NAME _____________________________________ Cornell Net id __________

CS1110 Prelim 1 25 February 2010

This 90-minute exam has 6 questions (numbered 0..5) 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 can tear the pages apart; we have a stapler at the front of the room.

Question 0 (2 points): Write your name and NetID, legibly, at the top of each page.

Question 1 (20 points). Use the back of the next page to answer this question.

(a) 8 pts. At the bottom of the page are two class definitions. Draw one folder (object, instance) of each. Do not show the partition for superclass `Object`.

(b) 12 pts. Write a class definition for a class `ModelVariation` that

1. Is a subclass of `Model`;
2. Has suitable specifications on methods and definitions on fields;
3. Has a double field `price`, which is the price of the variation of the model (for example, it might be barebones, or it might be deluxe and have GPS built in);
4. Has a constructor with the auto maker, the model, and the price as parameters;
5. Overrides function `toString()` to return a complete representation of the instance similar to the way function `toString` in class `Model` does.

```
/** An instance is a car manufacturer*/
public class Automaker {
   // automaker's name, e.g. Toyota
   private String name= null;

   /** Constructor: automaker named s*/
   public Automaker(String s) {
        name= s;
    }

   /** Constructor: automaker with
             unknown name "" */
   public Automaker() {
        name= "";
    }

   /** = a repr. of this auto maker */
   public String toString() {
        return "Automaker: " + name;
    }
}
```

```
/** A car model made by an automaker
                  (e.g. Corolla */
public class Model extends Automaker {
   /** model */
   private String model;

   /** Constructor: an instance with
      auto maker m and model mo*/
   public Model(String m, String mo) {
      super(m);
      model= mo;
   }

   /** Set the model to m */
   public void setModel(String m) {
       model=  m;
   }

   /** = a repr. of this instance,
      giving automaker and model */
   public String toString() {
      return super.toString() + " " +
            model;
   }
}
```

Question 2 (20 pts).

(a) Suppose we want to keep track of the number of objects of class `Automaker` that have been created. Write the declaration of the variable that will contain this value (this variable will be defined in class `Automaker`). Then, rewrite the constructors in class `Automaker` to maintain this variable. (Don't scribble over the constructors on the previous page; just rewrite them here:)

(b) 6 pts. Below, write the body of another constructor for subclass `Model`. The specification is given. The body of this constructor must be a single statement.

```
/** Constructor: a model made by automaker m but with "" for the
                 model name, since the name is not yet chosen. */
public Model(String m) {



}
```

(c) 4 pts. Complete the body of the function `getObjectName`, specified below, which is to be declared in class `Automaker`. Remember that `Automaker` is automatically a subclass of class `Object`, the superest class of them all.

```
/** = the name on the tab of this object */
public String getObjectName() {



}
```

Question 3 (20 points). To the right is a definition of class Student. Suppose these variables are declared:

Student p1;
Student p2;
Student p3;

(a) Draw the three variables below, as named boxes.

(b) Now consider these statements:

p1= **new** Student("Bill", "bk12");
p2= **new** Student("Bill", "bk13");
p3= p2;
p3.setName("Jack");

Execute the four statements, using the variables you drew in step (a). You need not draw frames for calls; we do not want to see them. However, you must draw any objects that are created during execution.

```
/** An instance represents a Student*/
public class Student {
  // the student's name
  private String name;

  // the student's netid
  private String netid;

  /** Constructor: a Person with
        name n and netid i*/
  public Student(String n, String i) {
      name= n;
      netid= i;
  }

  /** = the Student's name */
  public String getName() {
      return name;
  }

  /** set the Student's name to n */
  public void setName(String n) {
      name= n;
  }

  /** = "this Student and s have the
         same netid" */
  public boolean equals(Student p) {
      return netid.equals(p.netid);
  }

  /** = a representation of this
        Student*/
  public String toString() {
      return netid + ":" + name;
  }
}
```

Question 4: (20 pts) Write the function below, which for a date like "`February 25, 2010`" produces the string "`2010.2.25`". The table below gives methods of class String that you can use.

Return	**Method**	**Purpose**
`char`	`s.charAt(i)`	= the character at position `i` of `s`
`int`	`s.length()`	= the number of characters in `s`
`int`	`s.indexOf(n)`	= the index within `s` of the first occurrence of String `n` (`-1` if none)
`int`	`s.indexOf(n, k)`	= the index within `s` of the first occurrence of String `n` that begins at or after index `k` (`-1` if none)
`String`	`s.substring(h,k)`	= a String consisting of characters in `s[h..k-1]`, ie. `s[h], s[h+1], …, s[k-1]`
`String`	`s.substring(h)`	= a String consisting of characters `s[h..s.length()-1]`

In addition, assume that function `month(String m)`, given the month as a string `m`, produces the number of the month —e.g. `month("January") = 1`.

/** = date d in the form "year.month.day" . An example is "2009.1.2.

Precondition: d is in the form exemplified by "January 2, 2009". That is, it contains the month, followed by 1 blank, followed by the day, followed by a comma and a blank, followed by the year. */

public static String fix(String d) {

}

Q 0		/02
Q 1		/20
Q 2		/20
Q 3		/20
Q 4		/20
Q 5		/18
Total		/100

Question 5 (18 points): Answer the following questions concisely:

(a) 5 pts. What is an argument? A parameter?

(b) 5 pts. What is a local variable? What is its scope?

(c) 4 pts. Draw a frame for the call `C.p(6, 7+1)` of the following procedure `p`, which is declared in class `C`. We want to see what the frame for the call looks like after the argument values are assigned to the parameters but before the method body is executed.

```
public static void p(double y, int z) {
     int x;
     x= y * z;

}
```

(d) 4 pts. Think carefully about how the call `p(6,7+1)` of part (c) is executed, and write down in which step of this execution local variable `x` is created.