Code for Programming Assignment #5

By: Eric Dockery

Date: 11/14/13

Problem # 1 Programming Project 9.8

```
import javax.swing.*;

public class HorizontalRace {
	public static void main (String[] args){
		JFrame frame = new JFrame("The Horizontal Race");
		frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

		frame.getContentPane().add(new Racer());
		frame.pack();
		frame.setVisible(true);


	}
}
```

```
import java.awt.*;

import java.awt.event.*;


import javax.swing.*;




public class Racer extends JPanel {

	private final int width = 400, height = 200;

	private final int DELAY = 20, Length = 80;

	private Timer timer;

	private int  x, y, crossPoint;

	public Racer(){
        timer = new Timer(DELAY, new RaceListener());

        x = 0;
        y = height/2;
        crossPoint = width/2;

        setPreferredSize(new Dimension(width, height));
        setBackground( Color.green);
        timer.start();

    }

    public void paintComponent (Graphics page){
        super.paintComponent(page);
        page.setColor(Color.blue);
        page.drawLine(crossPoint, 0, crossPoint, height);

        if ( x+ Length < crossPoint){
            page.setColor(Color.red);
            page.drawLine(x, y,x +Length , y);

        }
        else if (x>crossPoint){
            page.setColor(Color.black);
            page.drawLine(x, y,x +Length , y);
        }
        else {
            page.setColor(Color.red);
            page.drawLine(x, y,crossPoint , y);
            page.setColor(Color.black);
            page.drawLine(crossPoint, y,x +Length , y);

        }

    }
    private class RaceListener implements ActionListener {

        @Override
        public void actionPerformed(ActionEvent e) {
            // TODO Auto-generated method stub
            x+= 1;
            if( x >= width){
                x = -Length;
                };
        repaint();
        }




    }
```

```
}
```

Problem #2 Programming Project 9.10

```
import java.awt.*;
import java.awt.Event.*;
import javax.swing.*;

public class Watch {

	public static void main (String[] args){
		JFrame frame = new JFrame ("Stop Watch");
		frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
		frame.getContentPane().add(new WatchPanel());
		frame.pack();
		frame.setVisible(true);
	}
}

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class WatchPanel extends JPanel {

private final int WIDTH = 400, HEIGHT = 100;
private JLabel timeLabel;
private JButton start, stop, reset;

private int mil, sec, min;

private Timer timer;

public WatchPanel()
{
    mil = 0;
    sec = 0;
    min = 0;

    timeLabel = new JLabel("00:00:00");
    start = new JButton("Start");
    stop = new JButton("Stop");
    reset = new JButton("Reset");
    timer = new Timer(10, new StopWatchListener());


    start.addActionListener(new StartButtonListener());
    stop.addActionListener(new StopButtonListener());
    reset.addActionListener(new ResetButtonListener());

    add(timeLabel);
    add(start);
    add(stop);
    add(reset);

    setPreferredSize (new Dimension(WIDTH, HEIGHT));
    timeLabel.setForeground(Color.red);
    setBackground (Color.white);
}

private class StopWatchListener implements ActionListener
{
    public void actionPerformed(ActionEvent event)
    {
        mil +=1;

     if(mil >= 99)
     {
         mil=0;
         sec +=1;
     }
     if(sec >= 60)
     {
         sec =0;
         min +=1;
     }
```

```
            if(min > 99)
            {
                mil = sec = min = 0;
            }
              String time = String.format("%02d:%02d:%02d", min, sec, mil);
              timeLabel.setText(time);
           }

       }

       private class StartButtonListener implements ActionListener
       {
                    public void actionPerformed(ActionEvent event)
                    {
                            timer.start();
                    }
            }

       private class StopButtonListener implements ActionListener
   {
                    public void actionPerformed(ActionEvent event)
                    {
                            timer.stop();
                    }
            }
```

```
    private class ResetButtonListener implements ActionListener
{
                    public void actionPerformed(ActionEvent event)
                    {
                   mil = sec = min = 0;
                   timeLabel.setText("00:00:00");


                    }
            }
}
```

Problem #3 Programming Project 10.3

```
public class Speaking {

	private Speaker current;

	public static void main(String[] args) {

		Speaking TheSpeaker = new Speaking();

		TheSpeaker.current = new Philosopher();
		TheSpeaker.current.announce("Philospher");
		TheSpeaker.current.speak();

		TheSpeaker.current = new Dog();
		TheSpeaker.current.announce("Dog");
		TheSpeaker.current.speak();

		TheSpeaker.current = new Wizard();
		TheSpeaker.current.announce("Wizard");
		TheSpeaker.current.speak();

		}

}

public interface Speaker {

	public void speak();
	public void announce(String str);

	}

public class Wizard implements Speaker{

	public void speak(){
		System.out.println("I will cast a spell on you!");

	}
	 public void announce(String str)
	    {
	      System.out.println("The " + str + " is speaking");
	    }
}


public class Dog implements Speaker {

	public void speak(){
		System.out.println("Bow wow ,Bow wow, Bow wow!!");


	}
	 public void announce(String str)
	    {
	      System.out.println("The " + str + " is speaking");
	    }
}

public class Philosopher implements Speaker {
	public void speak(){
		System.out.println("This is a philosophical statement!!");

	}
	 public void announce(String str)
	    {
	      System.out.println("The " + str + " is speaking");
	    }

}
```

