which simplify to

$$\frac{1}{1}, \frac{3}{2}, \frac{10}{7}, \frac{43}{30}, \frac{225}{157}, \frac{1393}{972}, \ldots$$

For $k = 2$, we get

$$2, 2+\frac{1}{3}, 2+\frac{1}{3+\frac{1}{4}}, 2+\frac{1}{3+\frac{1}{4+\frac{1}{5}}}, \ldots$$

which simplify to

$$\frac{2}{1}, \frac{7}{3}, \frac{30}{13}, \frac{157}{68}, \frac{972}{421}, \frac{6961}{3015}, \ldots$$

Your task is to generate the infinite sequence of *numerators* of such a sequence.

Hint: The numerator of each term can be obtained from the previous two numerators!

Task Details

Your task is to implement a function called `cf_numerators`. This function has a single parameter k, an `int`.

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 400.

Example Calls

Example 1 Function Call

```
[*take(6, cf_numerators(1))]
```

Copy

Example 1 Return Value

```
[1, 3, 10, 43, 225, 1393]
```

Copy

Example 2 Function Call

```
[*take(6, cf_numerators(2))]
```

Copy

Example 2 Return Value

```
[2, 7, 30, 157, 972, 6961]
```

Copy

Constraints

When your program is run:

- The function `cf_numerators` will be called at most 6 times.
- At most 100 elements will be consumed from the returned generator.
- $1 \leq k \leq 100$.

Scoring

- You get 80 ❤ points if you solve all test cases.

Submit solution

[CS 11] Practice 7

Points: 80 (partial)

Time limit: 6.0s

Memory limit: 1G

Author: kvatienza (Kevin Atienza)

Problem type

Allowed languages ~~NONE~~, py3

Clarifications

Report an issue

No clarifications have been made at this time.