CMSC 113 – COMPUTER SCIENCE 1
Lab#6 StdDraw Library – Practice

In this lab you will practice how to use the **StdDraw** library for creating simple graphics and animations.

Task#1: Review – StdDraw

The **StdDraw** library provides four useful **canvas control functions**, as show below (See pages 144-154).

```
public class StdDraw (basic control commands)

   void  setCanvasSize(int w, int h)        create canvas in screen window of
                                            width w and height h (in pixels)

   void  setXscale(double x0, double x1)    reset x-scale to (x0, x1)

   void  setYscale(double y0, double y1)    reset y-scale to (y0, y1)

   void  setPenRadius(double radius)        set pen radius to radius
```

By default, the canvas scale is set to (0, 1.0) for both X- and Y-coordinates. However, you can change the canvas settings using the above commands. The default size of pen radius is 0.002 (or 1/500th of default canvas size, or 1-pixel).

Drawing Shapes: The following table summarizes the shapes you can draw:

```
public class StdDraw (shapes)

   void  circle(double x, double y, double radius)

   void  filledCircle(double x, double y, double radius)

   void  square(double x, double y, double r)

   void  filledSquare(double x, double y, double r)

   void  rectangle(double x, double y, double r1, double r2)

   void  filledRectangle(double x, double y, double r1, double r2)

   void  polygon(double[] x, double[] y)

   void  filledPolygon(double[] x, double[] y)
```

Task#2: Draw a house

Study the picture of the house on the right carefully. Essentially it uses six shape commands to draw the house. On a piece of paper, draw the layout of the house and then write down commands to recreate the house. Do your drawing one step at a time using the steps shown below:

```
// Set the canvas to 600x600 pixels
// Set the X- and Y- scale to (0, 600)
// Draw the yellow wall
// Draw the red roof
// Draw the orange windows
// Draw the green door
// Draw the dark-gray door knob.
```
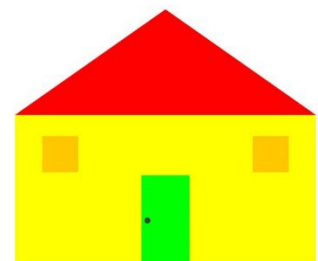

Please, DO NOT try and write commands all at once to draw the entire house!
You will need to understand each drawing command, the coordinates you will need to provide, and then draw. Color names are listed on page 150 of your text. You can make it an iterative, trial-and-error, process. Doing this for each step above will be helpful to you. Show your result to the instructor when completed.

