Your Name: ______________________ Your EID: ____________

Circle Your Discussion Section:
54075: Ian Wehrman, Friday, 9:00 – 10:00a, ENS 109
54080: Ian Wehrman, Friday, 10:00 – 11:00a, GSB 2.122
54085: David Rager, Friday, 10:00 – 11:00a, JES A218A
54088: Behnam Robatmili, Friday, 12:00 – 1:00p, RLM 5.122
54090: Behnam Robatmili, Friday, 1:00 – 2:00p, CBA 4.344
54095: Nathan Wetzler, Friday, 1:00 – 2:00p, JES A209A

Midterm Exam 2

CS313K Logic, Sets, and Functions – Spring, 2009

Instructions

Write your name and EID above and circle the unique ID of your discussion section! Write your answers in the space provided. If your proofs fill more than the space provided, you may write on the back of the page but please put "PTO" ("please turn over") at the bottom and put the Question number at the top of each back page you use. If you use extra paper, be sure to put your name and EID and the Question number on each page!

There are 10 questions worth a total of 200 points. Those requiring proofs are worth more than those not requiring proofs. Partial credit will be given, so do your best on each question. You have until 5:00 pm.

The functions `app`, `tp` ("`true-listp`"), and `mem`, which have all been used in class, are the familiar functions of those names but I've included their definitions on the last page of the exam for your reference. Assume you don't know anything about the function symbols `P`, `Q`, `f`, `g`, and `h` except whatever is said about them in the statement of each problem.

You may refer to the course notes (the red book) during the exam. You may refer to your own notes if they are on paper. No computers are allowed. No talking is allowed. No cellphones. Remove sunglasses, hats, baseball caps, etc.

The last section of the exam has no questions. It just lists the `defun`s of `app`, `tp`, and `mem`.

Question 1 (10 points): Suppose this is a theorem:

`T1:  (P x y)` $\rightarrow$ `(f (g x) y)` $=$ `(f x y).`

Then is the following a theorem?

`((Q (f a a))` $\wedge$ `(P a a))` $\rightarrow$ `(Q (f (g a) a))`

Circle the correct answer:

<u>YES</u> **no**

Question 2 (10 points): Suppose this is a theorem:

`T1:  ((P x)` $\wedge$ `(P y))` $\rightarrow$ `((Q (f x y))` $\leftrightarrow$ `(Q y))`

Then is the following a theorem?

`((P a)` $\wedge$ `(Q a))` $\rightarrow$ `(Q (f a b))`

Circle the correct answer:

yes **<u>NO</u>**

Question 3 (10 points): Suppose this is a theorem (same `T1` as above):

`T1:  ((P x)` $\wedge$ `(P y))` $\rightarrow$ `((Q (f x y))` $\leftrightarrow$ `(Q y))`

Then is the following a theorem?

`((P a)` $\wedge$ `(Q a))` $\rightarrow$ `(Q (f a a))`

Circle the correct answer:

<u>YES</u> **no**

Question 4 (10 points): Suppose this is a theorem:

```
T1:  ((natp x) ∧ ¬(P y)) → ((Q (f x y)) ↔ (Q y))
```

Then is the following a theorem?

```
((¬(Q b)) ∧ (Q (f 23 b))) → (P b)
```

Of course, `natp` is the familiar function of that name from the notes.

Circle the correct answer:

<u>YES</u> **no**

Question 5 (10 points): Suppose we want to prove

```
(consp x) → (mem e x)
```

by induction. Is this a legal induction argument?

Base:

```
¬ (consp x) → ((consp x) → (mem e x))
```

Induction Step:

```
  ((consp x)
   ∧
   (mem e (cdr x)))
→
  (mem e x).
```

Circle the correct answer:

yes **<u>NO</u>**

The "induction hypothesis" above, `(mem e (cdr x))`, ought to be `(consp (cdr x)) → (mem e (cdr x))`.

