Questions 5-7

Bookmark this page

Question 5

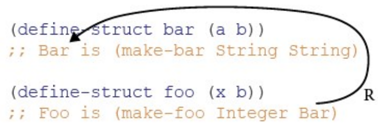
1/1 point (graded)

Just as we can have self-reference without reference (`ListOfString`), we can also have reference without self-reference. Consider these two partial data definitions:

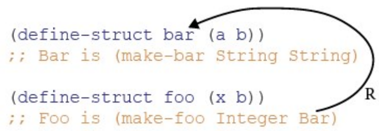
```
(define-struct bar (a b))
;; Bar is (make-bar String String)

(define-struct foo (x b))
;; Foo is (make-foo Integer Bar)
```

What are the correct arrows for this figure?

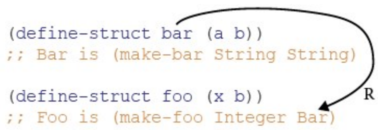
- (selected)



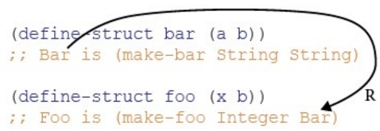
-



-



-



Explanation

The reference should go from `Bar` in `(make-foo Integer Bar)` to the `Bar` before the word is.

Submit

Show Answer

Answers are displayed within the problem

Question 6

1/1 point (graded)

What is the correct template for `foo`?

-

```
(define (fn-for-foo f)
        (... (foo-x f)
             (foo-b f)))
```

-

```
(define (fn-for-foo f)
        (... (fn-for-bar (foo-x f))
             (foo-b f)))
```

-

```
(define (fn-for-foo f)
        (... (fn-for-bar (foo-x f))
             (fn-for-bar (foo-b f))))
```

- (selected)

```
(define (fn-for-foo f)
        (... (foo-x f)
             (fn-for-bar (foo-b f))))
```

Explanation
`(foo-x f)` is of type `Bar`, so it is wrapped in `fn-for-bar`

Submit

Show Answer

Answers are displayed within the problem

Question 7

1/1 point (graded)

Drag the template rules to create the correct template rules for `foo`. Only use as many template rules as are necessary.

```
;; Template Rules Used
;; - compound
;; - reference
;; - atomic
```

Answer:

Explanation
`Foo` is compound. The first field is a simple atomic type, Integer, and the second field is a reference to a non-primitive type.

Submit

Show Answer

Answers are displayed within the problem

Previous Next