ANSWER KEY
Exam 2
Computer Programming 230
Dr. St. John
Lehman College
City University of New York
Thursday, 5 November 2009

1. True or False:

   (a) T In Alice, there is an event that is processed as long as the world is running.

   (b) F In Alice, an event can execute its statements only once.

   (c) F An array is the only data structure available in Alice.

   (d) F In Alice, it is not possible to have lists of objects.

   (e) T A constructor is a method that creates a new object.

   (f) T A new class can be derived from an existing one.

   (g) F In Java, a variable can be used before it is declared.

   (h) F In Java, GUI components cannot contain other components.

   (i) T To generate a keyboard event, a component must have the keyboard focus.

   (j) T In Java, every event has a related listener interface.

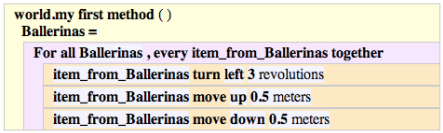
2. (a) What is an event? Give an example.

   `An event is something that happens. For example, when the mouse is dragged or a key is clicked are both examples of events.`

   (b) What is a listener in Java? Give an example.

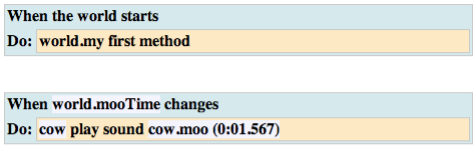
   `A listener responds to events in Java. For example, when a key is pressed, an event is triggered that the KeyListener responds to.`

3. Write the Alice code that would make the items in a list, called `Ballerinas`, turn in place 3 times and jump, in unison:

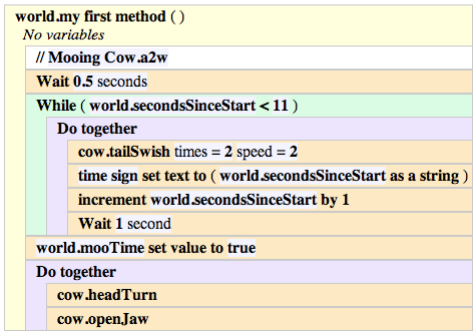


4. Answer the questions below, based on the following code:

Events



Methods



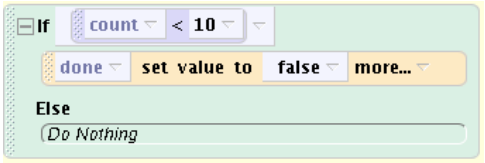
- What happens when the world starts?

  The method, my first method, is called. The cow swishes his tail back and forth for 10 seconds, a a sign counts up to 10.

- What does the other event do? When mooTime is true, cow plays the moo sound.

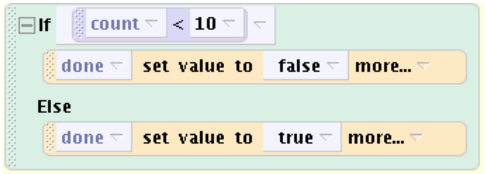
5. Line up the Alice statements with the corresponding statement in Java:

A



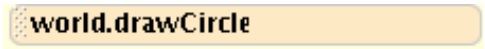
II

B



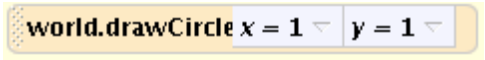
IV

C



III

D



I

E

V

I `world.drawCircle(x,y);`

II
```
if ( count < 10 )
    done = false;
```

III `world.drawCircle();`

IV
```
if ( count < 10 )
    done = false;
else
    done = true;
```

V `name = Integer.toString(count);`

6. To the right of each line of code, indicate the values stored in the variables immediately after those lines have been executed. If a variable is uninitialized, enter `?` in the box for that variable.

(a)
```
int x, y, z;
String a, b;

x = 5;
```

x	y	z	a	b
5	?	?	?	?

(b) `y = x/5;`

x	y	z	a	b
5	1	?	?	?

(c) `a = "Hi";`

x	y	z	a	b
5	1	?	“Hi”	?

(d) `b = a+a;`

x	y	z	a	b
5	1	?	“Hi”	“HiHi”

(e) `z = x*x+1;`

x	y	z	a	b
5	1	26	“Hi”	“HiHi”

(f) `a = "23";`

x	y	z	a	b
5	1	26	”23”	”HiHi”

(g) `y = Integer.parseInt(a);`

x	y	z	a	b
5	23	26	“23”	“HiHi”

(h) `x = y-1;`

x	y	z	a	b
22	23	26	“23”	“HiHi”

(i) `a = z % x;`

x	y	z	a	b
22	23	26	“4”	“HiHi”

(j) `x--;`

x	y	z	a	b
21	23	26	“4”	“HiHi”

7. Fill in the missing methods below:

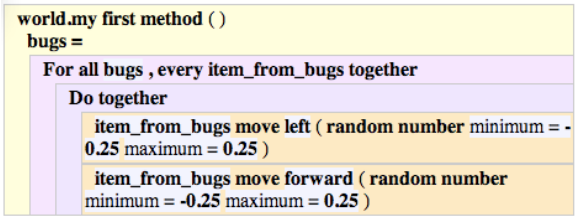
```
//********************************************************************************
//  Circle.java       Programming with Alice and Java
//
//  Represents a circle with a particular size, color, and location. The
//  circle can be drawn filled or unfilled.
//********************************************************************************
import java.awt.*;
public class Circle
{
   private int radius;
   private Color color;
   private int x, y;  // the circle's center point
   //-----------------------------------------------------------------------------
   //  Sets up the circle with the specified location, size, and color.
   //-----------------------------------------------------------------------------
   public Circle(int xCenter, int yCenter, int size, Color circleColor)
   {


         x = xCenter;
         y = yCenter;
         radius = size;
         color = circleColor;


   }
   //-----------------------------------------------------------------------------
   //  Draws the circle, filled, in the specified graphics context.
   //-----------------------------------------------------------------------------
   public void drawFilled(Graphics gc)
   {


        gc.setColor(color);
        gc.fillOval(x-radius, y-radius, radius*2, radius*2);


   }
}
```

8. Write the constructor for the `PushCounterPanel`. The `count` should be initialized to 0, the button should have the title, `Push me!`, and should have an associated listener from the class, `ButtonListener`. Also, add the button and label so they appear on the panel and set the background color for the panel.

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
public class PushCounterPanel extends JPanel
{
   private int count;
   private JLabel label;
   private JButton push;
   //-----------------------------------------------------------------------
   //  Sets up the interface on the panel.
   //-----------------------------------------------------------------------
   public PushCounterPanel()
   {


        count = 0;
        push = new JButton("Push me!");
        ButtonListener listener = new ButtonListener(this);
        push.addActionListener(listener);

        label = new JLabel("Pushes: " + count);

        add(push);
        add(label);

        setBackground (Color.cyan);
        setPreferredSize (new Dimension(350, 60));


   }
   //-----------------------------------------------------------------------
   //  Increments the counter and updates the label accordingly.
   //-----------------------------------------------------------------------
   public void incrementCount()
   {
      count++;
      label.setText("Pushes: " + count);
   }
}
```

9. Write a complete Alice program that has 5 bugs moving in random directions. When a bug is clicked with the mouse, it disappears from view.

`world.my first method:`



`Events:`

10. Write a **complete** Java program that prints `Hello World` to the console:

```
public class Hello {
  public static void main(String args[]) {
    System.out.println("Hello World");
  }
}
```