Problem # 4 Programming Project 10.5

```
public class Movies {

	public static void main (String[] args){
		DVD[] movies = new DVD[7];
		movies[0] = new DVD ("The Godfather", "Francis Ford Coppla", 1972,
24.95, true);
		movies[1] = new DVD ("District 9", "Nevil Blomkamp", 2009, 19.95,
false);
		movies[2] = new DVD ("Iron Man", "Jon Favreau", 2008, 15.95, false);
		movies[3] = new DVD ("All About Eve", "Joseph Mankiewicz", 1950, 17.50, 
false);
		movies[4] = new DVD ("The Matrix", "Andy & Lana Wachowski", 1999, 19.95, 
true);
		movies[5] = new DVD ("Iron Man 2", "Jon Favreau", 2010, 22.99, false);
		movies[6] = new DVD ("Casablanca", "Michael Curtiz", 1942, 19.95, 
false);

		Sorting.TitleSort(movies);
		for (DVD dvd:movies){
		System.out.println(dvd);
		}
	}
}
public class Sorting{
	public static void TitleSort(Comparable[] list){
		int min;
		Comparable temp;
		for (int index = list.length-1; index >0 ; index--){
			min = 0;
			for( int scan = 1; scan <= index; scan++){

				if(list[scan].compareTo(list[min]) <0){
					min= scan;
				}
			}
			temp = list[min];
			list[min]= list[index];
			list[index] = temp;

		}

	}
}

import java.text.NumberFormat;

public class DVDCollection {
	private DVD[] collection;
	private int count;
	private double totalCost;

	public DVDCollection(){
		collection = new DVD[100];
		count = 0;
		totalCost =0.0;

	}

	public void addDVD (String title, String director, int year, double cost, 
boolean bluray){
		if(count == collection.length)
		{
			increaseSize();
			}
```

```java
        collection[count] = new DVD (title, director, year, cost, bluray);
        totalCost+= cost;
        count++;
    }
    public String toString(){
        NumberFormat fmt = NumberFormat.getCurrencyInstance();
        String report = "~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\n";
        report += "My DVD Collection\n\n";
        report += "Number of DVDs: " +count + "\n";
        report += "Average cost:" +fmt.format(totalCost/count);

        report += "\n\n DVD List: \n\n";


        for( int dvd = 0; dvd<count; dvd++){
            report += collection[dvd].toString() +"\n";
        }

            return report;
        }

    private void increaseSize(){
        DVD[] temp = new DVD[collection.length *2];
        for (int dvd = 0; dvd< collection.length; dvd++){
            temp[dvd] = collection [dvd];
        }
        collection = temp;
        }

}

import java.text.NumberFormat;

public class DVD implements Comparable{
    private String  title;
    private String director;
    private int year;
    private double cost;
    private boolean bluray;
    public DVD ( String title, String director, int year, double cost, boolean
bluray){
        this.title=title;
        this.director = director;
        this.year = year;
        this.cost = cost;
        this.bluray = bluray;


    }
    public String getTitle(){
        return title;
    }

    public String toString(){
```

```
		NumberFormat fmt = NumberFormat.getCurrencyInstance();
		String description;

		description = fmt.format(cost) + "\t" +year + "\t";
		description += title + "\t" +director;
		if (bluray){
			description += "\t" + "Blu-ray";

		}
		return description;

	}
	public int compareTo(Object nextObject){
		DVD otherDVD = (DVD)nextObject;
		return getTitle().compareTo(otherDVD.getTitle());
	}
}
```

Problem #5 Programming Project 8.19

```
import javax.swing.JFrame;

public class RubberLines {

	public static void main( String[] args){
		JFrame frame = new JFrame ("Rubber Lines");
		frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

		frame.getContentPane().add(new RubberLinesPanel());

		frame.pack();
		frame.setVisible(true);

	}

}
import java.awt.*;
import java.awt.event.*;
import javax.swing.JPanel;
import java.util.*;

public class RubberLinesPanel extends JPanel {

	private Pointpoint1 = null, point2 = null;
  private ArrayList<Point[]> pointsList = new ArrayList<Point[]>();
	public RubberLinesPanel()
	{
		LineListener listener = new LineListener();
		addMouseListener(listener);
		addMouseMotionListener(listener);
		setBackground(Color.black);
		setPreferredSize(new Dimension(400, 200));
	}
	public void paintComponent(Graphics page)
```

```
       {
             super.paintComponent(page);
             page.setColor(Color.yellow);
    Iterator<Point[]> pointIter= pointsList.iterator();

    while(pointIter.hasNext())
    {
      Point[] pair = pointIter.next();
      if (pair[0] != null && pair[1] != null)
        page.drawLine(pair[0].x, pair[0].y, pair[1].x, pair[1].y);
    }
       }
       private class LineListener implements MouseListener, MouseMotionListener
       {
             public void mousePressed(MouseEvent event)
             {
                    point1 = event.getPoint();
             }
             public void mouseDragged(MouseEvent event)
             {
                    point2 = event.getPoint();
      Point[] points = {point1, point2};
                    pointsList.add(points);

      repaint();
             }

             public void mouseClicked(MouseEvent event){}
             public void mouseReleased(MouseEvent event)
    {
      pointsList.clear();
      point2 = event.getPoint();
      Point[] points = {point1, point2};
      pointsList.add(points);

      repaint();
    }
             public void mouseEntered(MouseEvent event){}
             public void mouseExited(MouseEvent event){}
             public void mouseMoved(MouseEvent event){}
       }
}
```