CS1110 Prelim 1 8 Oct 2009

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 `Customer` and `PrefCustomer` 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. Be sure to fill in the values for variables correctly. Do this based on the specifications of the methods in the classes. Do not draw the partitions for class `Object`.

Also draw the static variable and make sure you show properly what is in it. If you draw an object of class `Vector`, draw its list of objects in some reasonable way.

```java
new Customer("Obama", 1961, "o@a.com")  new PrefCust("Skorton", -1, null, "bronze")
```

**Question 2** (30 points) **Writing method bodies.** The bodies of the methods in `Customer` and `PrefCustomer` (on the last two pages of this exam) have been not been written. Write them, following any directions given in comments in the method bodies. Do not write any other methods. Specifications of methods that you may need are in the table at the bottom of the last page of this exam.
Question 3 (25 points). String manipulation.

Suppose that we choose to replace all constructors in class Customer with a single new constructor whose parameter is an email address as input; the constructor extracts values for the fields from that email address.

For the purposes of this problem, assume each email address has one of the two forms indicated in the box to the right.

(a) **22 pts.** Write the body of this new constructor below. Do not call any other constructors (since this problem states that the existing ones are being replaced).

The table at the bottom of the last page of this exam describes functions that you may need. If you are doing any complicated combinations of String operations, you may wish to add comments so that we can tell what you probably meant to do (and thus assign partial credit in the case of small but confusing typos.)

```java
/** Constructor: a customer with email address addr. The last name and birth year (-1 if unknown) are inferred from addr.
   Precondition: addr has one of the forms given in the box above and to the right. */
public Customer(String addr) {
```

(b) **3 pts.** List below the names of all wrapper classes mentioned in the table of method specifications that appears at the bottom of the last page and explain one purpose of such wrapper classes.
Question 4 (27 points) Miscellaneous topics.

(a) 4 pts. We made field customers in Customer static. Explain in 1-2 sentences whether it is better for customers to be static or non-static (if both options are equally good, explain why.)

(b) 4 pts. Field customers in Customer is public instead of private. Write an expression that could appear in any other class and that would evaluate to the number of Customer objects in Vector customers — without having to reference an object of class Customer.

(c) 6 pts. The first step in evaluation of the call new Customer(“Obama”, 1961, “o@a.com”) on the first page was to draw the object. The second step was to execute the constructor call that appears in that new expression. Below, do the first (and only the first) of the 4 steps in execution of that constructor call.

(d) (5 pts) Write a sequence of statements that exchanges the values of Customer variables c1 and c2. If other variables are needed, just declare them (this sequence would appear in a method body, so declared variables would be local variables):

```
// Swap c1 and c2
```

(e) (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 customer. */
public class Customer {
    private String lname; // last name (null if unknown)
    private int byear;    // birth year (-1 if unknown)
    private String eaddr; // email address (null if unknown)
    // list of all created Customers
    public static Vector<Customer> customers = new Vector<Customer>();
    /** Constructor: a customer with last name lname,
        birth year y (-1 if unknown), and email address a (null if unknown). */
    public Customer(String lname, int y, String a) {
    }
    /** Constructor: customer with last name n, birth year y
        (-1 for unknown), and unknown email address */
    public Customer(String n, int y) {
        /* Write this body as a single statement */
    }
    /** = String repr. of customer, consisting only of the last name. */
    public String toString() {
    }
}
// NOTE: Class PrefCustomer is defined on the next page
/** An instance is a "preferred" customer */
public class PrefCustomer extends Customer {
    // The level of preference; one of "bronze", "silver", "gold"
    private String level;

    /** Constructor: a preferred customer with last name lname,
    birth year y (use -1 if unknown),
    email address addr (use null if unknown), and
    level lev, which is one of "bronze", "silver", "gold" */
    public PrefCustomer(String lname, int y, String addr, String lev)
    {
        // Note: the fields of class Customer are private.
    }

    /** = the level of this preferred customer – one of
     "bronze", "silver" and "gold" */
    public String getLevel() {
    }

    /** = String representation of this preferred customer,
    in the form
    <last name>, <level>
    e.g. "gries, silver", "lee, gold" */
    public String toString() {
        // Note: the fields of class Customer are private.
    }
}

<table>
<thead>
<tr>
<th>String, Character, and Integer functions (we assume you know charAt, length, and substring)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Return</strong></td>
</tr>
<tr>
<td>-----------</td>
</tr>
<tr>
<td>int</td>
</tr>
<tr>
<td>String</td>
</tr>
<tr>
<td>int</td>
</tr>
<tr>
<td>boolean</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Vector methods</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>void v.add(p)</td>
</tr>
<tr>
<td>int v.size()</td>
</tr>
</tbody>
</table>