

INSTRUCTIONS

- You have 10 minutes to complete this quiz.
- The exam is closed book, closed notes, closed computer, closed calculator.
- Mark your answers **on the exam itself**. We will *not* grade answers written on scratch paper.
- For multiple choice questions, fill in each option or choice completely.
  - ☐ means mark **all options** that apply
  - ☐ means mark a **single choice**

Last name	
First name	
Student ID number	
CalCentral email (_@berkeley.edu)	
Discussion Section	____ _
<i>All the work on this exam is my own.</i> (please sign)	

0. **Your thoughts?** If Scheme was a character, what would it look like?

### 1. A Deep Problem

deep-squares, which takes a deep list of numbers and returns a list with each value squared, is given below.

```

1 (define (deep-squares lol)
2   (cond ((null? lol) '())
3         ((list? (car lol))
4          (cons (deep-squares (car lol))
5                (deep-squares (cdr lol)) ))
6         (else (cons (square (car lol))
7                     (deep-squares (cdr lol)) ))))

```

For which of the following inputs will deep-squares not work as intended?

- (a) (deep-squares '()) ☒ Works ☐ Broken  
 (b) (deep-squares '(1 (2 3) 4)) ☒ Works ☐ Broken  
 (c) (deep-squares '(1 (2 3) ((4)) 5)) ☐ Works ☒ Broken

Which line number contains the bug? ☐ 1 ☐ 2 ☐ 3 ☒ 4 ☐ 5 ☐ 6 ☐ 7

### 2. ... That Factors Into Your Learning

Implement the factors procedure in Scheme, which takes an integer  $n$  that is greater than 1 and returns a list of all of the factors of  $n$  from 1 to  $n - 1$  in increasing order. You may not need to use all the lines.

*Hint:* The built-in modulo procedure returns the remainder when dividing one number by the other.

```
scm> (modulo 5 3)
```

```
2
```

```
scm> (modulo 14 2)
```

```
0
```

```
(define (factors n)
```

```
  (define (factors-helper i n)
```

```
    (if (= i n)
```

```
        nil
```

```
        (if (= (modulo n i) 0)
```

```
            (cons i (factors-helper (+ i 1) n))
```

```
            (factors-helper (+ i 1) n)
```

```
        )))
```

```
  (factors-helper 1 n)
```

```
)
```

```
scm> (factors 6)
```

```
(1 2 3)
```

```
scm> (factors 7)
```

```
(1)
```

```
scm> (factors 28)
```

```
(1 2 4 7 14)
```