## Problem 1-3

The following is a revised data definition for `Person` and `ListOfPerson`:

```
(define-struct person (name age children))
;; Person is (make-person String Natural ListOfPerson)
;; interp. a person with first name, age and a list of their children
#;
(define (fn-for-person p)
  (...(person-name p)     ;String
      (person-age p)      ;Natural
      (fn-for-lop (person-children p))))

;; ListOfPerson is one of:
;; - empty
;; (cons Person ListOfPerson)
;; interp. a list of persons
#;
(define (fn-for-lop lop)
  (cond [(empty? lop)(...)
        [else
         (... (fn-for-person (first lop))
              (fn-for-lop (rest lop)))]))
```

Take a moment to do some reference analysis and write down some examples.

### Questions 1-3

3/3 points (graded)

How many self-references are there?

1 ✓ **Answer:** 1

**Explanation**

`ListOfPerson` has a self-reference in it.

How many references are there?

0 ✓ **Answer:** 0

**Explanation**

All remaining references are part of a mutual reference cycle, so there are 0.

How many mutual-references are there?

2 ✓ **Answer:** 2

**Explanation**

`ListOfPerson` has a reference to `Person` and vice versa.

Submit

Answers are displayed within the problem