Section 8.1 The Starting Point: Newton’s Lab
Ex: 8.1 nothing happens
Ex: 8.2 creates sun, moon, planets -- different variations of them.
Ex: 8.3 act(), getMass()
Ex: 8.4 Sun (240), Planet(4.3), Moon (0.8)
Ex: 8.5 removes all objects from the world
line of code to remove objects: removeObjects (getObjects(Actor.class));

Sections 8.2 Helper Classes: SmoothMover and Vector
Can you create an instance of an abstract class? no
Is an abstract class a superclass or a subclass? superclass
Ex: 8.6. How many methods does Body inherit from SmoothMover? 10
List some of the methods here: accelerate, addToVelocity, getExactX, getExactY, getSpeed, invertHorizontalVelocity, invertVerticalVelocity, move, setLocation

Ex: 8.7 Which of the methods names appear twice? setLocation
How do the 2 versions differ? the parameters are integer in one and double in the other

Overloading(p 138): Overloaded methods have the same names but different parameters
Therefore, the two methods have different signatures. and which method to execute is determined by examining the parameters. This is done during runtime not when the code is compiled

Exercise 8.8. Study the documentation for Vector and SmoothMover.
How many methods are available in Vector? 11
List 2 of them: add, getDirection, getLength, getX, getY, revertHorizontal, revertVertical, scale, setDirection, setLength, setNeutral

How many methods are available in SmoothMover? 10
List 2 of them: accelerate, addToVelocity, getExactX, getExactY, getSpeed, invertHorizontalVelocity, invertVerticalVelocity, move, setLocation(int, int), setLocation(double, double)

Exercise 8.9 Which methods from SmoothMover can you call interactively? Every method has an access specifier of public so all of them can be called interactively.
Example: public void setLocation (double x, double y)
Exercise 8.10 - 8.11 Does the code compile after removing the ‘this’? __Yes_________.
Create a new Body and record the mass __zero________________
What should the mass be? ___300 (that is what was passed into the parameter)_________
Replace the ‘this’ so the line of code is once again this.mass = mass;
Create a new Body and record the mass ___300________________________

Do you think you should keep the this operator or remove it? __keep it__, Why?
The mass does not correctly get set if you don’t.

Note: Whenever passing a parameter into a constructor that has the same name as an instance field in a class you need to use the ‘this’ keyword to differentiate between the parameter being passed in and the instance current object’s instance field.

Read pg 140-141

Explain in your own words the purpose of the keywords “static final” in this line of code.

private static final double GRAVITY = 5.8;

Static creates a variable accessible by all instances of the class.
Final makes the variable a constant value that cannot be changed

What purpose does creating a constant variable serve or why would you want a constant variable?

Declaring a variable as constant prevents the value from being changed by mistake. Values, such as the constant for gravity, should never changed so a constant is appropriate. It also creates more readable code using a variable such as GRAVITY rather than the magic number 5.8. A magic number is any value that isn’t self explanatory.

Section 8.4

8.14: Sun goes up (-90) at speed of .03, planet goes down (90) at speed of 2.2
8.15: Line of code to move left instead of right:

public Body()
{
    this (20, 300, new Vector(180, 1.0), defaultColor);
}
or

public Body()
{
    this (20, 300, new Vector(-180, 1.0), defaultColor);
}
Explanation: The first argument when creating a new vector is the direction and the second is the speed. Changing the direction to 180 or -180 has the effect of moving the object to the left.

Special Note: The following line of code would also make the object move left because the speed is set to a ‘negative’.

```java
this (20, 300, new Vector(0, -1.0), defaultColor);
```

Because of how vector movement is calculated this negative values makes the vector move in the ‘opposite’ of its original direction. If the direction is any angle other than zero the effects may not be what is desired. Since the second parameter represents a ‘length’ of a vector and since lengths should not be positive it is not good practice to use a -1.0 for the speed.

Section 8.5
8:17 - 8.18: Range for red, green blue parameters is 0 - 255
8:19: The import statement is the second line in code right after import greenfoot.*

```java
import greenfoot.*;
import java.awt.Color;
```

8.20: Error: Cannot find symbol, class Color
8:21 - 8:26 Complete the code to applyForces (see pg 146 in text). Textbook provides explanation and code. Students need to add to the Body class. java.util.List needs to be imported as well.
Section 8.7

How many parameters does the sqrt method have?  
- zero  
- one  
- two  

What data type are the parameters?  
- integer  
- double  
- string  

What data type does the sqrt method return?  
- integer  
- double  
- string  

EX 8.28  max(int, int) or just max()  
Ex 8.29. Variables matches shown below.

<table>
<thead>
<tr>
<th>Variable from Formulas</th>
<th>Variable from Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>dx</td>
</tr>
<tr>
<td>b</td>
<td>dy</td>
</tr>
<tr>
<td>c</td>
<td>distance</td>
</tr>
<tr>
<td>G</td>
<td>GRAVITY</td>
</tr>
<tr>
<td>mass1</td>
<td>this.mass</td>
</tr>
<tr>
<td>mass2</td>
<td>other.mass</td>
</tr>
</tbody>
</table>

8:30 - 8.31: Planets circulate around sun, moon around planet  
When gravity is changed the planets move quickly further away, maybe slingshot some around each other. the higher the gravity the more the slingshot,
8.32 - 8.33: Experimenting to try to get a stable space. Don’t spend too much time with it.

**Drill and Practice: Some possible solutions**

8.35: randomBodies(20); // change the input to whatever number you like
8.36: Experimenting -- move on fairly quickly
Ex: 8.37 An example - changed the bolded code to make obstacles go diagonal.
8.34: Study the code

```java
public void createObstacles()
{
    int i = 0;
    while (i < soundFiles.length)
    {
        addObject (new Obstacle (soundFiles[i] + ".wav"), 80 + i*60, 50 + i * 40);
        i++;
    }
}
```

8:38 - 8:39 - various options will occur

8:40 - 8:41: Create a changeColor Method in the Body class. It will be something like this:

```java
//Changes the color of the current body
private void changeColor()
{
    int r = Greenfoot.getRandomNumber(255);
    int g = Greenfoot.getRandomNumber(255);
    int b = Greenfoot.getRandomNumber(255);
    Color color = new Color (r,g,b);
    GreenfootImage image = this.getImage();
    int size = image.getWidth();
    image.setColor (color);
    image.fillOval (0, 0, size-1, size-1);
    setImage (image);
}
```

**Vocab Practice**

A -- 3
B -- 1
C -- 4
D -- 2
E -- 5

<table>
<thead>
<tr>
<th>Overloading</th>
<th>same name different parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>this</td>
<td>refers to current object</td>
</tr>
<tr>
<td>constant</td>
<td>use the keyword final to create constant variables. These variables cannot be changed during execution</td>
</tr>
<tr>
<td>helper class</td>
<td>Vector or SmoothMover. A class that adds functionality and uses abstraction to hide the details.</td>
</tr>
<tr>
<td>-------------</td>
<td>--------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>abstract class</td>
<td>SmoothMover. It is a superclass that is generic – it provides a template for the subclasses under it. Because it is abstract you cannot create an object of SmoothMover.</td>
</tr>
</tbody>
</table>