Programming Project 5
100 points

Submission Instructions
Open Eclipse and create a Java Project called *Project5*. Add a package named *tag* to this project. Add a class named *FreezeTagDriver* and a class named *MovingRectangle* to this package.

At the top of each file, enter a comment with your name, the assignment number, and the date.

When you are finished, export your project from Eclipse and upload it to Canvas before the due date. To do this, right click on the project name and select Export. Select General->Archive File and click Next. Select the project you wish to export and click Browse to browse to a location to save your file. Name this file *YourLastNameYourFirstNameProject5*. Upload this file to Canvas.
Remember that late assignments are not accepted in this course.
in this course.

Assignment

In this project, we will create a game called Freeze Tag.

Part 1:
Five rectangles start on the canvas at random locations. These rectangles have different widths and heights and different x and y velocities. They also have a random starting color. The width, height, and magnitude of the x and y velocities of each rectangle will not change during the game.

The rectangles move around the screen bouncing off the walls. Each time a rectangle bounces off a wall, its color changes to a new random color. The player uses the mouse to try to click on the rectangles. If the player successfully clicks on a rectangle, the rectangle will freeze and turn red. If the player can freeze all five rectangles, he or she wins.

Part 2:
In part two we will add a new feature to the game. When a moving rectangle collides with a frozen rectangle, the frozen rectangle will be unfrozen or, in other words, the frozen rectangle will start moving again. It should start moving with the same velocity (speed and direction) that it was moving before it stopped.

Your project will have two classes: FreezeTagDriver and MovingRectangle.

The MovingRectangle class should have the following private instance variables:

- xCoord, a int that stores the rectangle's current x-coordinate
- yCoord, a int that stores the rectangle's current y-coordinate
- width, an int that stores the rectangle's width
- height, an int that stores the rectangle's height
- xVelocity, an int that stores the rectangle's velocity in the x direction
- yVelocity, an int that stores the rectangle's velocity in the y-direction
- canvasSize, an int that stores the size of the StdDraw canvas
- color, a variable of type java.awt.Color that stores the rectangle's current color
- frozen, a boolean that stores whether the rectangle is frozen or not

The MovingRectangle class should have one constructor. This constructor should have 7 parameters: the starting x- and y-coordinates of the rectangle, the width and height of the rectangle, the x- and y-velocities, and the canvas size. The constructor should also initialize the color instance variable to a random color and initialize frozen to false.

The MovingRectangle class should have the following public methods:

draw()
Draws the rectangle at its current position with its current color.

move()
If the rectangle is not frozen, the rectangle should move based on its x and y velocities. If the rectangle is at the edge of the canvas (top, bottom, left, or right), it should "bounce" off the edge. When it bounces off the edge, its color should change to a new random color.

setColor (Color c)
A method that has one parameter of type Color and sets the rectangle's color to that color.

setRandomColor()
A method that sets the rectangle's color to a random color.

containsPoint(double x, double y)
This method has two parameters which represent the x and y coordinate of a point. The method should return true if the point is within the rectangle and false otherwise.

isFrozen()
This method returns true if the rectangle is frozen (not moving) and false otherwise.

setFrozen(boolean val)

This method sets the frozen instance variable to the value of the parameter.

isIntersecting (MovingRectangle r) (for part 2)
This method has one parameter of type MovingRectangle. The method returns true if the calling object and the parameter are intersecting and false if they are not. Hint: In Programming Project 2, we figured out how to determine if two rectangles intersect.

Your main method in the FreezeTagDriver class should do the following:

Part 1:

- Set the size and scale of the Canvas
- Create an array that contains five moving rectangle objects.
- Loop until the program exits (i.e. until the StdDraw window is closed). Your loop should clear the canvas and then call the move and draw method for each MovingRectangle object.

Within the animation loop you should check to see if the mouse has been clicked. (Hint: use the StdDraw.mousePressed method). If the mouse has been clicked inside a MovingRectangle object, the rectangle's color should be set to red and the rectangle should be frozen.

If all rectangles are frozen, you should display a message letting the player know that he or she wins.

Part 2:
Within the animation loop you should check to see if a moving rectangle has collided with a frozen rectangle. If a collision has occurred, the frozen rectangle's color should be set to a random color and the rectangle should start moving again.

Hints for completing the project:

This is a much larger program than we have written before. I recommend that you start early and work incrementally, adding one feature at a time.

See if you can get part 1 working with one rectangle first. Then add the array of five rectangles. Once you have part 1 working move on to part 2.

Grading Criteria (100 points possible)

Points	Criteria
0-5 points	**Correctness:** Does the MovingRectangle class contain the private instance variables described above?
0-5 points	**Correctness:** Does the MovingRectangle constructor properly initialize all instance variables?
0-10 points	**Correctness:** Does the MovingRectangle class contain the methods described above?
0-2 points	**Correctness:** Does the driver program correctly set the Canvas size and scale?
0-10 points	**Correctness:** Does the driver program use an array of five objects of type MovingRectangle? When each rectangle object is created, is it assigned a random starting position, width, height, x and y velocity and random color?
0-3 points	**Correctness:** Does the program continue until the StdDraw window is closed?
0-15 points	**Correctness:** Do the rectangle objects move correctly? Do they bounce off the edges? Do they change to a new random color when they bounce?
0-15 points	**Correctness:** Do the rectangles freeze and turn red when they are clicked?
0-10 points	**Correctness:** Does the game detect if all rectangles have been frozen and then display an appropriate "win" message?
0-20 points	**Correctness:** When a moving rectangle collides with a frozen rectangle, does the frozen rectangle begin moving again? (Part 2)
0-5 points	**Style:** Is the code easy to read? Is the code indented in a style similar to that shown in the textbook? Are blank lines used to divide the code into sections? Is a comment with the required information included at the top of the file? Are comments used to provide details that are not obvious? Are meaningful variable names used? (See Section 1.4 of the textbook for documentation and style guidelines.)