Midterm Examination, Cmput 325
October 24, 1996

INSTRUCTIONS: This is an open book, open notes exam. The time for this exam is 80 minutes, and the mark total is also 80. Write all answers in these pages. Write legibly. Write your name and ID number on top of this page now.

1. (16 marks)

1.1 (10 marks) Reduce the following λ-expressions to their normal forms. Show all the reduction steps.

(a)

$$
\begin{aligned}
&(\lambda x \mid (\lambda xy \mid xy)xy)s \\
&\rightarrow (\lambda xy \mid xy)sy \\
&\rightarrow sy
\end{aligned}
$$

(b)

$$
\begin{aligned}
&(\lambda xy \mid xxy)(\lambda x \mid xy) \\
&\rightarrow (\lambda w \mid (\lambda x \mid xy)(\lambda x \mid xy)w) \\
&\rightarrow (\lambda w \mid (\lambda x \mid xy)yw) \\
&\rightarrow (\lambda w \mid yyw) \\
&\rightarrow_\eta yy
\end{aligned}
$$

(c)

$$
\begin{aligned}
&((\lambda xy \mid x\,(\lambda y \mid (+\,y\,2)))\;(\lambda fx \mid f\,(f\,x)))\;5 \\
&\rightarrow (\lambda y \mid (\lambda fx \mid f\,(f\,x))\,(\lambda y \mid (+\,y\,2)))\;5 \\
&\rightarrow (\lambda fx \mid f\,(f\,x))\,(\lambda y \mid (+\,y\,2)) \\
&\rightarrow (\lambda x \mid (\lambda y \mid (+\,y\,2))((\lambda y \mid (+\,y\,2))x) \\
&\rightarrow (\lambda x \mid (\lambda y \mid (+\,y\,2))(+\,x\,2)) \\
&\rightarrow (\lambda x \mid (+\,(+x\,2)\,2)))
\end{aligned}
$$

1.2 (6 marks) Consider the following lambda expressions. Indicate which ones can be converted to a common expression. You should consider all three kinds of conversion, i.e. α-conversion, β-conversion, and η-conversion.

(1) $(\lambda xy \mid x(xy))\,(\lambda z \mid sz)$

(2) $(\lambda yx \mid y(yx))\,(\lambda z \mid sz)$

(3) $(\lambda w \mid ssw)$

(4) ss

(1) can reduce to (2) by three steps of α-conversion (think about why three steps are needed). (3) reduces to (4) by η-reduction.

Many of you made the mistake that (1) reduces to (3). This is not the case.

$$
\begin{aligned}
&(\lambda xy \mid x(xy))\,(\lambda z \mid sz) \\
&\to (\lambda y \mid (\lambda z \mid sz)((\lambda z \mid sz)y)) \\
&\to (\lambda y \mid (\lambda z \mid sz)(sy)) \\
&\to (\lambda y \mid s(sy)) \\
&\to_\alpha (\lambda w \mid s(sw))
\end{aligned}
$$

This is different from (3).

2. (22 marks)

2.1 (6 marks) Define the logic connective IMPLY in terms of T and F in lambda calculus. Reduce your definition to normal form if it is not already in normal form.

Recall the truth table for logic implication is:

```
   x     y       (IMPLY x y)
-----------------------------
   T     T           T
   F     T           T
   F     F           T
   T     F           F
-----------------------------
```

$$(\lambda xy \mid xyT)$$

But there are many other solutions that are also correct, though not as simple as this one.

2.2 (6 marks) In the language of abstract programming, we can define a function for membership test using letrec as follows:

```
    letrec member(x,y) = if null(y) then F
                         else if eq(x, car(y)) then T
                         else member(x, cdr(y))
    in ....
```

Code this function using pure lambda notation. Assume the primitives null, eq, car, and cdr.

$$Y\,(\lambda fxy \mid (null\,y)\,F\,((eq\,x\,(car\,y))\,T\,(f\,x\,(cdr\,y))))$$

2.3 (4 marks) What are the values for x and y when the innermost $x + y$ is evaluated?

```
  let x = 5 and
      y = 8 in let x = x+3 and
                   y=x+y+1 in x+y

answer: x is 8 and y is 14
```

2.4 (6 marks) Consider

$$((\lambda f \mid (\lambda x \mid fx))\ (\lambda x \mid x))\ s$$

(a) (2 mark) what is the normal form of this lambda expression? The normal form is s.

