CS 61A Midterm #1 – September 28, 2005

Your name _____

Login: cs61a - _____

Discussion section number _____

TA's name _____

This exam is worth 40 points, or about 13% of you total course grade. The exam contains 6 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 6 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.

Question 1 (7 points):

What will Scheme print in response to the following expressions? If an expression produces an error message, you may just write "error"; you don't have to provide the exact text of the message. If the value of an expression is a procedure, just write "procedure"; you don't have to show the form in which Scheme prints procedures.

(* (+ 5)) _____

(se '(a) (word 'b 'c)) _____

(every (bf x) '(ab cd ef gh)) _____

(let ((k keep)) (k even? '(1 2 3))) _____

(word (first '(cat)) (butlast 'dog)) _____

((word 'but 'first) 'plop) _____

(cond ('hello 5) (#t 6) (else 7)) _____

Question 2 (8 points):

The primitive procedure member? takes a word and a sentence as arguments, and returns #t if the word is one of the words in the sentence, or #f otherwise.

We want a procedure members? that takes two sentences as arguments, returning #t if every word of the first sentence is a member of the second:

```
> (members? '(i will) '(i know she will))
#t
> (members? '(love me do) '(love you to))
#f
```

(a) Write the program by filling in the blanks below.
USE RECURSION, NOT HIGHER-ORDER FUNCTIONS.

```
(define (members? words sent)

    (cond ((empty? _____) _____)

          ((_____

            _____)

          (else _____)))
```

(b) Rewrite the program, but this time
USE HIGHER-ORDER FUNCTIONS, NOT RECURSION

Question 3 (3 points):

One or more of the following procedures generates an iterative process. Circle them. Don't circle the ones that generate a recursive process.

```
(define (butfirst-n num stuff)
        (if (= num 0)
                stuff
                (butfirst-n (- num 1) (bf stuff))))

(define (member? thing stuff)
        (cond ((empty? stuff) #f)
                ((equal? thing (first stuff)) #t)
                (else (member? thing (bf stuff)))))

(define (addup nums)
        (if (empty? nums)
                0
                (+ (first nums)
                   (addup (bf nums)))))
```

Question 4 (3 points):

Suppose that you have a procedure (f n) that requires time $\Theta(n)$, and a procedure (g n) that requires time $\Theta(n^2)$. What is the order of growth required for a program to compute (* (f n) (g n))? **There may be more than one correct answer; check all that are correct.**

_____A. $\Theta(n)$ _____B. $\Theta(n^2)$

_____C. $\Theta(n+n^2)$ _____D $\Theta(n^3)$

Question 5 (6 points): True or false?

_____A $\Theta(n^2)$ algorithm always takes longer than a $\Theta(n)$ one.

_____A recursive procedure generates an iterative process if it makes only one recursive call from each invocation

_____A recursive procedure does not generate an iterative process if it makes more than one recursive call from each invocation.

Question 6 (12 points):

Steve Sportman needs help in writing his tournament simulation program. He knows what he wants to do, but he's not quite sure how to implement his program since he is completely new to Scheme. You decide to help out.

This is an elimination tournament, in which the winner of a game in each round of the tournament is eligible to go on to the next round. A bracket is a list of the teams still eligible. You will present a bracket as a sentence containing the names of the teams. The teams are grouped in pairs that play each other is this round. For example, the bracket

(TeamA TeamB TeamC TeamD)

mean that Team A plays against TeamB, and TeamC plays against TeamD.

If there are an odd number of teams, the last team "gets a bye" and automatically goes on to the next round.

(a) The basic ingredient in simulating a tournament is to be able to simulate one game. We want to be able to try various approaches to determining the winner of each game, so various pieces of the program will take, as one of their arguments, a game simulator function. A game simulator is a function that takes two team names as its arguments, and returns the name of the winning team. For example, here's a game simulator that picks that winner at random:

```
(define (random-sim team1 team2)
        (if (zero? (random 2)) team1 team2))
```

Write a procedure called flip that takes a game simulator S as its argument. It returns another game simulator that's the exact opposite of S: If S picks its first arguments as the winner, this new simulator picks the second argument, and vice versa.

Question 6 continued:

(b) Steve wants to be able to simulate one round on the bracket so that

> (one-round simulate-game '(TeamA TeamB TeamC TeamD TeamE))
(TeamB TeamC TeamE)

(The result could also be (TeamB TeamD TeamE) or any other combination of a winner between A and B, a winner between C and D, and team E, which gets a bye in this round.)

One-round takes two arguments, a game simulator function and a bracket; it returns a new bracket that shows the teams to compete in the next round.

The example above shows that TeamB, TeamC, and TeamE won their rounds. (In the next round, TeamB is set to play TeamC, and TeamE will again get a bye.)

```
(define (one-round game-sim bracket)

    (cond ((empty? bracket) '())

    (_____________________________________________________

     _____________________________________________________)

    (else _____________________________________________

        _____________________________________________)))
```

(c) What Steve really wants is a procedure that finds the winner for an entire tournament by calling one-round repeatedly until only one team is left. Write tournament, with the same two arguments as one-round, that returns a word, the name of the winning team.