CSE 341 — Ruby Discussion Questions

1. What do the following Ruby expressions do?

```
x+2
octopus.swim("fast")
octopus.swim "fast"
octopus.tentacles = 8
Aquarium.new("clownfish")

["clown", "fish"].each {|s| puts s}
[1,2,3].map { |j| j*10}

sum=0
4.times {sum=sum+10}
```

2. Write a Ruby class `Book`, which has fields for title and author. When you create a new instance of book you should give values for those fields. Also define getters (but not setters) for them. Finally, write a statement that makes a new instance of `Book` with a suitable author and title.

3. Write a class `Delay` that implements delays (like the delay function 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. Write a `min` method for the `Enumerable` mixin. You'll need to decide how to handle finding the minimum of an empty collection. Bonus points for handling this in the same way Ruby itself does!

   Hint: look at the implementation of `map` at the end of the `inheritance.rb` handout.

5. Consider the class and module definitions in `self_super.rb` linked from the 341 Ruby web page. Suppose we define a class `C6` as follows:

```
class C6 < C1
  include M2
end
```

   What is the result of evaluating these expressions?

```
x = C6.new
x.test1
x.test2
x.kind_of?(C6)
x.kind_of?(M2)
x.kind_of?(M1)
C6.ancestors
C6.superclass
C6.superclass.superclass
C6.superclass.superclass.superclass
C6.superclass.superclass.superclass.superclass
```

6. Write a literal that evaluates to a hash with the keys 1, 2, and 3, and the corresponding values `"sam"`, `"sue"`, and `"fred"`.

7. Suppose `words` is an array of strings, for example

   ```
   words = ["squid", "octopus", "clam", "squid", "squid"]
   ```

   Write one or more Ruby statements that create a hash named `counts`, whose keys are the strings in `words` and whose values are the number of occurrences of that word. So for the example above

   ```
   counts = {"squid"=>3, "octopus"=>1, "clam"=>1}
   ```

8. Suppose the `Octopus`, `Cephalopod`, `Animal` hierarchy from the file `self_super.rb` has been loaded. What is the result of evaluating the following expressions?

   ```
   c = Cephalopod.new
   c.class
   c.class.superclass
   c.class.superclass.superclass
   c.class.superclass.superclass.superclass
   c.class.superclass.superclass.superclass.superclass
   c.class.class
   c.class.class.class
   ```