Question 6 (20 points): Show a proof of or a counterexample to the following:

```
 ((A → (B ∧ C))
  ∧
  (B → D)
  ∧
  ((D ∧ C) → E))
→
  ((A ∧ Q) → E)

Proof:  In the proof below I separate equivalent
formulas by a line of hyphens.  Start with the
formula above.
-----------------------------------------------------------------
Promote and implicitly use associativity of and:
-----------------------------------------------------------------
 ((A → (B ∧ C))
  ∧
  (B → D)
  ∧
  ((D ∧ C) → E)
  ∧
  A
  ∧
  Q)
 → E
-----------------------------------------------------------------
Forward Chain [using hyps 1 and 4 (and associativity of and)]
[I wouldn't mind if they did not bother to specify which hypotheses
are involved in the forward chaining.]:
-----------------------------------------------------------------
 (B
  ∧
  C
  ∧
  (B → D)
  ∧
  ((D ∧ C) → E)
  ∧
  A
  ∧
  Q)
 → E
-----------------------------------------------------------------
Forward Chain [with hyps 1 and 3]:
-----------------------------------------------------------------
 (B
  ∧
  C
  ∧
```

```
  D
  ∧
  ((D ∧ C) → E)
  ∧
  A
  ∧
  Q)
 → E
----------------------------------------------------------------
Forward Chain (twice) [with hyps 2, 3, and 4]
----------------------------------------------------------------
 (B
  ∧
  C
  ∧
  D
  ∧
  E
  ∧
  A
  ∧
  Q)
 → E
----------------------------------------------------------------
Basic
----------------------------------------------------------------
t
Q.E.D.
```

Question 7 (30 points): Show a proof of or a counterexample to the following:

```
(tp (app nil a)).

Counterexample:  Let a be 7.  (tp (app nil 7)) = (tp 7) = nil.
```

Question 8 (30 points): Show a proof of or a counterexample to the following:

```
(e ≠ d) → ((mem e (cons d x)) ↔ (mem e x)).

Proof:
-----------------------------------------------------------------
Rewrite with def mem, simplifying with (endp (consp d x)) = nil
and if-ax1, and car-cons, and cdr-cons.  By the way, I wouldn't INSIST they
justified these simplifications if they wrote down the correct expanded form.
-----------------------------------------------------------------
(e ≠ d) → ((if (equal e d) nil (mem e x)) ↔ (mem e x)).
-----------------------------------------------------------------
Hyp 1
-----------------------------------------------------------------
(e ≠ d) → ((if nil nil (mem e x)) ↔ (mem e x)).
-----------------------------------------------------------------
If-ax2
-----------------------------------------------------------------
(e ≠ d) → ((mem e x) ↔ (mem e x)).
-----------------------------------------------------------------
Any student who just said ``this is obvious'' should get full
credit in my mind.  But if you had to prove it, you could
rewrite using the Tautology `def ↔' and then use Basic
three times.
-----------------------------------------------------------------
T

Q.E.D.
```

Question 9 (35 points): Suppose `h` is defined as follows:

```
(defun h (x y)
  (if (endp x)
      y
      (h (cdr x) y)))
```

Show a proof of or a counterexample to the following:

```
(h x y) = y.

Proof.  Induct on x with σ  ={x ← (cdr x)}.

Base Case:
¬(consp x) → (h x y) = y.
-----------------------------------------------------------------
Expand (h x y) using def h and the rule that ¬(consp x) → (endp x)=nil
and if-ax2.
-----------------------------------------------------------------
¬(consp x) → y = y.
-----------------------------------------------------------------
Reflexivity and Basic
-----------------------------------------------------------------
T

Induction Step:
((consp x)
 ∧
 (h (cdr x) y) = y)
→
(h x y) = y
-----------------------------------------------------------------
Expand (h x y) using def h and the rule that (consp x) → (endp x)=t
and if-ax1 and distinct constants.  (I'm content to ignore such details.)
-----------------------------------------------------------------
((consp x)
 ∧
 (h (cdr x) y) = y)
→
(h (cdr x) y) = y
-----------------------------------------------------------------
Basic.
-----------------------------------------------------------------
T
Q.E.D.
```

Question 10 (35 points): Suppose P23 is defined as follows:

```
(defun P23 (x)
  (if (endp x)
      t
      (if (equal (car x) 23)
          (P23 (cdr x))
          nil)))
```

You may assume the following four simple theorems about P23 and app. These follow easily from the definitions of P23 and app.

```
T1: ¬(consp x) → ((P23 x) ↔ t)
T2:   (consp x) → ((P23 x) ↔ ((car x)=23 ∧ (P23 (cdr x))))
T3: ¬(consp x) → (app x y)=y
T4:   (consp x) → (app x y) = (cons (car x) (app (cdr x) y))
```

Below is a "proof" of the formula

```
((P23 a) ∧ (P23 b)) → (P23 (app a b)).
```

Fill in the space between the braces {...} to explain each step. Some steps represent the application of several rules. You may omit uses of Rewriting with the Basic, Short Circuit, and Associativity of "∧" tautologies. But you should be especially careful to list every application of the following rules: Rewriting with Promotion or Forward Chain, Rewriting with the definitions of P23 or app, Rewriting with the theorems T1, T2, T3, or T4, use of the Hyp rule for hypothesis i, and use of Cases. If you get to an illegal step, just say so and don't go any further. Each formula below is well-formed, so don't worry about checking the parentheses etc. Partial credit will be given, so explain all the steps you can.

