Programming Languages [Fall 2016]
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 sentence]

2. [20 points]
a) Implement *map* in terms of *foldr* (without using side effects), or if it can't be done, briefly explain why not.

```
fun map F L =
```

b) Now implement *foldr* in terms of *map* (without using side effects), or if it can't be done, briefly explain why not.

```
fun foldr F v L =
```

3. [20 points]
Define the dynamic semantics of CBV, CBN, and Full-β λ_{UT} using evaluation contexts.

4. [10 points]
Encode NAND expressions into λ_{UT}, assuming standard encodings of Booleans.

5. [20 points]

Assuming that progress and preservation theorems hold for λ_{ST}, prove the following standard type-safety theorem:

$$\forall e_1, e_2, \tau : (e_1 : \tau \ \wedge \ e_1 \rightarrow^* e_2) \ \Rightarrow \ (e_2 : \tau \ \wedge \ ((\exists v : e_2 = v) \vee (\exists e_3 : e_2 \rightarrow e_3)))$$

6. [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. ☺)