Name: Hemos ID:

CSE-490 Logic in Computer Science 2006
Final exam

- This is a closed-book exam. No other material is permitted.
- It consists of 6 problems worth a total of 200.
- There are 16 pages in this exam, including 4 work sheets.
- Try to use work sheets before writing your answers. Write your answers clearly and legibly.
- You have 3 hours for this exam.

	Problem 1	Problem 2	Problem 3	Problem 4	Problem 5	Problem 6	Total
Score							
Max	60	35	20	20	45	20	200

1 Short answers [60 pts]

1.1 Proof terms [10 pts]

Question 1. [5 pts] Show a proof term of type $(A \supset B) \supset (A \supset (B \wedge A))$.

Question 2. [5 pts] Show a proof term of type $((A \vee B) \wedge \neg B) \supset A$.

1.2 Classical logic [5 pts]

Question 3. [5 pts] In order to prove A *true* using the sequent calculus for classical logic, which of $A \Longrightarrow \cdot$ and $\cdot \Longrightarrow A$ should we prove?

1.3 CPS translation [15 pts]

For each case below, complete the definition of the CPS translation M° of a given proof term M. Please write A° for the translation of type A.

Question 4. [10 pts]

Case $\dfrac{\Gamma; \Delta \vdash_{\mathsf{K}} M : A \supset B \quad \Gamma; \Delta \vdash_{\mathsf{K}} N : A}{\Gamma; \Delta \vdash_{\mathsf{K}} M\ N : B}$

$(M\ N)^\circ =$

Question 5. [5 pts]

Case $\dfrac{\Gamma; \Delta \vdash_{\mathsf{K}} M : A \quad x : A\ \mathit{false} \in \Delta}{\Gamma; \Delta \vdash_{\mathsf{K}} \mathsf{throw}\ M\ \mathsf{to}\ x : C}$

$(\mathsf{throw}\ M\ \mathsf{to}\ x)^\circ =$

1.4 Linear logic [30 pts]

Question 6. [5 pts] $(A \mathbin{\&} (B \otimes C)) \multimap ((A \mathbin{\&} B) \otimes (A \mathbin{\&} C))$ *true* is provable. True or false?

Question 7. [5 pts] $(A \mathbin{\&} (B \oplus C)) \multimap ((A \mathbin{\&} B) \oplus (A \mathbin{\&} C))$ *true* is provable. True or false?

Question 8. [5 pts] $(!(A \mathbin{\&} B)) \multimap (A \otimes B)$ *true* is provable. True or false?

Question 9. [10 pts] Suppose that we use the following elimination rule for $\mathbf{0}$:

$$\frac{\Delta \Vdash \mathbf{0}\ \mathit{true}}{\Delta \Vdash C\ \mathit{true}}\ \mathsf{0E}$$

Then, is the following rule derivable?

$$\frac{\Delta \Vdash \mathbf{0}\ \mathit{true}}{\Delta, \Delta' \Vdash C\ \mathit{true}}\ \mathsf{0E}'$$

If derivable, explain how the derivation works. If not, state so.

Question 10. [5 pts] Give a proof term for $(!A \otimes !B) \multimap (A \mathbin{\&} B)$ *true*.

2 Judgmental formulation of falsehood judgments [35 pts]

In this problem, we will build a system of logic which is based on falsehood judgments $A\ \mathit{false}$ instead of truth judgments $A\ \mathit{true}$. The set of propositions is inductively defined as follows:

$$A \quad = \quad A \vee A \mid A \wedge A \mid \perp \mid \top \mid \neg A$$

Note that we do not use implication $\supset$.

The invariant to be maintained is that $A\ \mathit{false}$ is provable in our system if and only if $\neg A\ \mathit{true}$ is provable in classical logic. Since $A\ \mathit{true}$ is provable in classical logic if and only if $\neg\neg A\ \mathit{true}$ is provable, our system is actually another formulation of classical logic in that it is capable of simulating classical logic. That is, to test the provability of $A\ \mathit{true}$ in classical logic, we test the provability of $\neg A\ \mathit{false}$ in our system.

Write your answer in the natural deduction style, but do **not** use hypothetical judgments. Your answer should **never** use truth judgments $A\ \mathit{true}$.

