Instructor: Jia You

Final Examination, Cmput 325
Dec. 15, 2000, 9AM at CAB 243

LAST NAME ____________________________________

FIRST NAME ____________________________________

STUDENT ID ____________________________________

INSTRUCTIONS: This is a closed book exam. The time for this exam is 2 hours.

QUESTION 1 (20 marks): ______________________

QUESTION 2 (25 marks): ______________________

QUESTION 3 (27 marks): ______________________

QUESTION 4 (20 marks): ______________________

QUESTION 5 (8 marks): ______________________

TOTAL:______________________ OUT OF 100

1. [20 marks]
We use the following logic connectives: $\neg$ for *not*, $\wedge$ for *and*, $\vee$ for *or*, and $\leftarrow$ for *logic implication*; $\forall$ is the universal quantifier and $\exists$ the existential quantifier.

[2 marks] Consider the propositional formula $(c \vee d) \leftarrow (a \vee b)$ where a, b, c, and d are propositions. Show one *interpretation* in which the formula evaluates to FALSE.

[4 marks] Consider a formula of predicate logic:

$$[\exists X\ p(f(b), X)] \leftarrow [\forall X\ q(X, a)]$$

where a and b are constant symbols, f is a function symbol, and q and p are predicate symbols. Give one interpretation in which the formula evaluates to TRUE.

[4 marks] Consider the following disjunctive normal form.

$$(a \wedge b) \vee (\neg a \wedge b \wedge c) \vee (\neg a \wedge \neg c)$$

Give all the models of this formula. In addition, indicate which models are minimal. Use the notation that we used in the last assignment: a model is denoted by a set M of ground atoms, those that are in M are assigned to true and those that are not in M are assigned to false.

[2 marks] Let $\Gamma = \{a \vee b \vee c,\ d \leftarrow a,\ d \leftarrow b\}$. Does the relation $\Gamma \models d$ hold? Justify your answer.

[4 marks] For each pair of atoms below, determine whether they are unifiable. If yes, you should also show a unifier. Assume X, Y, and Z are variables.

(a) $p(g(a), Y, g(Y))$
$p(g(X), g(X), g(g(a)))$

(b) $p(X, Y, f(X, a))$
$p(f(a, Z), b, f(f(a, a), Z))$

[4 marks] Consider Γ consisting of the following clauses:

$$\begin{array}{l} p(a,b) \leftarrow \\ p(X,X) \leftarrow p(X,Y) \\ p(X,g(Y)) \leftarrow p(X,Y), q(Y,a) \\ q(b,a) \leftarrow p(a,a) \\ q(X,g(Y)) \leftarrow q(X,Y) \end{array}$$

where X and Y are variables and a and b are constants.

- Show five atoms ϕ from the Herbrand base such that $\Gamma \models \phi$.

- Show five atoms ϕ from the Herbrand base such that $\Gamma \not\models \phi$.

2. (25 marks)

[12 marks] Consider the following Prolog program

```
p([], L).
p([A|X],[A|Y]) :- p(X,Y).
s(X,Y) :- p(X,Y).
s(X,[A|Y]) :- s(X,Y).
```

For each goal below, indicate the result of the Prolog execution. In the case that the goal is proved, you should also show the bindings for the variables in the goal, if there is any. You only need to show the first set of bindings given by Prolog.

```
?- s([d,e], [a,b,c,d]).
```

```
?- s([d], [a,b,c,W]).
```

```
?- s([a,W], [a,b,c,W]).
```

```
?- s([a,b,c], [b|W]).
```

[8 marks] Consider the following program and goal

```
a(X,Y,Z) :- b(X,Y), c(Y,Z).
c(X,Y) :- d(X), g(Y).

b(1,W).

g(1).
g(2).
d(3).
d(4).

?- a(X,Y,Z).
```

Show all the answers (even repeated ones) that will be generated by Prolog for the goal (of course, assuming that the user types a ; after each answer is generated). Your answers should be shown in the order in which they are generated.

Note: Missing answers, extra answers, and incorrect order of answers will all result in reduction of marks.

[5 marks] Now suppose that the second clause in the above program is replaced by

```
c(X,Y) :- d(X), !, g(Y).
```

Indicate which answer(s) you gave above will not be generated for the same goal. If no answers will be generated, then just say so.

