LAST NAME_____first name_____ Cornell Net id_____

CS1110 Prelim 1 6 October 2011

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 the 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.

Some questions deal with the two classes `Holiday` and `ReligiousH` shown on pages 4, 5.

Question 1 (34 points) Writing method bodies. In class `ReligiousH` (on page 4), which is to be a subclass of `Holiday`, there are missing parts for you to complete, indicated by underlines. Complete them, following any directions given in notes in the method bodies. Do not write any other methods. Be careful. 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**, and (3) know where private fields cannot be used.

Question 2 (20 points) Drawing objects. Draw the static components of `Holiday` somewhere below (do not draw file drawers) —you do not have to draw the method body. Then, evaluate the two new-expressions given below, the left one first, drawing objects under the new-expressions; above each new-expression, write its value. Include the partition for class `Object`, and put in two methods that you know are declared in `Object`. You may draw separate objects for String values or just write the String value in a field; either one will do.

Be sure to fill in the values of fields correctly. Do this based on the specifications of the methods in the classes. Assume these are the first new-expressions to be evaluated.

```
new Holiday("Xmas", "2011.12.25")
```

```
new religiousH("Passover", "Judaism",
                                  "2011/4/19", 8)
```

LAST NAME_____first name_____ Cornell Net id_____

Question 3 (22 points) String manipulation

(a) Write the body of function `lastIndexOfV`, specified below. Take advantage of function `Math.max(x,y)`, which yields the maximum of `x` and `y`. Hint: You need only find the maximum of a certain set of values.

Do *not* write `firstIndexOfV`, but you can use it in part (b).

```
/** = index of last lower-case vowel (one of a, e, i, o, or u) in s (-1 if none) */
public static int lastIndexOfV(String s) {




}
```

```
/** = index of first lower-case vowel (one of a, e, i, o, or u) in s (-1 if none) */
public static int firstIndexOfV(String s) {…}
```

(b) Ubbi Dubbi is a play language that works by adding "ub" before each vowel in a word —e.g. *spoke* becomes *spubokube* and *hello* becomes *hubellubo*. If you don't believe this, google "Ubbi Dubbi". There are four or five variations of Ubbi Dubbi. Below, write the function that translates a word using our own variation of Ubbi Dubbi:

CS1110 variation of Ubbi Dubbi: a word is translated as follows:

1. If s starts with an upper-case letter: the translation of s is s with its last lower-case vowel (if any), preceded by "ub", e.g. *Toronto* → *Torontubo* and *Syzygy* → *Syzygy*.
2. If s starts with an lower-case letter: the translation of s is s with its first lower-case vowel (if any), preceded by "ub", e.g. *hello* → *hubello* and *nth* → *nth*.

```
/** = s translated into the CS1110 variation of Ubbi Dubbi
      Precondition: s consists only of letters in a..z and A..Z. Only the first letter of s can be a
                    capital letter —all the rest are in a..z . s contains at least one character. */
public static String CS1110Ub(String s) {




}
```

String and Character functions (we assume you know `charAt`, `length`, and `substring`)		
Return	**Method**	**Value returned**
`int`	`s.indexOf(n)`	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)
`boolean`	`Character.isUpperCase(c)`	= "c is an uppercase (i.e. capital) letter"

Question 4 (22 points) Short questions

(a) What is the principle regarding filling in fields of the superclass and subclass in the constructor of a subclass?

(b) On page 5 is a class D, which contains a procedure m. Execute the *first two steps* (and only the first two steps) of the four steps of executing the call given below, drawing the frame for the call to the right of the call and doing anything else required in the first two steps.

```
D.m(4);
```

(c) When, during execution of a method call, is a local variable created (or drawn)? If you answered (b) correctly, that should help you.

(d) What is the inside-out rule? It can be explained in one (long) sentence.

LAST NAME_____first name_____ Cornell Net id_____

Must be a subclass of class Holiday, on next page

```
/** An instance maintains info about a religious holiday */

public _____ {
   private String r; // the name of the religion of this holiday
   private int days; // length of holiday, in days
   /** Constructor: a new religious holiday with name n,
         religion r, beginning on date w, and d days long. */
   public ReligiousH(String n, String r, String w, int d){
      // Note: write as many statements as are necessary




      _____;
   }

   /** Constructor: a new religious holiday with name n,
         religion r, beginning on date w, and 1 day long. */
   public ReligiousH(String n, String r, String w){
      // Note: the body of this constructor must contain exactly 1 statement.


      _____;
   }

   /** = String representation of this holiday in the following format:

            <religion>, <length of holiday>-day <string produced by toString in superclass>

       e.g. "Muslim, 30-day Ramadan on 1-30 August 2011." */
   public String toString(){
      return




                  _____ ;
   }
}
```

```
/** An instance maintains info about a holiday */
public class Holiday {
    private String hol;   // the holiday
    private String date;  // the date, in any form
    private int holNum;   // holiday's number in the list of holidays
                          // —i.e. the number of holidays when this one was created

    private static int number= 0; // total number of holidays

    /** Constructor: a holiday with name n. */
    public Holiday(String n, String d) {
        hol= n;
        date= d;
        holNum= number;
        number= number + 1;
    }

    /** = this holiday's number */
    public int getHolidayNum() {
        return holNum;
    }

    /** = total number of holidays */
    public static int getNumHolidays() {
        return number;
    }

    /** = String representation of the holiday */
    public String toString() {
        return "Holiday " + hol + " on " + date + ".";
    }
}
```

```
public class D {
   public static void m(int n) {
      if (n > 0) {
        int k;
        k= n+5;
        System.out.println(k);
      }
      System.out.println(2);
   }
}
```

0 _____ out of 02

1 _____ out of 34

2 _____ out of 20

3 _____ out of 22

4 _____ out of 22

Total _____ out of 100