Question 1. [5 pts] Propose introduction and elimination rules for disjunction $\vee$.

Question 2. [5 pts] Propose introduction and elimination rules for conjunction $\wedge$.

Question 3. [5 pts] Show a local reduction $\Longrightarrow_R$ for conjunction $\wedge$.

Question 4. [5 pts] Show a local expansion $\Longrightarrow_E$ for conjunction $\wedge$.

Question 5. [5 pts] Propose introduction and elimination rules for falsehood $\perp$ and truth $\top$.

Question 6. [5 pts] Propose introduction and elimination rules for negation $\neg$.

Question 7. [5 pts] Give a proof of $\neg(A \vee \neg A)$ *false*.

3 First-order logic [20 pts]

In this problem, we will examine the logical equivalence between $\exists x.A$ and $\neg\forall x.\neg A$.

Please write your answer in the natural deduction style, but do **not** use hypothetical judgments.

Question 1. [5 pts] Prove $(\exists x.A) \supset \neg\forall x.\neg A \ \mathit{true}$ in constructive logic.

Question 2. [5 pts] Prove $(\neg\exists x.A) \supset (\forall x.\neg A) \ \mathit{true}$ in constructive logic. (Question 2-1 in Assignment 4)

Question 3. [5 pts] Use the result in the previous question to prove a modified judgment $(\neg\forall x.\neg A) \supset \neg\neg\exists x.A\ \mathit{true}$ in constructive logic.

Question 4. [5 pts] Is $(\neg\forall x.\neg A) \supset \exists x.A\ \mathit{true}$ provable in classical logic? True or false? Explain why in **<u>no more than three sentences</u>**.

4 Sequent calculus for first-order logic [20 pts]

In this problem, we will examine the following logical equivalence

$$\forall x.A(x) \supset B \equiv (\exists x.A(x)) \supset B$$

where $A(x)$ indicates that a term variable x is found in A whereas B does not contain x. (Quantifiers have the lowest operator precedence, so we have $\forall x.A(x) \supset B = \forall x.(A(x) \supset B)$.

For this problem, we will use sequent calculus exclusively. Please do **not** write your proof in the natural deduction style.

Question 1. [10 pts] Give a proof of $\forall x.A(x) \supset B \longrightarrow (\exists x.A(x)) \supset B$. If not provable, state so.

Question 2. [10 pts] Give a proof of $(\exists x.A(x)) \supset B \longrightarrow \forall x.A(x) \supset B$. If not provable, state so.

5 Proof term with definitional equality [45 pts]

In class, we defined the reduction relation $\Longrightarrow_R$ between terms as follows:

$$\begin{array}{rcl}(\lambda x \in \tau.\, t)\; s & \Longrightarrow_R & [s/x]t \\ \mathbf{rec}\; f(\mathbf{0})\; \mathbf{of}\; f(\mathbf{0}) \Rightarrow t_0 \mid f(\mathbf{s}(x)) \Rightarrow t_s & \Longrightarrow_R & t_0 \\ \mathbf{rec}\; f(\mathbf{s}(t))\; \mathbf{of}\; f(\mathbf{0}) \Rightarrow t_0 \mid f(\mathbf{s}(x)) \Rightarrow t_s & \Longrightarrow_R & [\mathbf{rec}\; f(t)\; \mathbf{of}\; f(\mathbf{0}) \Rightarrow t_0 \mid f(\mathbf{s}(x)) \Rightarrow t_s/f(x)][t/x]t_s\end{array}$$

We say that two terms t and s are *definitionally equal*, written as $t = s$, if there exists a term r such that $t \Longrightarrow_R^* r$ and $s \Longrightarrow_R^* r$ where $\Longrightarrow_R^*$ stands for the reflexive and transitive closure of $\Longrightarrow_R$. As in Assignment 5, we assume that $\Longrightarrow_R$ can be applied to any part of t or s. For example, if t and s are definitionally equal, $\mathbf{s}(t)$ and $\mathbf{s}(s)$ are definitionally equal, too. In most cases, $t = s$ holds because either $t \Longrightarrow_R^* s$ or $s \Longrightarrow_R^* t$ holds.

