CMSC330 Spring 2010 Final Exam

Name ________________________________

Do not start this exam until you are told to do so!

Instructions

- You have 120 minutes for to take this exam.
- This exam has a total of 130 points. An average of 1 minute per point.
- This is a closed book exam. No notes or other aids are allowed.
- If you have a question, please raise your hand and wait for the instructor.
- Answer essay questions concisely using 1-2 sentences. Longer answers are not necessary and a penalty may be applied.
- In order to be eligible for partial credit, show all of your work and clearly indicate your answers.
- Write neatly. Credit cannot be given for illegible answers.

	Problem	Score	Max Score
1	Programming languages		8
2	Ruby		6
3	Regular expressions & finite automata		18
4	OCaml types & type inference		6
5	OCaml programming		6
6	OCaml polymorphic types & higher order functions		15
7	Context free grammars & parsing		16
8	Operational semantics		8
9	Lambda calculus		6
10	Lambda calculus encodings		9
11	Garbage collection		8
12	Polymorphism		8
13	Java multithreading		16
	Total		130

1. (8 pts) Programming languages
    a. (4 pts) Briefly define a weak type system. Provide a code fragment example.

    b. (4 pts) Recall that functional languages allow functions to be passed as arguments; e.g., map takes a function as an argument and applies it to elements of a list. Java does not directly support passing functions/methods as arguments. Briefly describe how you can encode function-passing in Java to implement something like map. (1 sentence should be sufficient).

2. (6 pts) Ruby
    a. (4 pts) Name an important difference between Ruby's *nil* and Java's *null*.

    b. (2 pts) What is the output (if any) of the following Ruby program? Write FAIL if code does not execute.

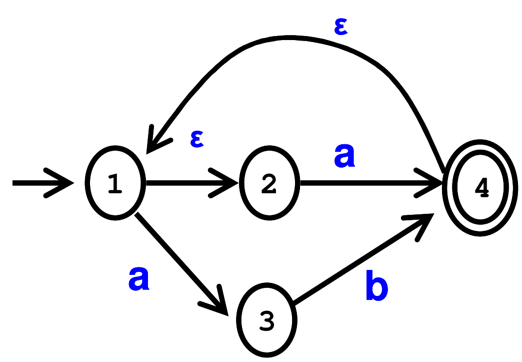
```
a = { }                    # Output =
a[1] = "foo"
puts a[0]
```

3. (18 pts) Regular expressions & finite automata

   a. (5 pts) Give a DFA that is equivalent to the regular expression a*b*c.

   b. (3 pts) Give a regular expression (formal REs or in Ruby) that accepts all 3-digit binary numbers. If you choose to use Ruby, you may not use the {3} feature.

   c. (10 pts) Convert the following NFA to a DFA using the subset construction algorithm. Be sure to label each state in the DFA with the corresponding state(s) in the NFA.



4. (6 pts) OCaml Types and Type Inference
   a. (2 pts) Write an OCaml expression with the following type
      int -> int -> int

   b. (4 pts) Give the type of the following OCaml expression
      let f x y = y (x+1)

5. (6 pts) OCaml programming

   Write a function *convert* that takes as its argument a function of type
   (‘a -> (‘b -> ‘c)) and returns a function of type (‘a * ‘b) -> ‘c.

   Example:

   ```
   let add x y = x + y ;;
   let addPair = (convert add) ;;
   addPair (1,2) = 3
   ```

6. (15 pts) OCaml polymorphic types & higher-order functions

Consider the OCaml type *shape* implementing a square and rectangular shapes:

```
type shape =
  Square of int             (* height *)
| Rectangle of int * int   (* height & width *)
```

a. (5 pts) Write a function *area* of type (shape -> int) that takes a shape and returns its area. Recall the area of a square = height^2, rectangle = width*height

Examples:

```
area (Square 2) = 4
area (Rectangle (3,5)) = 15
```

b. (10 pts) Write a function *totalArea* that, given a list of shapes, produces the total area of all of the shapes. Your function should not use any recursion, but instead should use fold and/or map in combination with *area* and anonymous functions. Solutions using recursion and/or helper functions may receive partial credit.

Examples:

```
totalArea [Square 2] = 4
totalArea [Square 2 ; Rectangle (3,5)] = 19
totalArea [ ] = 0
```

```
let rec map f lst =  match lst with
  [] -> []
  | h::t -> (f h)::(map f t)
```

```
let rec fold f a lst = match lst with
   [] -> a
   | (h::t) -> fold f (f a h) t
```

7. (16 pts) Context free grammars & parsing

Consider the following grammar (where S = start symbol and terminals = **[**, **]**, **;**, **e**):

$S \rightarrow$ **[**A**]** | *epsilon*

$A \rightarrow$ S **;** A | **e**

a. (3 pts each) Indicate whether the following strings are generated by this grammar

i.	**[e;e]**	Yes	No	(circle one)
ii.	**[[e];;e]**	Yes	No	(circle one)

b. (4 pts) Compute First sets for S and A

c. (6 pts) Write the function parse_A( ) used in a predictive, recursive descent parser for the grammar. You may assume parse_S( ), match( ) and the variable *lookahead* is provided.

