**Programming Languages [Fall 2010]**

**Practice Test III**

**NAME: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Instructions:**

1) This test is 4 pages in length.

2) You have 2 hours to complete and turn in this test.

3) Short-answer questions include a guideline for how many sentences to write. Respond in complete English sentences.

4) This test is closed books, notes, papers, friends, neighbors, etc.

5) Use the backs of pages in this test packet for scratch work. If you write more than a final answer in the area next to a question, circle your final answer.

6) Write and sign the following:

“I pledge my Honor that I have not cheated, and will not cheat, on this test.”

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

 Signed: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_1. [10 points]

What is a programming language? [1-2 sentences]

2. [10 points]

What is a thunk? [1 sentence]

3. [10 points]

If the following is a well-typed ML program, what is its type? If not, why not?

 fun f(x) = f(x);

4. [10 points]

Implement the standard map function in ML. Your solution must (1) be only one line of code, (2) use ML syntax, (3) use one of SML/NJ’s built-in *fold* functions (but use no other built-in functions), and (4) begin as follows:

fun map F L =

5. [20 points]

Define the dynamic semantics of λUT using evaluation contexts.

6. [20 points]

Assuming that progress and preservation theorems hold for λST, prove the following standard type-safety theorem:

e1,e2,τ : (e1:τ  e1 \* e2) e2:τ  ( (v: e2=v)  (e3: e2 e3) ) )

7. [20 points]

Define the first-order abstract syntax of diML, with all the extensions we’ve formalized in class, like aggregate data types, references, exceptions, etc. (You should also be able to define the higher-order abstract syntax and static and dynamic semantics of the fully extended version of diML. Given enough time, you should also be able to prove its soundness. ☺)