Midterm test 1 CIS280, Spring 2001, Saunders

Name:____________________

## 1 21 points

Write the value of each of the following lisp expressions. If the expression is an ill formed lisp expression, write "error".

_____ `(- (* 8  3) (/ 12 4))`

_____ `(+ (* (- 9 7) 2 5) 4)`

_____ `(* (- (+ (4 5) 3) 2))`

```
(define a 3)
(define b (- a 1))
```

_____ `(> a b)`

_____ `(and (= a b) (/ b (- a 3)))`

_____ `(or ( = a b) (+ 1 1) (< b a))`

_____ `(if (and (< b a) (< a (* b b))) a b)`

## 2 12 points

Testing your scoping knowledge. Assume we first make these definitions

```
(define x 2)
(define y (* 2 x))
(define z (+ x y ))
(define (fun1 x y)
  (+ z y z))
(define (fun2 a b)
  (let ((x (+ a b))
        (y (* 2 x)) )
    (+ x y z) ))
```

Now evaluate the following Scheme expressions: (Hint: if uncertain about scope rules for the `let`, first re-express it using `lambda`.)

_____ `(fun1 2 3)`

_____ `(fun1 x 3)`

_____ `(fun2 3 4)`

## 3 15 points

Procedures that return procedures. We can "smooth" a continuous function by changing the value of the function at each point to be the average of the values at nearby points. Assume we have a function $f(x)$, and a "nearby" distance $d$. Then define

$$\text{smooth}(x) = \frac{f(x-d) + f(x) + f(x=d)}{3}.$$

Part A. Write `smoother` in Scheme. The inputs are a function $f$ of one argument and a distance $d$. The output is a *new function of one argument, $x$,* defined as the smooth above. Hint: use lambda to create this new function. Here is a start:

```
(define smoother
  (lambda (f d) ; the 2 arguments to smoother are a function and a distance.
          ; body which creates a function of x
          ...
  )
)
```

Part B. Show what you would type in the Scheme interpreter to find the smoothed value of `(sin x)` at x = 2, for distance 0.01. You don't have to show the result the interpreter would produce!

## 4 20 points

Consider the function

```
(define f
  (lambda (n)
          (if (< n 1)
              0
              (+ 1 (f (quotient n 2))) )))
```

Part A. What is

______ (f 4)

______ (f 6)

______ (f 8)

______ (f (expt 2 100))

______ (f (- (expt 2 100) 1))

Part B. Write an iterative version of $f$. Your helper function may be defined inside or outside your definition of $f$. Hint: Remember that the main body of $f$ merely provides the initial values to the helper function.

## 5 12 points

The C++ function `sums` below is designed to return the number of ways a positive integer n can be written as a sum of positive integers. In counting the number of ways, the order of writing the terms is considered irrelevant, so for example 7 + 2 + 1 is the same way as 1 + 7 + 2. Thus sums(4) is 5 because the ways are 1 + 1 + 1 + 1, 1 + 1 + 2, 1 + 3, 2 + 2, and 4. Notice that just writing 4 is counted as a sum (it is considered a sum of one term).

```
int sums(int n){ return sums_with_max(n, n); }

// n is the number whose sums to compute.  m is the max term to use.
// Requiring 0 < m <= n.
int sums_with_max(int n, int m)
{  if (n == 0 || k == 1) return 1;
   else
     return sums_with_max(n, m-1)// sums that don't use an m.
            +
            sums_with_max(n-m, min(m, n-m)); // for the sums using at least 1 m.
}
```

Part A. Write `sums` (and `sums_with_max`) in Scheme.

Part B. Which is true, circle it, of the process generated by `sum`.

It is: iterative, linear-recursive (but not iterative), or tree recursive.

## 6 20 points

Exercise on speed of fast and slow exponentiation.

```
(define (expt-v1 b e)
  (if (zero? e)
      1
      (* b (expt-v1 b (- e 1))) ))

(define (expt-v2 b e)
  (cond ((zero? e) 1)
        ((even? e) (square (expt-v2 b (quotient e 2))))
        ((odd? e)  (* b (expt-v2 b (- e 1)))) ))
```

Part A. (Remark: 1024 is `(expt 2 10)` and $1048576 = 1024^2 =$ `(expt 2 20)`.) Answers which are correct to within a factor of 2 will be counted as correct.

______ If `(expt-v1 1.01 1024)` takes one second, about how long do you expect `(expt-v1 1.01 2048)` to take?

______ If `(expt-v1 1.01 1024)` takes one second, about how long do you expect `(expt-v1 1.01 1048576)` to take?

______ If `(expt-v2 1.01 1024)` takes one second, about how long do you expect `(expt-v2 1.001 2048)` to take?

______ If `(expt-v2 1.01 1024)` takes one second, about how long do you expect `(expt-v2 1.001 1048576)` to take?

______ If `(expt-v1 1.01 1024)` takes one second on computer X, about how long do you expect `(expt-v2 1.01 1024)` to take on computer X?

Part B. Write a definition of `expt-v1` which uses `cond` rather than `if`.

Part C. Write a definition of `expt-v2` which uses `if`(s) rather than `cond`.

Gee-whiz question about compound interest ( This is not part of the test): If you invest 1 dollar at 1% interest per month, after 100 years, what do you have?
Answer: (expt 1.01 1200), which is 153337 dollars, a 5 orders of magnitude increase!