8. (8 pts) Operational semantics

In an empty environment, to what value v will the expression (fun z = + z z) 6 evaluate to? In other words, find a v such that you can prove the following:

- • ;(fun z = + z z) 6 $\rightarrow$ v

Use the operational semantics rules given in class, included here for your reference. Show the complete proof that stacks uses of these rules.

Number

$$\frac{}{\bullet;\ \mathrm{n} \rightarrow n}$$

Lambda

$$\frac{}{\mathrm{A};\ \mathrm{fun}\ \mathrm{x} = \mathrm{E} \rightarrow (A,\ \lambda x.E)}$$

Addition

$$\frac{\mathrm{A};\ \mathrm{E}_1 \rightarrow n \qquad \mathrm{A};\ \mathrm{E}_2 \rightarrow m}{\mathrm{A};\ +\ \mathrm{E}_1\ \mathrm{E}_2 \rightarrow n + m}$$

Function application

$$\frac{\mathrm{A};\ \mathrm{E}_1 \rightarrow (A',\ \lambda x.E) \qquad \mathrm{A};\ \mathrm{E}_2 \rightarrow v \qquad \mathrm{A},\ \mathrm{A}',\ \mathrm{x}:v;\ \mathrm{E} \rightarrow v'}{\mathrm{A};\ (\mathrm{E}_1\ \ \mathrm{E}_2) \rightarrow v'}$$

Identifier

$$\frac{}{\mathrm{A};\ \mathrm{x} \rightarrow A(x)}$$

9. (6 pts) Lambda calculus

   (3 pts each) Evaluate the following λ-expressions as much as possible

   a. $(\lambda x.\lambda y.x\ y)\ (\lambda z.z)\ a$

   b. $(\lambda z.\lambda y.z\ y\ z)\ y$

10. (9 pts) Lambda calculus encodings

   a. (3 pts) Consider the function $D = \lambda x.x\ x$. What happens when you apply D to itself ?

   b. (6 pts) Consider the standard encodings for the booleans *true* and *false*. What does the following function Q compute? Hint: try passing two booleans for a & b, and see what you get from evaluating the function (if you do this, show your work clearly for partial credit).

$$Q = \lambda a.\lambda b.\lambda x.\lambda y.a\ x\ (b\ x\ y).$$

true	$= \lambda x.\lambda y.x$
false	$= \lambda x.\lambda y.y$

(8 pts) Garbage collection

Consider the following Java code.

```
class Avatar {
  static Navi Tom, Norm, Jake;
  private void blockBuster( ) {
    Tom = new Navi(1);      // object 1
    Norm = new Navi(2);     // object 2
    Jake = new Navi(3);     // object 3
    Jake = Tom;
}
```

c. (4 pts) What object(s) are garbage when blockBuster ( ) returns? Explain.

d. (4 pts) Briefly describe the difference between reachability and liveness in the context of garbage collection.

11. (8 pts) Polymorphism

Consider the following Java classes:

```
class A { public void a( ) { … } }
class B extends A { public void b( ) { … } }
class C extends B { public void c( ) { … }}
```

(4 pts each) Explain why the following code is or is not legal

a. int count(Set<? extends A> s) { … } … count(new TreeSet<C>());

b. int count(Set<? extends A> s) { for (C x : s) x.c( ); … }

12. (16 pts) Java multithreading

For this problem you may use either Java 1.4 or 1.5 synchronization.

Consider the following code for implementing a 1-place buffer:

```
public class Container {
  private Object buf = null;
  public void put(Object x) throws FullException {
    if (buf != null) throw new FullException();
    buf = x;
  }
```

```
  public Object get() throws EmptyException {
    if (buf == null) throw new EmptyException();
      Object x = buf;
      buf = null;
      return x;
  }
}
```

a. (3 pts) This code is only appropriate for single threaded applications. Give an example of what could go wrong if two threads use a Container implemented with this code at the same time.

b. (3 pts) Add synchronization so that multiple threads may use this code at the same time. The semantics should be the same: if a thread tries to remove something from the buffer, but the buffer is empty, it will throw an exception. Likewise if the buffer is full, and a thread attempts to put something in, it will throw an exception, just as with the code now.

c. (10 pts) Add two methods to your amended Container class that implement variants of get() and put(), called take() and store(), which act as follows: if a thread calls take() the thread should return the contents of the buffer or wait until something is put there by another thread, and return that. Conversely, if a thread calls store() but something is already there, then it should wait until the buffer is empty. The methods take() and store() do not need to throw any exceptions.

Some helpful functions:

Object o;	// any Object	[Java 1.4]
o.wait()	// sleeps until signaled	
o.notifyAll()	// wakes up all threads sleeping on object	

m = new ReentrantLock()	// returns new lock	[Java 1.5]
c = m.newCondition( )	// returns conditional variable for lock	
m.lock( )	// acquires lock, block if another thread holds lock	
m.unlock( )	// releases lock	
c.await( )	// sleeps until notified	
c.signalAll( )	// wakes up all threads sleeping on condition var	