CSE 341 - Programming Languages
Final exam - Autumn 2008

Your Name:

Open book and notes. No laptop computers, PDAs, or similar devices. (Calculators are OK, although you won't need one.) Please answer the problems on the exam paper — if you need extra space use the back of a page.

90 points total

1. (10 points) Suppose we have the following definition of the `map` function in Haskell.

```
map f [] = []
map f (x:xs) = f x : map f xs
```

Circle each type declaration that is a correct type for `map`. (Not necessarily the most general type, just a correct one.)[1]

```
map :: ([Integer] -> [Integer]) -> [[Integer]] -> [[Integer]]

map :: (a -> a) -> [a] -> [a]

map :: (a -> b) -> [b] -> [a]

map :: (a -> b) -> [a] -> [b]
```

Which of the above types, if any, is the most general type for `map`?

[1] Oh no, I hear you cry! Not *this* question again . . .

2. (6 points) Consider the following Scheme expression:

```
(let* ((x a)
       (y (+ x 10)))
  (+ x y))
```

Write an equivalent Scheme expression that binds `x` and `y` to the same values as above and returns the value of `(+ x y)` but that only uses `let` and not `let*`.

3. (10 points) Write a Ruby class `Delay` that implements delays (as we saw in Scheme). The following code shows how it should work:

```
n = 0
d = Delay.new {n=n+1; 3+4}
d.force
d.force
v = d.force
e = Delay.new {1/0}
```

After we evaluate these statements `v` should be 7, but `n` should only be 1 (since we only evaluate the block once). Further, since we never force `e`, we shouldn't get a divide-by-zero error.

4. (10 points)

(a) Write a `repeat` rule in CLP(R) with three arguments: `N`, `X`, and `Xs`. It succeeds if `Xs` is a list consisting of `N` copies of `X`. Here are some sample goals and results:

```
?- repeat(3,a,Xs).
   Xs = [a,a,a]

?- repeat(0,a,Xs).
   Xs = []

?- repeat(-2,a,Xs).
   no.
```

(b) What answers are returned for the following goals using the rule you just defined for Question 4a? (Part of answering Question 4b is deciding what a reasonable behavior is in these situations.) Continue on the back of this page as needed. If there are a finite number of answers give them all; if an infinite number give at least the first three.

```
?- repeat(N,Q,[a,a,a,a]).
```

```
?- repeat(N,Q,[a,b,c]).
```

```
?- repeat(N,a,As).
```

5. (12 points) Consider the following CLP(R) rule:

```
sum([],0).
sum([X|Xs],X+S) :- sum(Xs,S).
```

(a) Draw the simplified derivation tree for the goal `sum([100], N).`

(b) Draw the simplified derivation tree for the goal `sum(As, 10).` (This is an infinite tree; include at least 3 answers in the tree that you draw.)

Again, continue on the back of the page if necessary.

6. (10 points) This question use the following hierarchy of Java classes for simple geometric shapes. (This is the same hierarchy you used for the last homework assignment.)

```
public interface GeometricShape {
    public void describe();
}

public interface TwoDShape extends GeometricShape {
    public double area();
}

public interface ThreeDShape extends GeometricShape {
    public double volume();
}

public class Circle implements TwoDShape  {
    ....
}

public class Cone implements ThreeDShape  {
    ....
}

public class Rectangle implements TwoDShape  {
    ....
}


public class Sphere implements ThreeDShape  {
    ....
}
```

Suppose we have three different implementations of a `total_area` method to find the total area of some shapes.

```
public static double total_area1(ArrayList<TwoDShape> shapes)
{
    double sum = 0;
    for(TwoDShape shape : shapes) {
        sum = sum + shape.area();
    }
    return sum;
}
```

```
public static double total_area2(
    ArrayList<? extends GeometricShape> shapes)
{
    double sum = 0;
    for(GeometricShape shape : shapes) {
        sum = sum + ((TwoDShape) shape).area();
    }
    return sum;
}

public static double total_area3(
    ArrayList<? extends TwoDShape> shapes)
{
    double sum = 0;
    for(TwoDShape shape : shapes) {
        sum = sum + shape.area();
    }
    return sum;
}
```

Now suppose that we have four variables declared as follows:

```
ArrayList<Rectangle> rects;
ArrayList<TwoDShape> two_d_shapes;
ArrayList<GeometricShape> geo_shapes;
double area;
```

We then assign various shapes to the different arraylists. (So `rects` will only contain instances of `Rectangle`, while `geo_shapes` might contain instances of `Circle`, `Cone`, `Rectangle`, or `Sphere`.)

Consider the following code that uses these methods and variables. Write "compiles" next to the following statements that compile correctly, and write "doesn't compile" next to the ones that don't compile. Finally, if there is a possibility that the statement will cause a cast exception to be raised, write "possible exception" next to it.

```
area = total_area1(rects);
area = total_area1(two_d_shapes);
area = total_area1(geo_shapes);

area = total_area2(rects);
area = total_area2(two_d_shapes);
area = total_area2(geo_shapes);

area = total_area3(rects);
area = total_area3(two_d_shapes);
area = total_area3(geo_shapes);
```

7. (6 points) What is something that Ruby mixins can do that Java interfaces can't?

8. (6 points) Consider the following Java code fragments. In each case, does the code compile correctly? If so, does it execute without error, or is there an exception?

   (a)
   ```
   Object[] a = new String[10];
   a[0] = "hi there";
   ```

   (b)
   ```
   ArrayList<Object> a = new ArrayList<String>();
   a.add("hi there");
   ```

   (c)
   ```
   Object[] a = new String[10];
   a[0] = new Point(10,20);
   ```

9. (10 points) What is duck typing? What is an advantage of duck typing over a more standard approach, and what is a potential problem? Finally, could you program in Java in a way that follows the duck typing philosophy? Why or why not?

10. (10 points) In Java, a class can be covariant in the return type of an inherited method. It turns out that this feature is new in J2SE 5.0; before that, the return type had to be invariant (i.e., exactly the same as the return type in the method in the superclass or interface).

    For a widely used language like Java, the language designers must concern themselves with compatability issues. What compatability issues were raised by this change? Are there programs that used to compile without error in pre-5.0 Java that now get a compile error? Are there programs that formerly wouldn't compile that now do compile? What about runtime behavior? Are there programs that compiled in old and new Java but that can behave differently? If there are such programs with different compile time or run time behavior, give an example, and explain why there is a difference.