Consider the system of first-order logic with datatypes augmented with a new rules DefEq exploiting definitional equality:

$$\frac{\Gamma \vdash M : A(t) \quad t = s}{\Gamma \vdash M : A(s)}\;\mathsf{DefEq}$$

That is, if terms t and s are definitionally equal and proof term M is shown to have type $A(t)$, then M can be assigned type $A(s)$ as well. Our goal is to use the rule DefEq to find a proof term of the following type (or proposition), which states that every natural number is either even or odd:

$$\forall x \in \mathsf{nat}.(\exists y \in \mathsf{nat}. y + y =_{\mathsf{N}} x) \vee (\exists y \in \mathsf{nat}. \mathbf{s}(y + y) =_{\mathsf{N}} x)$$

- We write $t + s$ for $plus\; t\; s$ where $plus$ is defined as follows:

$$\begin{array}{l} plus \in \mathsf{nat} \rightarrow \mathsf{nat} \rightarrow \mathsf{nat} \\ plus \;=\; \lambda x \in \mathsf{nat}.\, \lambda y \in \mathsf{nat}.\, \mathbf{rec}\; p(x)\; \mathbf{of}\; p(\mathbf{0}) \Rightarrow y \mid p(\mathbf{s}(z)) \Rightarrow \mathbf{s}(p(z))\end{array}$$

- You may use a proof term $eqNat$ of type $\forall x \in \mathsf{nat}. x =_{\mathsf{N}} x$ in your answer.
- You may use a proof term $trans$ of type $\forall x \in \mathsf{nat}. \forall y \in \mathsf{nat}. \forall z \in \mathsf{nat}. x =_{\mathsf{N}} y \supset y =_{\mathsf{N}} z \supset x =_{\mathsf{N}} z$ in your answer.
- We use the following typing rules for proof terms for $=_{\mathsf{N}}$.

$$\frac{}{\Gamma \vdash \mathbf{eqI_0} : \mathbf{0} =_{\mathsf{N}} \mathbf{0}}\; =_{\mathsf{N}}\mathsf{I}_0 \qquad \frac{\Gamma \vdash M : m =_{\mathsf{N}} n}{\Gamma \vdash \mathbf{eqI_s}(M) : \mathbf{s}(m) =_{\mathsf{N}} \mathbf{s}(n)}\; =_{\mathsf{N}}\mathsf{I}_{\mathsf{s}}$$

$$\frac{\Gamma \vdash M : \mathbf{0} =_{\mathsf{N}} \mathbf{s}(n)}{\Gamma \vdash \mathbf{eqE_{0s}}(M) : C}\; =_{\mathsf{N}}\mathsf{E}_{0\mathsf{s}} \qquad \frac{\Gamma \vdash M : \mathbf{s}(m) =_{\mathsf{N}} \mathbf{0}}{\Gamma \vdash \mathbf{eqE_{s0}}(M) : C}\; =_{\mathsf{N}}\mathsf{E}_{\mathsf{s}0} \qquad \frac{\Gamma \vdash M : \mathbf{s}(m) =_{\mathsf{N}} \mathbf{s}(n)}{\Gamma \vdash \mathbf{eqE_s}(M) : m =_{\mathsf{N}} n}\; =_{\mathsf{N}}\mathsf{E}_{\mathsf{s}}$$

Question 1. [5 pts] $\mathbf{0} + x = x$ holds. True or false?

Question 2. [5 pts] $x + \mathbf{0} = x$ holds. True or false?

Question 3. [5 pts] $\mathbf{s}(x) + y = \mathbf{s}(x + y)$ holds. True or false?

Question 4. [5 pts] Define a proof term $comp$ of the following type. You may use proof term $eqNat$.

$$\forall x \in \mathsf{nat}. \forall y \in \mathsf{nat}. x + \mathbf{s}(y) =_{\mathsf{N}} \mathbf{s}(x + y)$$

Question 5. [5 pts] Define another proof term $comp'$ of the following type. You may use proof term $comp$.

$$\forall x \in \mathsf{nat}. \mathbf{s}(x) + \mathbf{s}(x) =_{\mathsf{N}} \mathbf{s}(\mathbf{s}(x + x))$$

Question 6. [20 pts]
Give a proof term of the following type:

$$\forall x \in \mathsf{nat}. (\exists y \in \mathsf{nat}. y + y =_{\mathsf{N}} x) \vee (\exists y \in \mathsf{nat}. \mathbf{s}(y + y) =_{\mathsf{N}} x)$$

You may use proof terms $comp'$ and $trans$ in your answer. You may omit type annotations in injection terms. For example, you may write $\mathsf{inl}\ M$ when $\mathsf{inl}_A\ M$ is expected.

