Name: _____ C. S. 1 Constructors Lab 2

1. Open the Shapes project. Right click on the Square diagram and select *Open Editor*

   - This is the source code for the Square object. You will notice that all the attributes listed in the Object Inspector are there.
   - Scroll down slightly – Under the attributes is the ***constructor*** for the Square object. Its ***signature*** is: `public Square()`.
   - Scroll down until you see `public void moveRight()`
   - What method does `moveRight()` invoke? _____
   - How many pixels does the object move? _____
   - Change the code so that object moves twice that amount.
   - Click on the ***Compile*** button on the top. Since you have changed the code you have to have the computer "translate" (compile) the new code.
   - Test your new code by creating a new Square object. Invoke its *Object Inspector*. Check the `xPosition` attribute before and after invoking its `moveRight()` method.

2. Let's automate exercise 6 from Lab 1.

   - Code below should look similar to what you wrote for exercise 6. Type this code in-between the `public Square()` method and the `public void makeVisible()` method. See the board for more instructions.

```
/*
 * Automate exercise 6 on Lab 1
 */
public void exercise6()
{
    erase();
    changeSize( ? );                    //change each ? to what you wrote yesterday
    moveVertical( ? );
    moveHorizontal( ? );
    draw();
    slowMoveHorizontal( ?);
}
```

   - After you type this you must compile (1st button on the top) your new code. If there are any mistakes there will be an error message at the bottom. The error made most often is to forget to place a semi-colon (;) at then end of each line.
   - Instantiate a new square object and invoke your new `exercise6()` method. If the square diagram is cross hatched and does not list `new Square()`, you must choose the `compile` option.

3. Modify the exercise6() method so that the square makes a complete lap of the canvas. This code will be handed in.

Overloaded Constructors

Right now the Square class has a ***default constructor*** (a constructor with no parameters) whose ***signature*** is `public class Square()`. Wouldn't it be nice to have the ability to place our square where we would like it to start? We can do this by supplying another constructor. Our only constraint is the number of ***parameters*** for each constructor we write must be different. i.e. each constructor's signature must be different. That way the compiler can tell which one we wish to use.

- Enter the below code under the default constructor.

```
/**
 * Create a new square with default color that starts at xStart, yStart.
 */
public Square(int xStart, int yStart)
{
    size = 30;
    xPosition = xStart;
    yPosition = yStart;
    color = "red";
    isVisible = true;
    draw();
}
```

- Instantiate a new Square object using our ***overloaded constructor*** and verify that it works.

4. Write a new constructor that allows you to enter the starting position, the color, and the size. (This code will be handed in.)
   - Use the signature below:

```
public Square(int xStart, int yStart, String startColor, int startSize)
```

   - Verify that your constructor works by instantiating 4 new Square objects at each corner of the canvas. These objects should all be different colors and different sizes.

5. After we instantiate any object, we always have to make it visible. What a pain! Take a look at the source code for the method `makeVisible()`. Modify each of your constructors so that they will be visible to start.

6. Write a new method called, `relay()`, that takes 4 squares, all size 10, of different colors, positioned at the corners of the canvas before `relay()` is invoked, and simulates a relay race. Use the signature below.

```
public void relay(Square nw, Square ne, Square se, Square sw)
```