Comp 210
Midterm Exam #1
1998 Spring

Due 98.Feb.23 (Mon), 10:00 in class or DH3102.

Name: ________________________________

This is a closed-book, closed-computer **two hour** take-home exam. Read each problem carefully. If you need to make any reasonable assumption, just state them in the margin. Clarity and conciseness both contribute toward gettting full points. Please refrain from discussing the exam until after the due-date. You do not need to turn in any scratch-paper. *All submitted sheets must be stapled!*

Time started:____________

Time finished:____________

Problem 1	25	
Problem 2	25	
Problem 3	30	
Problem 4	20	
Total	100	

Pledge:

1. (25 pts)

   Write the function *append2*, which takes two <List-of-Symbol>s and returns the contents of the first followed by the contents of the second. Hand-evaluate `(append2 (list 'cyan 'teal) (list 'ochre))`, showing each recursive call; the answer should be `(list 'cyan 'teal 'ochre)` (If two lines denote equal values, connect them with "="; don't write down a series of unrelated values.)
   Hint: recur only on the first argument, never varying the second argument.

   For this problem, you don't need to write the data-definition and template for a <List-of-Symbol>, or a <List-of-List-of-Symbol>, though if you are stuck these might help you, and they can provide partial credit.

   You of course cannot appeal to the built-in function *append*.

2. (25pts)

Recall that a <NatNum>is

- 0, or
- `(add1` m`)`, where m is a <NatNum>.

(a) Write the template for a function which handles two arguments: a <List-of-Symbol> *and* a <NatNum>. Make sure all cases are covered.

(b) Write the function *nth-rest*, which takes a <NatNum> n and a <List-of-Symbol> los, and returns los without its first n items, or the symbol `'list-too-short` if los has fewer than n items.

```
  (nth-rest 2 (list 'four 'scores 'and 'twenty))
= (list 'and 'twenty)
```

You may simplify and combine cases if you like, but be careful! Indicate whether or not you can rearrange your `cond`'s question/answer pairs and still produce the correct answer.

3. (30pts)

(a) Write a data description for a <List-of-Doors> (where <Door>s will be defined below).

(b) Write a template for a <List-of-Doors>-handling function, assuming you will have a <Door>-handling function.

(c) There are three types of <Door>s, depending on what's on the other side: Either a room (which has a name, but nothing else), or a hallway (which has a name and a <List-of-Doors>), or the outside (which has no further information). Write a reasonable data description for a <Door>, along with any accompanying `define-struct`s.

(d) Write a template for a <Door>-handling function; remember to write in any appropriate recursive calls.

(e) Write the function *can-escape?*, which takes a <Door> and indicates whether it is possible to reach the outside. (Include the contract for each function you write.)

4. (20pts) Write the function *max*, which takes a *non-empty* list of numbers, and returns the largest number in the list. **Use an accumulator**, and do not use `local`. Hand-evaluate `(max (list 7 5 9 2))`, showing each recursive call.

   Be sure to include a contract and description for all functions you write. (You don't need to write any data description for this problem, though you are welcome to.)