Course > 10: Accumulators > Context Preserving Accumulators > Questions 3-4

Previous Next

Questions 3-4

Bookmark this page

Question 3

1/1 point (graded)

You start the process of adding an accumulator. Here is what you have so far:

```
(define (number-list lon0)
  ;; acc: Natural; 1-based position of (first lon) in lon
  (local [(define (number-list lon acc)
            (cond [(empty? lon)(...)]
                  [else
                   (... (first lon)
                        (number-list (rest lon)))]))]
    (number-list lon0)))
```

Which of the following are valid examples for the accumulator?

- ☐ `;; (number-list (list "first" "second" "third") "1: ")`
- ☑ `;; (number-list (list "first" "second" "third") 1)`
- ☑ `;; (number-list (list "second" "third") 2)`
- ☐ `;; (number-list (list "1: first" "second" "third") "2: ")`
- ☐ `;; (number-list empty 0)`
- ☑ `;; (number-list (list "third") 3)`

✓

Explanation

The accumuluator should be a Natural, and it should be the position of the first element of the current list in the original list.

Submit

Show Answer

Answers are displayed within the problem

Question 4

1/1 point (graded)

In adding the accumulator paramater, we have missed a spot where it should be added. Click to move the green dot to the location of the missing accumulator paramater.

```
(define (number-list lon0)
  ;; acc: Natural; 1-based position of (first lon) in lon
  (local [(define (number-list lon acc)
            (cond [(empty? lon)(... acc)]
                  [else
                   (...●(first lon)
                        (number-list (rest lon)
                                     (... acc)))]))]
    (number-list lon0 ...)))
```

✓

Explanation

We need to add `acc` at all of the `...`s, as well as adding an extra paramater to every call of `number-list`.

Submit

Show Answer

Answers are displayed within the problem

Previous Next