Assignment 1A – Working with the SmileyFace Class

When the user clicks the second button, construct two SmileyFace objects as follows. You'll be generating several random numbers. Please generate them in this order so that I can test your program using pre-defined scripts:

(1) radius for 1st SmileyFace

(2) x-pos for 1st SmileyFace

(3) y-pos for 1st SmileyFace

(4) radius for 2nd SmileyFace

(5) x-pos for 2nd SmileyFace

(6) y-pos for 2nd SmileyFace

(7) Random growth for 1st SmileyFace

(8) Random growth for 2nd SmileyFace

1. The radii should be a random integer between 5 and 100. Below are examples of how to get randoms:

int r = (int)(Math.random()\*10) // random num between 0 and 9

int r = (int)(Math.random()\*10) + 1 // random num between 1 and 10

int r = (int)(Math.random()\*100) + 21 // random num between 21 and 120

int r = (int)(Math.random()\*80) + 21 // random num between 21 and 100

1. The x-positions should be in the range of the window’s width adjusted for the radius so that the SmileyFace(s) will not extend off the screen. Use the getWidth( ) method of the JPanel class to get the window’s width; since the GUIDriver is also a JPanel, get the window’s width like this:

int x = getWidth();

1. The y-positions should be in the range of the window’s height adjusted for the radius so that the SmileyFace(s) will not extend off the screen.
2. Resize the window and click the button several times to make sure it is working properly and the SmileyFace(s) do not extend off the screen.
3. Now grow each SmileyFace object by a random factor between 0 and 1 (this may make them extend off the screen).

int f = Math.random(); // gives a random between 0 and 1

1. Now examine the SmileyFace objects to see if they overlap. To do this you will need to calculate the distance between their centers and make sure that that is larger than the sum of their radii. (That makes sense right?). Do this as follows:
   1. Instead of trying to calculate the new radii of the SmileyFace(s) based on the above manipulations, you can query each SmileyFace’s radius by calling .getRadius( ) on them.
   2. Also, instead of making the GUIDriver remember the x-position and y-position of each SmileyFace, it’s better to query each object for their x- and y- positions. The problem with this is that there is no way right now to query x and y. You will need to write .getX( ) and .getY( ) methods to provide that service. See how it’s done for .getRadius( ) and do it like that.
   3. Now that you can query each object for their x, y, and radius, do the calculations to see if they overlap. Below are examples that you will need to calculate the distance between their centers.

Examples of doing calculations using the Math class:

Math.sqrt(x) // returns the square root of x

Math.pow(x,2) // returns x to the 2nd power

* 1. If they overlap (if the distance between their centers is less than the sum of their radii), make the one on top (the second one) frown. (You will have to remember how to use an if statement).