Task#3: Write a Java program to generate N (input from command line argument) random points in a 500x500 pixel canvas. Use the colors (black, red, blue, green) for the points. If a point lies in the bottom left quadrant of the canvas, the point should be colored black, red if it lies in the bottom right quad of the canvas, blue for top left quad, and green for top right. Run your program for different values of N. It should look like the image shown below.
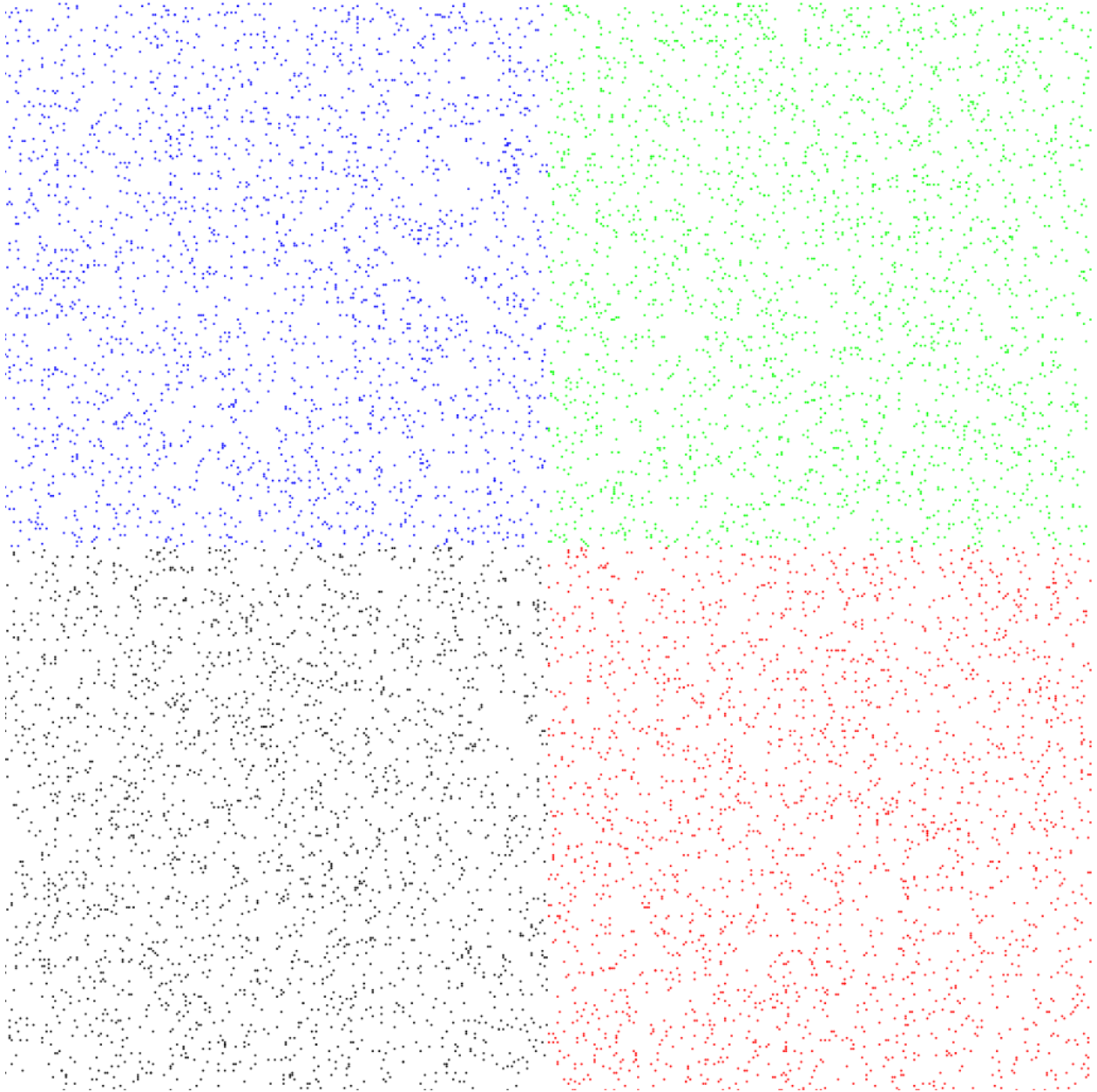
Task#4: Bouncing Ball

First study, then implement the program shown below:

```
public class Ball {                                  // Bouncing Ball Animation
    public static void main (String[] args) {
        int size = 500;        // Set canvas controls
        StdDraw.setXscale(0, size);
        StdDraw.setYscale(0, size);

        int bx = 10;                                 // ball <x,y> and radius
        int by = size/2;
        int br = 10;
        int dx = 1;                                  // ball displacement in x-

        StdDraw.enableDoubleBuffering();       // Offscreen drawing

        while ( true ) {                             // do forever
            StdDraw.clear();                         // clear offscreen canvas
            StdDraw.filledCircle(bx, by, br);   // draw ball
            StdDraw.show();                          // display offscreen canvas
            StdDraw.pause(5);                        // wait

            bx = bx + dx;                            // move the ball in x-dir
            if (bx+br >= size || bx-br <= 0)    // did it bump a L/R wall?
                dx = -dx;
        }
    } // main()
} // class Ball
```

(1) Run the program as shown above. You should see a black colored ball moving across and back.

(2) Change the color of the ball to RED.

(3) Add a y- displacement to the ball (**dy** = 1). Observe. The ball will escape the canvas from top/bottom.

(4) Add commands to restrict the ball so it does not escape.

Other stuff to try…

(5) What happens if you move the `StdDraw.clear()` command before the `while`-loop?

(6) Modify **dx** and **dy** so that they can be any integer in the range [1..3]. Run the program several times.

(7) Change the initial values of **bx** and **by** to be anywhere in the canvas (at least 10 away from walls).

When done with this Lab, send an e-mail to your instructor.