### Name: \_\_\_\_\_

#### Directions

**Please be honorable.** As long as everyone follows the honor code, take home exams benefit everyone. You are on your honor to follow the letter and spirit of these directions. You do not need to scribble a pledge on your exam to convince us you are honorable, but if you cheat please let us know so it does not unfairly impact other students. Please don't cheat.

**Work alone.** You may not discuss these problems or anything related to the material covered by this exam with anyone except for the course staff between receiving this exam and class Monday.

**Open resources.** You may use any books you want, lecture notes, slides, your notes, and problem sets. You may also use DrScheme or Python, but it is not necessary to do this. You may also use external non-human sources including books and web sites. If you use anything other than the course books, slides, and notes, cite what you used. You may not obtain any help from other humans other than the course staff.

**Answer well.** Answer all questions 1-9 (question 0 is your name, which hopefully everyone will receive full credit for, unlike on Exam 1), and optionally answer questions 10-12.

You may either: (1) write your answers on this exam or (2) type and write your answers into the <u>Word document template</u> (*/cs150/exams/exam2/exam2.doc*). Whichever you choose, you must turn in your answers printed on paper and they must be clear enough for us to read and understand. You should not need more space than is provided to write good answers, but if you want more space you may attach clearly-marked extra sheets.

The questions are not necessarily in order of increasing difficulty, so if you get stuck on one question you should continue on to the next question. There is no time limit on this exam, but it should not take a well-prepared student more than a few hours complete. You have until Friday to finish the exam so there is no time pressure, but you are also expected to make progress on your PS9 project this week.

**Full credit depends on the clarity and elegance of your answer, not just correctness.** Your answers should be as short and simple as possible, but not simpler. Your programs will be judged for correctness, clarity and elegance, but you will not lose points for trivial errors (such as missing a closing parenthesis).

					3001	63				
0	1	2	3	4	5	6	7	8	9	Total
10	10	10	10	10	10	10	10	10	10	100

Scores

## 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:	<i>P</i> , 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.

(a) Running time in Charme:

(b) Running time in MemoCharme:

(c) Running time in LazyCharme:

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

(a) Running time in Charme:

6.

(b) Running time in MemoCharme:

(c) Running time in LazyCharme:

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

(a) Running time in Charme:

7.

(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