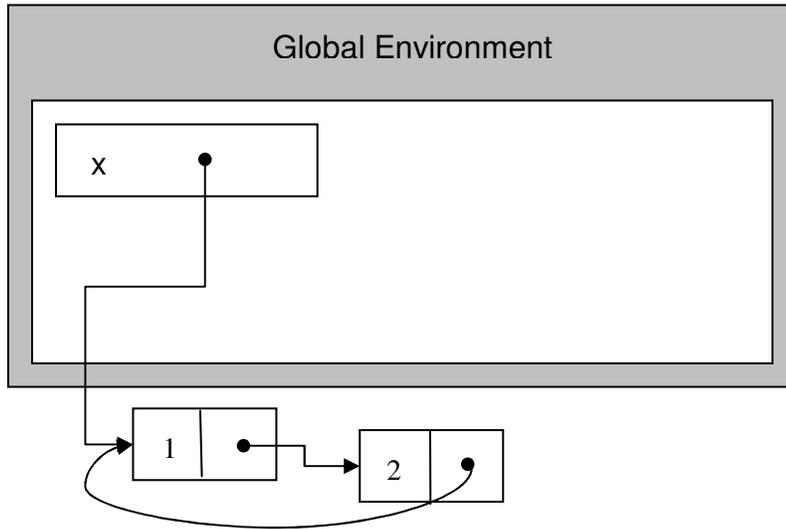




# Environments

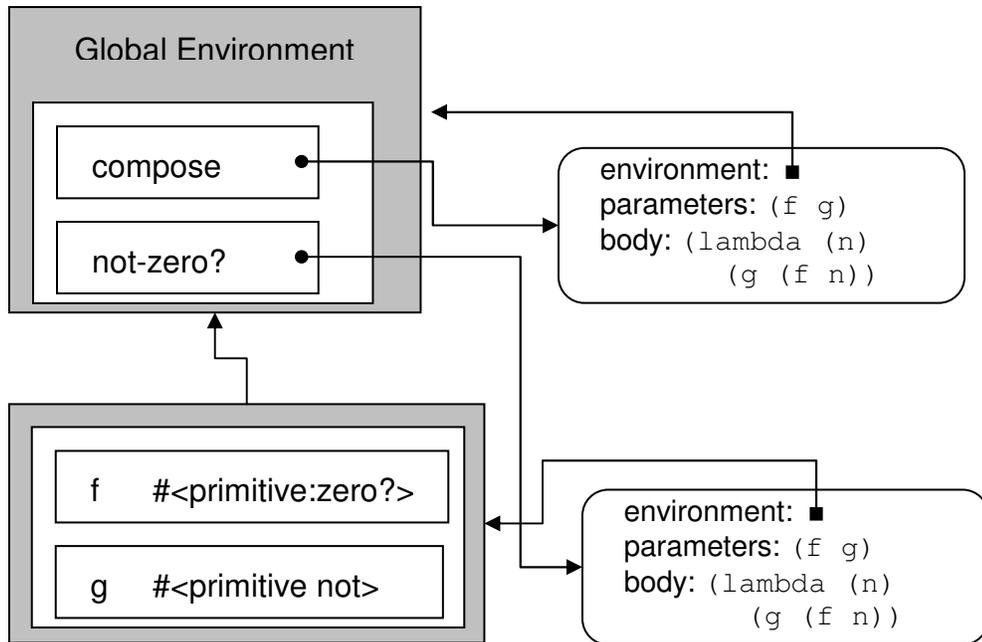
1. Consider the environment shown below (assume all the usual primitives are defined in the global environment, but not shown):



Provide one additional expression that follows the given definition, such that the environment shown above results when the sequence of expressions is evaluated.

```
(define x (cons 1 (cons 2 3)))
```

2. Consider the environment shown below (as in question 1, assume all the usual primitives are defined in the global environment, but not shown; the notation #<primitive:zero?> denotes the primitive procedure zero?):



Provide a sequence of Scheme expressions such that evaluating the sequence of expressions produces the environment shown above.

## Computability

3. Is the *Contains-Cross-Site-Scripting-Vulnerability Problem* described below computable or uncomputable? Your answer should include a convincing argument why it is correct.

