Collisions

Making the balls collide

Right now, the balls, if they intersect, just pass through each other. We want them to bounce off each other. First, we will add the following methods to our ball.py file:

```python
def intersects(self, otherCircle):
    (x, y) = self.position
    (x2, y2) = otherCircle.position
    dsqrd = (x2-x)**2 + (y2-y)**2
    R = self.radius + otherCircle.radius
    if dsqrd < R**2:
        return True
    else:
        return False
```

The `intersects` method will test to see if the 2 balls overlap. The `bounce` method (already in your code) will make the balls bounce off each other.

Now, for each frame of the game, we need to see if any of the balls are colliding. Open BallData.py and add the following code inside the `evolve` method (you probably want to put it BEFORE the code that moves the balls):

```python
#check to see if any ball collides with any other
for ball in self.balls:
    for otherBall in self.balls:
        if ball == otherBall:                       #we don't want to compare ball with
            continue
        else:
            if ball.intersects(otherBall):
                ball.bounce(otherBall)              #what to do when they DO hit
```

You should be able to test at this point and see if collision detection is working. Note that we are going to make the balls do something besides bounce in a later tutorial, but it is fun to watch them do it!

Enhancing collision detection (Optional)

Note that the formula inside the `bounce` method takes into account the mass of the ball. Even though our balls are different sizes, they all currently have the same mass. Let’s create a formula that calculates mass based on the radius of the ball. The mass or weight of the ball can generally be computed by multiplying the volume of the ball times the density. We are going to assume that all our balls have the same density, so let’s just calculate the volume of each.

The mass (or volume in our case) of a ball can be calculated using this algebraic formula:

\[ \text{Volume} = \frac{4 \times \pi \times R \times R \times R}{3} \]

In python, I wrote it as:

```python
self.mass = (4 * math.pi * self.radius ** 3)/3
```

Put that line of code in your `__init__` method (in ball.py), replacing the line that currently sets `self.mass`. 