(b) (4 marks) In the context-based interpreter, substitution is not performed right away as in β-conversion. Instead, bindings for formal parameters are recorded in a context. Show the context when fx is being evaluated.

Assume at the time this expression is evaluated, the context is CT0.
Then, fx is evaluated in the context $\{x -> s,\ f -> [\lambda x \mid x, CT0]\}$

3. (8 marks) Consider the following machine level representation:

```
       -------    -------
  S ->|  |  | -> |  |  | -> e
       -------    -------
         |          |
       -------     -------          -------
       |  | /|     |  |  |------------ >|  | /|
       -------     -------          -------
         |           |                |
         a         -------   -------   -------
                   |  |  | -> |  |/ |   |  | /|
                   -------   -------   -------
                     |         |         |
                     b         c         d
```

(a) (5 marks) Give two different s-expressions that are represented internally by the above structure (starting from S).

```
((a)  ((b c) (d)) . e)
((a)  ((b c) (d . nil)) . e)
```

(b) (3 marks) Show the changes (addition, redirection of links, etc.) to the above structure when the following code is executed:

```
(cons (car S) '(1 2))
```

When cons is invoked, a new memory cell

```
-------
|  |  |
-------
 |   |
 V   V
```

is created, where the car part (the left pointer) points to the location of the substructure in S that represents (a), and the cdr part points to a new structure that represents list (1 2).
In this example, no redirection of the existing links will ever happen.

4. (16 marks)
4.1 (8 marks) Consider the following Lisp definition:

```
(defun f (L)
       (cond ((null L) nil)
             (t (cons (cons (caar L) (cadar L)) (f (cdr L))))
       )
)
```

What will be returned after executing the following expressions?

```
                (f '((a 1) (b 2) (c 3)))
your answer: ((a . 1) (b . 2) (c . 3))

                (f '(((a b) (1 2)) ((b c) (2 3))))
your answer: (((a b) . (1 2)) ((b c) . (2 3)))
```

In one sentence, describe what function f does.

f takes a list of sublists and returns the list of dotted pairs, each formed from the first two elements in each of the sulists.

4.2 (8 marks) Consider the following Lisp definition:

```
        (defun g (X Y)
            (cond ((null Y) nil)
   ((null (cdr Y)) nil)
                  ((eq X (car Y)) (cons (cadr Y) (g X (cdr Y))))
                  (t (g X (cdr Y)))
            )
        )
```

What will be returned after executing the following expressions?

```
           (g 'd '(a b c d e f g))
your answer: (e)

           (g 'd '(a d f g d e r d u i))
your answer: (f r u)
```

What does g do?

(g X Y) returns a list of elements in Y that immediately follow the occurrences of X.

5. (18 points) In solving the following problems, you may use any Lisp primitive or built-in functions that have been allowed in the assignments.

5.1 (8 marks) Define a function

```
(defun last (L) ....)
```

which takes a non-empty list L and returns the last element in L. For example, `(last '(a b (c) (d)))` returns `(d)`.

```
(defun last (L)
        (if (null (cdr L))
            (car L)
            (last (cdr L))
        )
)
```

5.2 (10 marks) Define a function

```
(defun flattern (L) ....)
```

which takes a list L and returns the list of all the atoms appearing in L. For example, `(flattern '(((a) b) c (c) (d)))` returns `(a b c c d)`.

```
(defun flattern (L)
      (cond  ((null L) nil)
             ((atom (car L)) (cons (car L) (flattern (cdr L))))
             (t (append (flattern (car L)) (flattern (cdr L))))
       )
)
```