Theorem
((P23 a) $\wedge$ (P23 b)) $\rightarrow$ (P23 (app a b)).
Proof
{Induct on a with σ ={a←(cdr a)}}   (⟵ name the induction variable and show σ)

Base Case:
[¬(consp a) → (((P23 a) ∧ (P23 b)) → (P23 (app a b)))]

↔ {Promote}   (⟵ explain)

[(¬(consp a) ∧ (P23 a) ∧ (P23 b)) → (P23 (app a b))]

↔ {Rewrite with T1 [to get rid of (P23 a)]
and with T3 [to convert (app a b) to b]
[I would not expect them to explain why, just to
say ‘‘T1 and T3’’.]}   (⟵ explain)

[(¬(consp a) ∧ (P23 b)) → (P23 b)]

↔ {Hyp 2}   (⟵ explain)

[(¬(consp a) ∧ (P23 b)) → T]

↔

T

(Proof continued on next page.)

Induction Step:

```
[((consp a)
   ∧ (((P23 (cdr a)) ∧ (P23 b)) → (P23 (app (cdr a) b))))
  →
  (((P23 a) ∧ (P23 b)) → (P23 (app a b)))]
```

↔ {Promote} (⟵ explain)

```
[((consp a)
   ∧ (((P23 (cdr a)) ∧ (P23 b)) → (P23 (app (cdr a) b)))
   ∧ (P23 a)
   ∧ (P23 b))
 → (P23 (app a b))]
```

↔ {Rewrite with T2 [to convert (P23 a) to the conjunction of (car a)=23 and (P23 (cdr a))], and with T4 [to expand (app a b)]} (⟵ explain)

```
[((consp a)
   ∧ (((P23 (cdr a)) ∧ (P23 b)) → (P23 (app (cdr a) b)))
   ∧ (car a)=23
   ∧ (P23 (cdr a))
   ∧ (P23 b))
 → (P23 (cons (car a) (app (cdr a) b)))]
```

↔ {Forward Chain [using hyps 2,3,and 4]} (⟵ explain)

```
[((consp a)
   ∧ (P23 (app (cdr a) b)))
   ∧ (car a)=23
   ∧ (P23 (cdr a))
   ∧ (P23 b))
 → (P23 (cons (car a) (app (cdr a) b)))]
```

↔ {Hyp [3] [Again, I wouldn't mind if didn't specify which hyp.] } (⟵ explain)

```
[((consp a)
   ∧ (P23 (app (cdr a) b)))
   ∧ (car a)=23
   ∧ (P23 (cdr a))
   ∧ (P23 b))
 → (P23 (cons 23 (app (cdr a) b)))]
```

(Proof continued on next page.)

{The following formula is exactly the same as the one at the bottom of the preceding page.}

```
[((consp a)
  ∧ (P23 (app (cdr a) b)))
  ∧ (car a)=23
  ∧ (P23 (cdr a))
  ∧ (P23 b))
→ (P23 (cons 23 (app (cdr a) b)))]
```

↔ {`Rewrite with def P23, Hyp [1]`} (⟵ explain)

```
[((consp a)
  ∧ (P23 (app (cdr a) b)))
  ∧ (car a)=23
  ∧ (P23 (cdr a))
  ∧ (P23 b))
→ (P23 (app (cdr a) b))]
```

↔

`T`

□

Familiar Definitions

```
(defun app (x y)
  (if (endp x)
      y
      (cons (car x)
            (app (cdr x) y))))

(defun tp (x)           ; Called ``true-listp'' in the notes.
  (if (endp x)
      (equal x nil)
      (tp (cdr x))))

(defun mem (e x)
  (if (endp x)
      nil
      (if (equal e (car x))
          t
          (mem e (cdr x)))))
```