Fall, 2011 Name: ____________________

COP 4020 — Programming Languages I

Test on the Declarative Model

Special Directions for this Test

This test has 12 questions and pages numbered 1 through 7.
This test is open book and notes, but no electronics.
If you need more space, use the back of a page. Note when you do that on the front.
Before you begin, please take a moment to look over the entire test so that you can budget your time.
Clarity is important; if your programs are sloppy and hard to read, you may lose some points. Correct syntax also makes a difference for programming questions.
When you write Oz code on this test, you may use anything we have seen in chapters 1–2 of our textbook. But unless specifically directed, you should not use imperative features (such as cells) or the library functions `IsDet` and `IsFree`. Problems relating to the kernel syntax can only use features of the kernel language (given in Tables 2.1 and 2.2 of the textbook).
You are encouraged to define functions or procedures not specifically asked for if they are useful to your programming; however, if they are not in the Oz base environment, then you must write them into your test.

For Grading

Question:	1	2	3	4	5	6	7	8	9	10	11	12	Total
Points:	5	5	5	5	20	10	5	5	10	15	5	10	100
Score:													

1. (5 points) Which of the following best describes the semantics of Oz's dataflow variables? (Circle the correct answer.)
   - A. A dataflow variable starts with an arbitrary value (whatever happens to be in memory where the space for it is allocated).
   - B. A dataflow variable can only be assigned once, and a thread that tries to read it before the variable has a determined value will be suspended until the variable's value is determined.
   - C. A dataflow variable can be assigned any number of times, because that is the way variables work in all programming languages.
   - D. It is illegal for a thread to attempt to read from a dataflow variable that does not yet have a determined value.
   - E. Reading from a dataflow variable whose value is not yet determined throws an exception (an "unasssigned" exception).
2. (5 points) [Concepts] Which of the following is most convenient for programming? (Circle the correct answer.)
   - A. A programming language with lots of syntactic sugars.
   - B. A kernel language that has no syntactic sugars.
   - C. A programming language that is made to taste like ice cream.
   - D. A programming language that has lots of fiber and vitamins.
3. (5 points) [Concepts] Which of the following is correct? (Circle the correct answer.)
   - A. When programming, you should never use syntactic sugars, since they will confuse anyone who is not familiar with the language.
   - B. When programming, you should always desugar complex statements so that you can be paid for more lines of code.
   - C. When programming, you should always use syntactic sugars, so that your code is shorter and easier to read.
   - D. After programming, you should always brush your teeth, so that they are not decayed by the use of syntactic sugars.
4. (5 points) [Concepts]

   Suppose you are editing some code in your favorite programming language and you want to change all occurrences of a variable identifier's name from that name to some other name. If this edit is to be successful (leaving the meaning of the program unchanged), should the variable identifier you are changing be free or bound in the code you are editing? (a) Answer "free" or "bound" and (b) briefly explain why.

5. [Concepts] Consider the following Oz code.

```
local Res in
   local X in
      X = 99
      local Rec in
         Rec = agent(name: smart number: 86 phone: shoe)
         case Rec of
            agent(name: N number: X phone: P) then Res = X
         else Res = X
         end
         {Browse Res}
      end
   end
end
```

(a) (5 points) What value, if any is shown in the browser when this code executes?

(b) (5 points) Does the variable identifier `Res` occur free or bound in this code? (Answer "free," "bound," or "neither".)

(c) (5 points) Does the variable identifier `Browse` occur free or bound in this code? (Answer "free," "bound," or "neither".)

(d) (5 points) Does the variable identifier `N` occur free or bound in this code? (Answer "free," "bound," or "neither".)

6. (10 points) [Concepts]

   Consider the execution of the following Oz code.

```
local Res in
   local M in
      M = 5
      M = M * 20
      M = 2 * M
      Res = M
      {Browse Res}
   end
end
```

   Circle the letter that best describes what happens during the above code's execution.

   A. The browser displays `200`, because 200 is $5 \times 20 \times 2$.

   B. The browser displays `5`, because `M` can only be assigned once.

   C. Execution of the code suspends, because `Res` is not determined.

   D. This produces a failure (a "tell" exception is thrown).

   E. This produces a failure, and then later the browser displays `200`.

   F. The code does not compile, due to static scoping.

7. (5 points) [Concepts] Consider the following Oz code.

```
local Res in
   local N in
      N = anatom
      local P in
         thread
            local N in
               N = 3
               P = proc {$ X R} {Number.'+' N X R} end
            end
         end
         thread
            local Seven in
               Seven = 7
               {P Seven Res}
               {Browse Res}
            end
         end
      end
   end
end
```

Circle the letter of the true statement below.

A. When this code executes, it suspends, because when it runs the call `{P Seven Res}` neither `P` nor `Res` has a determined value.

B. When this code executes, it shows 10 in the browser, and no other result is possible, because the second thread waits until `P` has a determined value, and because Oz makes closures for **`proc`** values.

C. When this code executes, it has an exception, because `N` denotes the atom `anatom` and so `Number.'+'` throws an exception when trying to add `anatom` and 7.

D. When this code executes, it might show either show 10 in the browser or throw an exception, depending on which thread finishes before the other. With concurrent programs like this, it is impossible to predict the outcome.

The next two problems ask for sets of free or bound variable identifiers that occur in a program written in the Oz declarative kernel language. For these problems, write the entire requested set in brackets. For example, write $\{V, W\}$, or if the requested set is empty, write $\{\}$.

Also, recall that **declare** is *not* in the declarative kernel, so you should *not* imagine an implicit **declare** in the examples given for these problems.

8. Consider the following Oz statement in the declarative kernel language.

```
local Res in
   local X in
      local Y in
         {F X Y T Res}
      end
   end
end
```

   (a) (2 points) [Concepts] Write the entire set of the variable identifiers that occur free in the statement above.

   (b) (3 points) [Concepts] Write the entire set of the variable identifiers that occur bound in the statement above.

9. Consider the following statement in the declarative kernel language.

```
K = proc {$ X ?R}
       R = proc {$ Y ?Res}
              Res = X
           end
    end
local One in
   local Two in
      One = single
      local F in
         {K One F}
      end
   end
end
```

   (a) (2 points) [Concepts] Write the entire set of the variable identifiers that occur free in the statement above.

   (b) (8 points) [Concepts] Write the entire set of the variable identifiers that occur bound in the statement above.

10. (15 points) [Concepts] Desugar the following code into the syntax of the declarative kernel. (Assume that the free identifiers `Half` and `Times` denote functions.)

```
fun {Area B H}
   {Half {Times B H}}
end
```

11. (5 points) [Concepts] Does a closure for a procedure (or function) need to remember the values of the free variable identifiers or the bound variable identifiers that occur in its body? (a) Answer "free" or "bound" and (b) give a brief explanation for your answer.

12. (10 points) [Concepts] [UseModels] Consider the following Oz code.

```
local Res in
   local K in
      K = 6
      local Record in
         Record = clarinet(make: yamaha key: bflat owner: marie)
         case Record of
            clarinet(key: K make: M owner: _) then {Browse M|K|clarinet|nil}
         else
            {Browse nil}
         end
      end
   end
end
```

What happens when this code is executed? Briefly explain what happens, including what, if anything, is shown in the Browser.