Lab 03: Spinning Spheres

To begin, download the following code:

{} Spinning
Source Code

This code should compile and run with no edits.
In the code provided, there are three spheres that float around the center of the window.
Imagine how you would add another sphere. It'd be a bit annoying, as you'd have to have additional position, velocity, etc. variables floating around. Now imagine adding 10 spheres. That'd be hella annoying (or "wicked" annoying, depending on which part of the country you're in). And what if you wanted the user (rather than the coder) specify the number of spheres?

In this lab you will change the code so that there are a user-specified number of spheres. You'll do this by implementing a Sphere class (so as to avoid having massive amounts of repeated code) and then working with an array of `Sphere`s. This sort of process (changing the structure of a program) is called "refactoring" code.

In particular, you'll implement (at least) an `update` method and a `draw` method in `Sphere`.

- `update` should take in a double representing how much time has passed. It should update the `Sphere`'s position and velocity based on that.
- `draw` should take in a `Graphics` object (you'll pass in `g`, the `Graphics` object passed into `paintComponent`) and use it to draw the `Sphere` at its current position.

Your `while(true)` loop (in `Go`), in the end, should look like this:

```
while(true){
  for (Sphere s : spheres){
      s.update(1.0 / (double)FPS);
    }
}
```

where `spheres` is an a `Sphere[]` (an array of `Sphere`s).
Similarly, in the end, your `paintComponent` method should look like:

```
public void paintComponent(Graphics g) {
    g.setColor(Color.BLACK);
    g.fillRect(0, 0, WIDTH, HEIGHT);
    for (Sphere s : spheres){
      s.draw(g);
    }
}
```

Other than `.draw(Graphics g)` and `.update(double time)` is up to you to decide what fields and other methods the `Sphere` class should have.

The starting locations, sizes, and colors along with where the spheres begin is also up to you, but you should have **some of them go clockwise**, and others go **counter (anti) clockwise**.

For-each loops

If you're not familiar, the syntax `for (Sphere s : spheres)` is known as a "for-each" loop.
Suppose we have an array of `Sphere`s called `spheres`. The following two pieces of code do the same thing:

```
for (int i = 0; i < spheres.length; i ++){
    Sphere s = spheres[i];
    //some code here that mucks about with s
}
for (Spheres s : spheres){
  //that same code that mucks about with s
}
```

The line `for (Sphere s : spheres)` is read as "for each Sphere s in spheres".

Command-line arguments:

Recall that the main method has the following signature:

```
public static void main(String[] args)
```

This means that when we call `main`, we pass in a `String` array. This is done via the command line.
Try the following. Put:

```
for(String s : args){
  System.out.println(s);
  }
```

at the start of the `main` method and compile. Then run:

```
java Spinning
java Spinning 15
java Spinning lots of words here
```

As you can see, the words after `java Spinning` are placed into the `args` argument.
This lets the user specify values (e.g., the number of spheres they want) at run time.

A note about IDEs:
If you use an IDE (e.g., IntelliJ), you may run into troubles regarding command line arguments. You can either run your code from the terminal (a good thing to get used to doing), or do some googling for your IDE (along the lines of "command line arguments in intellij") to figure out how to input command line arguments.
The TAs and I can help, too.

Challenges

If you want a challenge:

- Instead of having every sphere go around the center point, have some spheres go around other spheres (like the Moon going around the Earth while the Earth goes around the Sun).
- If you run the program for a while, you'll notice that every sphere slowly drifts away from the center. Why? (Also, fix that so that the spheres all stay on the screen no matter how long it's run.)

What you should submit:

`Spinning.java` (Note: Your file must be called `Spinning.java`.)