**Input:**  $P$ , a specification (all the code and html files) for a dynamic web application.

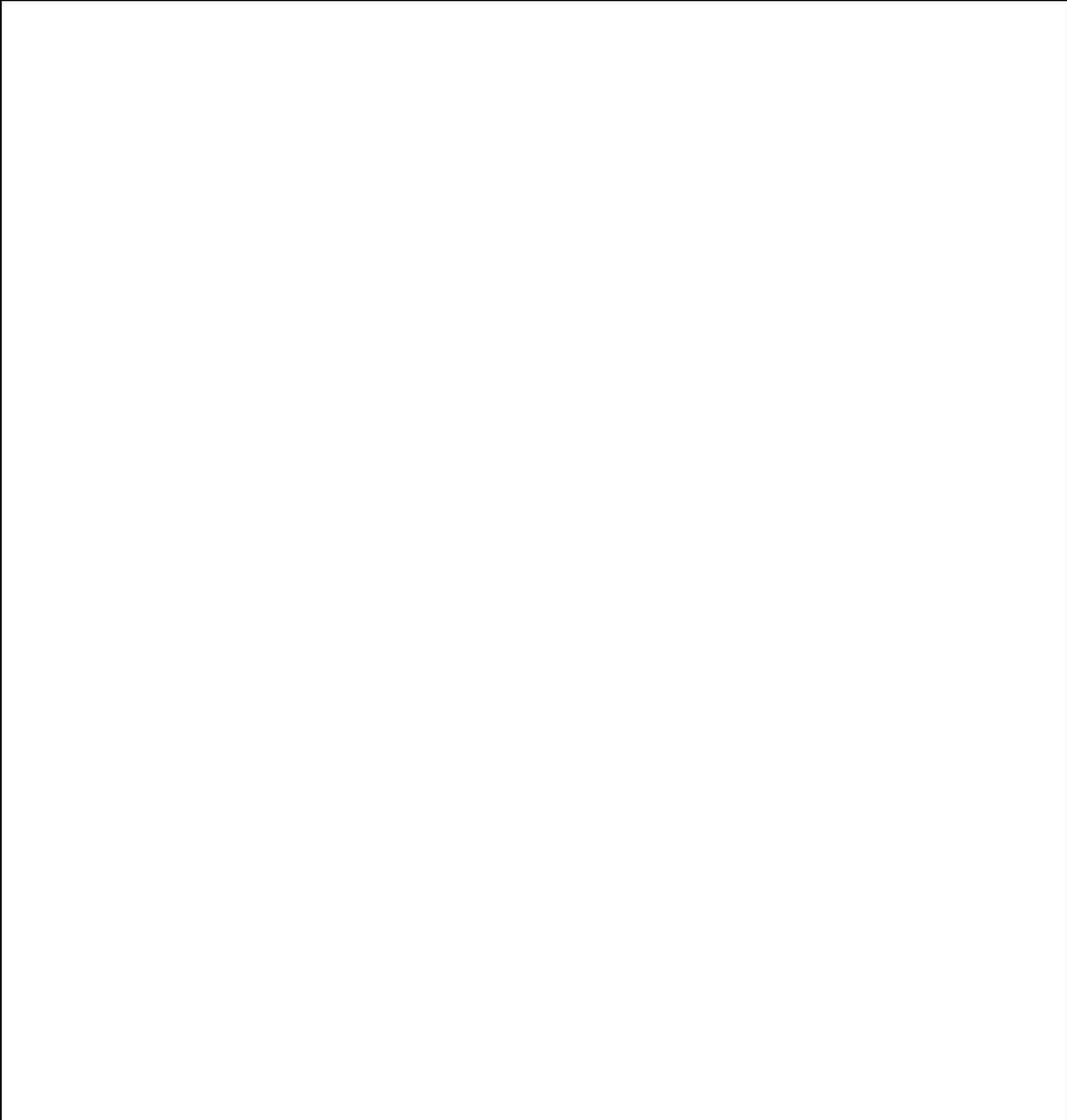
**Output:** If  $P$  contains a cross-site-scripting vulnerability, output **True**.  
Otherwise, output **False**.

As demonstrated in class, a cross-site-scripting vulnerability is an opportunity an attacker can exploit to get their own script running on a web page generated by the web application.

4. Is the *Remove-Cross-Site-Scripting-Vulnerabilities Problem* described below computable or uncomputable? Your answer should include a convincing argument why it is correct.

**Input:**  $P$ , a specification (all the code and html files) for a dynamic web application.

**Output:**  $P'$ , a specification for a dynamic web application. On inputs that are not cross-site-scripting attacks,  $P'$  behaves identically to  $P$ . On inputs that are cross-site-scripting attacks,  $P'$  ignores the attack input and displays a warning page.



## Interpreters and Asymptotic Running Time

For the next three questions, you are given a procedure definition. Your answer should describe its asymptotic running time when evaluated using (a) Charme, and (b) MemoCharme (the language defined at the end of PS7), and (c) LazyCharme. You may assume the Python dictionary type provides lookups with running time in  $O(1)$ . Your answers should include a clear supporting argument, and define all variables you use in your answer.

5.

```
(define mysterious
  (lambda (a)
    (cond
      ((> a 0) (mysterious (- a 1)))
      ((zero? a) 0))))
```

(a) Running time in Charme:

(b) Running time in MemoCharme:

(c) Running time in LazyCharme:

6.

```
(define duplicitous
  (lambda (a)
    (cond
      ((> a 0) (+ (duplicitous (- a 1))
                  (duplicitous (- a 1))))
      ((zero? a) 1))))
```

(a) Running time in Charme:

(b) Running time in MemoCharme:

(c) Running time in LazyCharme:

7.

```
(define temeritous
  (lambda (a b)
    (cond
      ((> a 0) (temeritous (- a 1)
                           (temeritous (+ a 1) b)))
      ((zero? a) 2))))
```

(a) Running time in Charme:

(b) Running time in MemoCharme:

(c) Running time in LazyCharme:

## Static Type Checking

8. (as promised, Exercise 14.1) Define the `typeConditional(expr, env)` procedure that checks the type of a conditional expression. It should check that all of the predicate expressions evaluate to a Boolean value. In order for a conditional expression to be type correct, the consequent expressions of each clause produce values of the same type. The type of a conditional expression is the type of all of the consequent expressions. (You may assume the `StaticCharme` interpreter described in Chapter 14.)

**9.** (based on Exercise 14.2) A stronger type checker would require that at least one of the conditional predicates must evaluate to a true value. Otherwise, the conditional expression does not have the required type (instead, it produces a run-time error). Either define a `typeConditional` procedure that implements this stronger typing rule, or explain convincingly why it is impossible to do so.

These questions are optional and worth no credit, but we appreciate your answers. (Feel free to use as much space as you want to answer these.)

**10.** Do you feel your performance on this exam will fairly reflect your understanding of the course material so far? If not, explain why.

**11.** What is the most interesting thing you have learned in this class?

**12.** What is the most disappointing thing about this class?

**End of Exam**