Homework should be submitted in my office (4115 AVW) BEFORE 9:20AM (slip them under my door if it is closed). Homework will not be accepted after 9:20AM as the homework problems may be reviewed in class that day. No late homework is accepted. **Late homework will receive no credit.** Be sure to label the problem you are solving clearly with the problem number and subsection. Typing your homework is not required, but homework should be legible. **Illegible solutions will receive no credit.** Be sure to put your **name** on the homework.

Remember, you must do this homework **on your own**, without any external sources. If you are unsure what is allowed, please contact the instructor.

1. **Creating Grammars**

   Write **unambiguous** grammars for the following languages:
   
   (a) \{a^nb^n|n \geq 0 \text{ and } n \text{ is odd}\}
   
   (b) All valid OCaml lists created using ::. Make sure your construction correctly shows the right association of this operator. Use \(v'(a)\) to indicate a value of type \(\prime a\). Examples of valid strings in this language: \(1 :: [], [], “hi” :: “there” :: []\)
   
   (c) All valid OCaml guards for if statements in the abbreviated language containing terminals:
   
   \((,)||,\&, <, >, =.\) Parenthesis must be appropriately matched with every opening left parenthesis followed by a closing right parenthesis. ( and ) have the highest precedence, followed by \&\& and || and the lowest precedence is given to \(<, >, =\) etc. Use \(v'(a)\) to indicate a value of type \(\prime a\). Note: You don’t need to worry about type compatibility. Assume that this is done separately.

2. **Ambiguity Proofs**

   Prove that the following grammar is **ambiguous**. Note: In class, we did not construct formal “proofs” for unambiguous grammars; however, for ambiguous grammars, we used example strings and the definition of ambiguity.
3. OCaml Types

(a) What is the type of the following OCaml function? Explain why this type is correct. (Yes, you can get the answer to the first part by typing this into a computer... but you will need to know how to do this without a computer for the exam, so think about it anyway.)

```OCaml
define func (f, l1, l2) = match l1 with
| []   -> []
| (h1::t1) -> match l2 with
  | []   -> [f h1]
  | (h2::t2) -> [f h1; f h2]
```

(b) Make a function that has the following type:

```OCaml
func : ('a -> 'b) * ('c * 'c -> 'a) * 'c -> 'b
```

You may check your answer using your computer, but make sure you can do it without this aide.