6 Linear soju machine [20 points]

In this problem, we will work out a linear encoding of a soju machine:

- For 250 won, it gives a bottle of soju.

For the sake of simplicity, we assume that the machine accepts only nickels (worth 50 won each) and dimes (worth 100 won each).

We use the following propositions, each of which may be interpreted as a resource.

- N: the machine has a nickel in its coin receptacle.
- D: the machine has a dime in its coin receptacle.
- SJ: the machine has a bottle of soju in its storage bin.

We write A^n for a notational abbreviation for $A \otimes A \otimes \cdots \otimes A$ (n times). For example, N^5 means that the machine has 5 nickels in its coin receptacle.

In the questions below, you will give a set Γ of axioms such that if the machine can make a state transition from X to Y, then $\Gamma; \cdot \vdash X \multimap Y$ is derivable in DILL and vice versa. All your axioms should be *local* in the sense explained below.

Suppose that there are m nickels and n dimes in the coin receptacle and l bottles of soju in the storage bin. We can describe the machine state as

$$N^m \otimes D^n \otimes SJ^l.$$

Then there are *infinitely many* legal state transitions for the machine. For example,

$$\begin{array}{l} N^5 \otimes D^2 \otimes SJ^{10} \multimap D^2 \otimes SJ^9 \\ N^7 \otimes D^2 \otimes SJ^8 \multimap N^2 \otimes D^2 \otimes SJ^7 \\ N^6 \otimes SJ \multimap N \\ \cdots \end{array}$$

Needless to say, we do not want to enumerate all these state transitions as axioms for the machine. Instead we would like to capture only essential, minimal, or *local* state transitions with *finitely many* axioms. This is feasible in linear logic (which is in fact the quintessential characteristic of linear logic). For example, the following single axiom accounts for all state transitions that deposit 5 nickels and dispense a bottle of soju:

$$N^5 \otimes SJ \multimap \mathbf{1}$$

Thus each of your axiom should have the form $X \multimap Y$ and also be local in this sense. A good working definition of a *local* axiom $X \multimap Y$ is that it neither X nor Y contains variables (such as m, n, l above) to count the number of resources.

Our prototype machine works by first accepting coins from a user. Suppose that you have inserted m nickels and n dimes. If there are l bottles of soju left in the storage bin, we can describe the machine state as

$$N^m \otimes D^n \otimes SJ^l,$$

which we take as the initial state of the machine. Then the machine dispenses a bottle of soju for each combination of coins **exactly** totaling 250 won. Here are two examples:

- Suppose that in the initial state, there are a total of 5 nickels and 2 dimes in the coin receptacle and 10 bottles of soju in the storage bin:

$$N^5 \otimes D^2 \otimes SJ^{10}$$

The machine decides to deposit 5 nickels and dispense a bottle of soju. Then there remain 2 dimes in the coin receptacle and 9 bottles of soju in the storage bin:

$$D^2 \otimes SJ^9.$$

Thus the state transition is described by

$$N^5 \otimes D^2 \otimes SJ^{10} \multimap D^2 \otimes SJ^9.$$

- Suppose that in the initial state, there are only 3 dimes in the coin receptacle. Since no combination of coins for 250 won is possible, the machine does not dispense a bottle of soju.

Question 1. [5 pts] Give a finite set Γ of axioms for this machine.

Question 2. [5 pts] Suppose that the machine is malfunctioning — it *may or may not* dispense a bottle of soju when depositing 250 won. What axioms need to be added?

Question 3. [10 pts] We have fixed the malfunctioning machine and also upgraded it so that it dispenses a bottle of soju if it has **at least** 250 won in its coin receptacle. Give **no more than two** axioms for this upgraded machine. Those axioms that you gave for the first machine are **no longer** used.

Work sheet

Work sheet

Work sheet

Work sheet