Hello my name is Ann Hsueh

CS 61A Midterm #3 ver1.01 April 13, 1998 Exam Version A

Your Name:

Your login: cs61a-

Discussion section number:

TAs name:

This exam is worth 40 points, or about 13% of your total course grade. The exam contains seven substantive questions, plus the following:

Question 0 (1 point):

Fill out this front page correctly and put your name and login correctly at the top of each of the following pages.

This booklet contains eight numbered pages including the cover page. Put all answers on these pages, please; don't hand in stray pieces of paper. This is an open book exam.

When writing procedures, don't put in error checks. Assume that you will be given arguments of the correct type.

Our expectation is that many of you will not complete one or two of these questions. If you find one question especially difficult, leave it for later; start with the ones you find easier.

READ AND SIGN THIS:

I certify that my answers to this exam are all my own work, and that I have not discussed the exam questions or answers with anyone prior to taking this exam.

If I am taking this exam early, I certify that I shall not discuss the exam questions or answers with anyone until after the scheduled exam time.

Question 1:

1. Write a brief scheme expression that generates the following stream (2 marks):

```
{ (1) (1 2) (1 2 3) (1 2 3 4) (1 2 3 4 5) (1 2  )  }
```

2. Given the following definition:

```
(define (stream-maker start fn)

      (define result
```

```
                    (cons-stream start (stream-map fn result)))

          result)
```

For each of the following streams, can you create the stream from one call to stream-maker? If "yes" then provide the call. If "no" then just write "no". (2 marks total)

- The stream of zeros and ones `0 1 0 1 0 1 0 1 0 1 0 1`
- The stream of words like `b ban banan bananan banananan`

1. Construct the stream of data

```
{ 0 1 10 11 100 101  }
```

that looks like the integers in binary. (2 marks)

Hint 1: You can use our old friend the word function to compute (word 1 1) to make 11. Or you can multiply 1 by 10 and add 1 to make 11. Hint 2: Think about defining a function that takes a stream of numbers (a b) and makes a new stream that looks like (a0 a1 b0 b1). Hint 3: Note that all the numbers in the stream except the first being with the digit 1.

Question 2:

1. Which of the following three sets require a stream representation? (Underline the ones that do) (1 mark)

   - the non-negative even numbers
   - the prime numbers
   - the even prime numbers

1. Eva writes a program that produces an infinite stream whose elements are randomly chosen integers in the range 0 to *n* inclusive; all of the numbers in this range appear in the stream. Louis claims to write a program that *sorts* this stream. In one English sentence, explain why Louis is lying.
2. Explain why Louis can write a program that looks as though it does this; Eva cannot tell whether it has sorted her stream or not. (1 mark)
3. Write a program that returns a stream that is indistinguishable from a sorted version of Evas stream. (1 mark)

Question 3:

The following expression have been entered in the order given, after opening a new Scheme session:

```
> (define x 100)

> (define (foo x)
```

```
(let ((x 5) (y x))

            (define (bar y)

                        (* x y))

            (lambda (x)

                        (if (= x y)

            (+ x (bar x))

            (+ y (bar x))))))
```

```
> ((foo 3) 4)
```

Answer the following questions, for one mark each:

1. What is returned from evaluating the last expression (i.e. `((foo 3) 4)`)?
2. During the execution of these expressions, how many frames are created by user-defined functions, *other than the global environment?*
3. How many procedure pairs (pairs of circles) are created as a result of executing these expressions?
4. Of these procedure pairs (pairs of circles), how many circles point to the global environment?
5. List everything *new* that is defined in the global environment as a consequence of running this sequence of commands.
6. Is there an environment frame in the environment diagram created by executing these expressions which looks like this?

   | y:3 |

   | |

   ---

7. If this frame exists, does it point to the global environment directly?
8. Is there an environment frame in the environment diagram created by executing these expressions which looks like this?

   | y:4 |

   | |

   ---

9. If this frame exists, does it point to the global environment directly?

Question 4:

In a new scheme session, you enter the following forms in the order given:

```
(define (f) (let ((a 3))
           (lambda (x) (set! a (+ a x)) a)))
(define my-f (f))
(define my-f2 (f))
(define g (let ((a 3)) (lambda (x) (set! a (+ a x)) a)))
(define my-g1 g)
(define my-g2 g)
```

Next to each of the forms below *which are entered in this order*, write the return value (0.5 marks each):

- `(my-f 5)`
- `(my-f 4)`
- `(my-f2 5)`
- `(my-f2 4)`
- `(my-g1 5)`
- `(my-g1 4)`
- `(my-g2 5)`
- `(my-g2 4)`

Question 5:

In a new scheme session, you enter the following expressions in the order given:

```
(define a (list a b c))
(define b (list 1 2 3))


(define (f x)
           (set! b (cdr b))
           (set-cdr! (cdr a) x))
```

```
(define g
           (let ((a (cddr a)))
                      (lambda (b)
                                 (set-car! b a))))


(define (h x)
           (f x)
           (g x)
           (set-cdr! x b)
           (set-cdr! (car x) b))


(h a)
```

Answer the following questions, for one mark each:

- When you evaluate `a` at the prompt, what does scheme print out?
- When you evaluate `b` at the prompt, what does scheme print out?
- Draw the box and pointer diagram for `a`:
- Draw the box and pointer diagram for `b`:
- During the execution of these expressions, how many different frames were created by user-defined functions *excluding the global frame*?
- During the execution of these expressions, how many different procedure pairs (pairs of circles) were created by user-defined functions?
- How many of these DID NOT contain a circle pointing to the global frame?

Question 6:

Into a fresh scheme session with our obj.scm loaded, you enter:

```
(define-class (staff) (method (help) "let me check")
                                 (defalut-method "see me later"))


(define-class (ta)
```

```
            (parent (staff))

            (method (help) "Did you try this yourself?")

            (method (re-grade) "Ill give you 1 more point")

            (default-method "see me during office hours"))


(define-class (prof) (parent (staff)) (default-method "send me email"))


(define rjf (instantiate prof))

(define paul (instantiate ta))
```

1. Next to each of the forms below *which are entered in this order*, write the return value (3 marks total):

    - `(ask staff help)`
    - `(ask paul help)`
    - `(ask rjf re-grade)`
    - `(ask rjf help)`

1. Since objects are procedures, in one English sentence explain why we encourage the use of ask. (i.e. explain why it is better to do `(ask paul help)` than `((paul help))`) (1 mark)

Question 7:

Into a clean Scheme session, you enter the following forms in the order given:

```
(define x 2)

(define s1 (make-serializer))

(define s2 (make-serializer))

(define f1 (s1 (lambda () (set! x (* x x)))))

(define f2 (s2 (lambda () (set! x (* x x)))))
```

- You now enter:

```
(parallel-exec f1 f2)
```

What are the possible values for `x` if this terminates? (2 marks)

- You now enter:

```
(define x 2)

(parallel-exec (s2 f1) (s1 f2))
```

What are the possible values for `x` if this terminates? (2 marks)

- If either of the above parallel executions may result in deadlock, underline it. (1 mark)