Questions 5-7

Bookmark this page

Question 5

1/1 point (graded)

We are close to completing our function:

```
;; (listof String) -> (listof String)
;; append each string's position in the list to the front of the string to number the list
(check-expect (number-list empty) empty)
(check-expect (number-list (list "first" "second" "third"))
              (list "1: first" "2: second" "3: third"))
;(define (number-list los) los)   ;stub

(define (number-list lon0)
  ;; acc: Natural; 1-based position of (first lon) in lon
  ;; (number-list (list "first" "second" "third") 1)
  ;; (number-list (list         "second" "third") 2)
  ;; (number-list (list                  "third") 3)
  (local [(define (number-list lon acc)
            (cond [(empty? lon)(... acc)]
                  [else
                   (... acc
                        (first lon)
                        (number-list (rest lon)
                                     (... acc)))]))]
    (number-list lon0 ...)))
```

What should go in the base case, after `(empty? lon)`?

- `"0:"`
- `empty` ✓
- `0`
- `false`

Explanation
As shown in the `check-expect` above, the empty list will just produce empty

Submit Show Answer

Answers are displayed within the problem

Question 6

1/1 point (graded)

How should we initialize the accumulator in the trampoline call?

- `(number-list lon0 0)`
- `(number-list lon0 1)` ✓
- `(number-list lon0 "0: ")`
- `(number-list lon0 "1: ")`

Explanation
The initial value of `acc` must be a natural number, and it should start at 1 because, we are using 1-based indexing to number the list.

Submit Show Answer

Answers are displayed within the problem

Question 7

1/1 point (graded)

How should we update the accumulator in the recursive call?

- `(number-list (rest lon) acc)`
- `(number-list (rest lon) (add1 acc))` ✔
- `(number-list (rest lon) (sub1 acc))`
- `(number-list (rest lon) 0)`

Explanation
Adding one to the accumulator will give us the position of the new first element of the list in `rest lon`.

Submit

Show Answer

Answers are displayed within the problem

Previous Next