LAST NAME_____first name_____ Cornell Net id_____

CS1110 Prelim 1 7 Oct 2010

This 90-minute exam has 5 questions (numbered 0..4) worth a total of 100 points. Scan the whole test before starting. Budget your time wisely. Use the back of these pages if you need more space. You may tear the pages apart; we have a stapler at the front of the room.

Question 0 (2 pts). Write your last name, first name, and Cornell NetId, legibly, at the top of each page.

Many of the questions deal with the two classes `Species` and `Animal` shown on the last two pages of this exam.

Question 1 (16 points) Drawing objects. Under each of the new-expressions given below, draw the object that results from evaluation of that new-expression. Do not draw the partitions for class `Object`.

Be sure to fill in the values of fields correctly. Do this based on the specifications of the methods in the classes. Also draw the static variable and make sure you show properly what is in it. Assume these are the first new-expressions to be evaluated. You do not need to draw the file drawers.

If you draw an object of class `Vector`, draw its list of objects in some reasonable way.

```
new Species("Indiana Bat", 1967)     new Animal("Red Wolf", 4)
```

Question 2 (35 points) Writing method bodies. The bodies of most methods in `Species` and `Animal` (on the last two pages of this exam) have been not been written. Write them, following any directions given in notes in the method bodies. Do not write any other methods. Be careful. Note that the purpose of this question is to check your ability to (1) write constructors in classes and subclasses, including calls on other constructors, (2) use **this** and **super**, (3) understand the use of preconditions, (4) know where private fields cannot be used, (5) understand Vectors and calls to methods in Vector objects, and (6) deal appropriately with a static variable.

The bottom of page 5 contains some Vector methods that might come in handy.

LAST NAME_____first name_____ Cornell Net id_____

Question 3 (20 points). String manipulation.

Suppose field `species` of class `Species` on page 4 can have one of two forms:

1. A name with no blanks, e.g. `"Whale"`.
2. A two word name, with two or more blanks between the words, e.g. `"Red  Fox"`.

(a) Make function `fixBlanks`, defined below, static if it should be static.

(b) Write the body of function `fixBlanks`, below, whose purpose is to return a string that is the same as its parameter but with many blanks (if there are many) replaced by a single blank. For example, the call `fixBlanks("Red  Fox")` evaluates to `"Red Fox"`. The table at the bottom of the page contains some more String methods that you can use —though not all of them are useful here.

```
/** = s  but with a sequence of blanks (if present) replaced by one blank.
    Precondition: String s has the form  w1  or  w1 bs w2
    where w1 and w2 are nonempty strings that contain no blanks
    and bs is a sequence of one or more blanks. */
public String fixBlanks(String s) {




}
```

String, Character, and Integer functions (we assume you know `charAt`, `length`, and `substring`)		
Return	**Method**	**Value returned**
int	`s.indexOf(n)`	the index within `s` of the first occurrence of String `n` (`-1` if none)
int	`s.lastIndexOf(n)`	index within `s` of the last occurrence of String `n` (`-1` if none)
`String`	`s.trim()`	a copy of s with beginning and ending spaces removed
int	`Integer.parseInt(s)`	the int that the String s represents; s must contain only digits except that the first character can be a minus sign.
`boolean`	`Character.isDigit(ch)`	"character ch is a digit, e.g. '0' "

Question 4 (27 points) Miscellaneous topics.

(a) **3 pts.** Field `endangered` in `Species` is **public**. Write an expression that could appear in any other class and that would evaluate to the number of `Species` objects in Vector `endangered` —without having to reference an object of class `Species`.

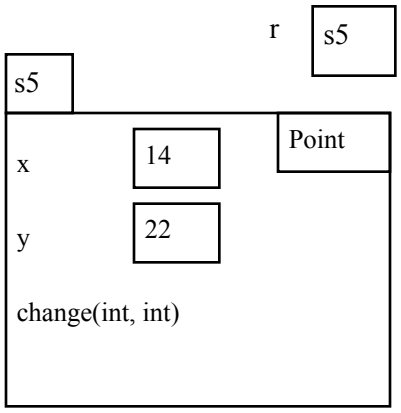
(b) **11 pts.** Write down the four steps in executing a procedure call.

Then, suppose that there is a class Point that has one method, defined as follows:

```
public void change(int a, int b)
      { x=a; y=b;}
```

Using object `s5` and variable `r` that appear to the right, perform (**only**) the first two steps of executing this call:

```
r.change(1,9);
```



(c) **(5 pts)** Consider the following conditional expression, where `isMale` is some pre-defined boolean function and age is some predefined int variable:

```
(isMale() ? age > 75 : age > 80)
```

Write an equivalent boolean expression —i.e. convert this conditional expression into a boolean expression that uses only &&, ||, and !. Do *not* assume that function isFemale() exists.

(d) **(8 pts)** What four kinds of variables can occur in a Java program, and where are they declared?

```
import java.util.*;

/** An instance is a species, perhaps on the endangered list. */
public class Species {
   private String species= "";    // Name of this species

   private int year= 0; // Year put on endangered list: >= 1900 (0 if not on the list)

   /** all species on the endangered list */
   public static Vector<Species> endangered= new Vector<Species>();

   /** Constructor: an instance for species species that was put on the endangered list
                    in year year (0 if not on endangered list) */
   public Species(String species, int y) { // Note: Deal appropriately with the static variable









   }

   /** Constructor: an instance for a species s that is not on endangered list */
   public Species(String s) { // Note: this body must be a single statement




   }

   /** Change this species' year on the endangered list to b.
       Precondition: b is not 0. */
   public void setYear(int b) { // Note: Don't forget to deal with the static variable









   }

   /** = description of this species */
   public String toString() {  // Note: We have completed this method. Don't do anything with it
      return species + (year == 0 ? "" :" endangered since " + year);
   }
```

```
/** An instance is a species of Animal */
public class Animal extends Species {
   private int legs; // number of legs this animal has

   /** Constructor: an n-legged Animal of species s that is
       not endangered */
   public Animal(String s, int n) { // body must contain exactly
                                    // two statements









   }

   /** = number of legs this animal has */
   public int legs() {




   }


  /** = description of this animal, in the form given by the
         toString function  in the superclass followed by:
         " with " <legs> " legs" */
   public String toString() {




   }
}
```

0 _____	out of 02
1 _____	out of 16
2 _____	out of 35
3 _____	out of 20
4 _____	out of 27
Total _____	out of 100

Vector methods		
	`Vector()`	Constructor for an empty Vector —no objects in it
void	`v.add(p)`	Append object p to Vector v's list of objects
int	`v.size()`	The length of Vector v's list of objects
Object	`v.get(i)`	Return the object at position i in v