JAVA GRAPHICS ANIMATION

In this exercise we'll start writing a program to make a ball bounce on the screen ...

We start off with all of the normal stuff:

```
package graphics;

import hsa2.GraphicsConsole;
import java.awt.Color;

public class Bounce1  {

    public static void main(String[] args)  {
        //and we start writing our program here
    }
}
```

But graphics programs end up using a lot of objects,
so we write the program using the constructor of the class:

```
package graphics;

import hsa2.GraphicsConsole;
import java.awt.Color;

public class Bounce1 {
    public static void main(String[] args) {
        new Bounce1();
    }

    //this is called a constructor. We'll learn about them later.
    Bounce1() {

    }
}
```

Add needed global variables.

(These are important once the program is split up into different parts, called methods or functions, that all need to access the same data.)

```
package graphics;
import hsa2.GraphicsConsole;
import java.awt.Color;

public class Bounce1 {
	public static void main(String[] args) {
		new Bounce1();
	}

	//Global variables
	GraphicsConsole gc = new GraphicsConsole(800,600);
	int ballx = 100, bally = 100;		//location of ball
	int diameter = 40;			//diameter of ball
	int xspeed = 2;  // normally this is set to 2 or 3 pixels. Later, try 35
	int yspeed = xspeed;
	int sleepTime = 5; //controls speed of animation. Normally 1-10

	Bounce1() { //constructor
	}
}
```

Please use these numbers for the global variables. They are the right values to make a really cool display at the end.

Write the main program
(Notice the while loop)

```
package graphics;
import hsa2.GraphicsConsole;
import java.awt.Color;

public class Bounce1 {
	public static void main(String[] args) {
		new Bounce1();
	}

	//Global variables
	GraphicsConsole gc = new GraphicsConsole(800,600);
	int ballx = 100, bally = 100;			//location of ball
	int diameter = 40;				//diameter of ball
	int xspeed = 2;  // normally this is set to 2 or 3 pixels. Later, try 35
	int yspeed = xspeed;
	int sleepTime = 5; //controls speed of animation. Normally 1-10

	Bounce1() { //constructor
		setup();

		while(true) {	//main animation loop
			moveAndDrawBall();
			gc.sleep(sleepTime);
			/* the final thing in the loop must be “sleep”.
			If it doesn't sleep the screen doesn't get redrawn */
		}
	}
}
```

Write the setup() method.

Notice that it only runs once (from the constructor), before anything else happens.

```
package graphics;
import hsa2.GraphicsConsole;
import java.awt.Color;

public class Bounce1 {
	public static void main(String[] args) {
		new Bounce1();
	}

	//Global variables
	GraphicsConsole gc = new GraphicsConsole(800,600);
	int ballx = 100, bally = 100;
	int diameter = 40;
	int xspeed = 2;
	int yspeed = xspeed;
	int sleepTime = 5;

	Bounce1() {
		setup();
		while(true) {
			moveAndDrawBall();
			gc.sleep(sleepTime);
		}
	}

	void setup() {
		gc.setAntiAlias(true);
		gc.setLocationRelativeTo(null);		//centre the window
		gc.setColor( Color.RED.darker() );
	}
}
```

Move the ball, then draw it.

```
package graphics;
import hsa2.GraphicsConsole;
import java.awt.Color;

public class Bounce1 {
	public static void main(String[] args) {
		new Bounce1();
	}

	//Global variables
	GraphicsConsole gc = new GraphicsConsole(800,600);
	[...]

	Bounce1() {
		setup();
		while(true) {
			moveAndDrawBall();
			gc.sleep(sleepTime);
		}
	}
	void setup() {
		gc.setAntiAlias(true);
		gc.setLocationRelativeTo(null);
		gc.setColor(Color.RED.darker());
	}

	void moveAndDrawBall() {
		//gc.clearRect(ballx, bally, diameter, diameter);	//uncomment if desired
		ballx += xspeed;
		bally += yspeed;
		gc.fillOval(ballx, bally, diameter, diameter);
	}
}
```

Problems:

- We want the ball to bounce off the sides.
  Try and figure this out. Then we'll go over it as a class.

- It would be nice to make it have a random colour each time it hits the wall.
  After the previous problem is solved, it's easy to get this working

- How would we make a second ball? 20 balls?
  Our single ball has 5 variables (x, y, vx, vy, diam).
  We can't just add dozens of new variables.
  This will require objects …