3. (27 marks)
Your Prolog solutions should be clean and understandable. You may not get full credits just because your program seems to work. Comments are not necessary unless you feel otherwise. You may use any Prolog builtin predicates. As a hint, in each question I indicate the size of my solution.

[8 marks] Consider the problem of defining a Prolog predicate

```
odd(X)
```

which is true if X is an odd, non-negative number. In this question you are going to define the predicate twice under two different representations of numbers.

(a) Define the predicate using the following representation:

```
   0  represents 0
s(X)  represents the successor of X, for any non-negative number X.
```

(My solution consists of three clauses without using the cut !.)

(b) Define the predicate using the Prolog builtin representation of numbers: 0, 1, 2, ... (My solution consists of three clauses without using the cut !.)

[5] We are interested in generating prefixes of a list. For example, the prefixes of the list `[a,b,c]` are `[]`, `[a]`, `[a,b]`, and `[a,b,c]`. You should define a Prolog predicate:

```
prefix(P, L)
```

where `L` is a given list of elements, and `P` is a prefix of `L`. Your definition should allow the user to generate all alternative prefixes without the repeated ones. The order in which the prefixes are generated is unimportant. (My solution has only two clauses without using the cut.)

[8 marks] Define a Prolog predicate

```
sum(L, S)
```

where L is a given (possibly nested) list of numbers and S is the sum of all these numbers in L. For example,

```
sum([4, 5, [1, [2]], 1], W)
```

should have W bound to 13.

Your program should terminate after the first answer is generated. Marking guide: 6 marks for correctly generating the first answer, and 4 marks for not generating additional answers. (My solution consists of five clauses and the cut is used.)

[6] Define a Prolog predicate

```
shift(L1,L2)
```

so that L2 is L1 'shifted rotationally' by one element to the right. The last element in L1 thus becomes the first in L2. You may assume that L1 contains at least two elements.

```
Example:
      ?- shift([1,2,3,4], L).
```

produces `L = [4,1,2,3]`. You program should terminate after the first answer is generated. (My solution consists of five clauses, two of which are used to define "append".)

4. (20 marks)

4.1 (12 marks) First consider the following Lisp definition

```
(defun h (X L)
      (if (null L)
          nil
          (if (eq X (car L))
              (h X (cdr L))
              (cons (car L) (h X (cdr L)))
          )
      )
)
```

[4 marks] Show what will be returned.

```
  (h 'a '(a b a c))            Your answer: ______________________

  (h '(a b) '(a (a b) b d))    Your answer: ______________________
```

Then, consider two more definitions

```
(defun f (L)
      (g L nil nil)
)

(defun g (L S M)
    (cond ((null L) (cons S (cons M nil)))
          ((member (car L) (cdr L))
                      (g (h (car L) (cdr L))
                         S
                         (cons (car L) M)))
          (t (g (cdr L)
                (cons (car L) S)
                M)
          )
     )
)
```

[8 marks] Show the results of executing the following expressions.

```
  (g '(a b a) nil nil)              Your answer: ______________________

  (g '(a b b c) '(1 2) '(3 4))      Your answer: ______________________

  (f '(a b b c a d))                Your answer: ______________________

  (f '(a b (b c) a d))              Your answer: ______________________
```

4.2 (8 marks) Recall the problem of generating all prefixes of a list that you did earlier in this exam in Prolog. Now we want to solve the same problem in Lisp. Since in Lisp we cannot generate one solution at a time, we have to generate all solutions and put them into a lisp. In this question you are to define a Lisp function

```
(defun prefix (L) ...)
```

which generates a list of all prefixes of L. The order in which these prefixes are generated is not important. For example,

```
(prefix '(1 2 3)) ==> ((1 2 3) (1 2) (1) ())
```

5. (8 marks) Consider lambda calculus.

[2 marks] Which *order* of reduction can guarantee termination whenever a normal form exists?

[2 marks] What would be the consequence if the *Church-Rosser property* did not hold?

[4 marks] For the λ-expressions below, indicate whether they can be converted to a common λ-expression (which need not be a normal form). Show the details of your work that lead to your conclusion. In each step, clearly indicate whether it is an alpha-reduction ($\rightarrow_\alpha$) or a beta-reduction ($\rightarrow_\beta$).

$(\lambda x \mid (\lambda y \mid x(yy))(\lambda x \mid xx))\ (\lambda x \mid x)$ $\qquad$ $(\lambda x \mid xx)\ ((\lambda x \mid x)(\lambda x \mid xx))$