Chapter 7 Worksheet

We are going to work with the GreenfootImage and programmatically draw a background for a scenario.

(1) Fill in the code in the constructor of Background to set the color of the background to yellow.

```
import greenfoot.*;  // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)

public class Background  extends World
{
    public Background() {
        super(600,600,1);









    }
}
```

(2) Add the method definition for a method named addDots that takes as a parameter an integer for the number of dots to put on the background.

```
public class Background  extends World
{
    public Background() {
        super(600,600,1);
        GreenfootImage background = getBackground();
        background.setColor(java.awt.Color.YELLOW);
        background.fill();
    }















}
```

Chapter 7 Worksheet

(3) Fill in the method addDots so it retrieves the background image and stores it in a local variable.

```
public class Background  extends World
{
    public Background() {
        super(600,600,1);
        GreenfootImage background = getBackground();
        background.setColor(java.awt.Color.YELLOW);
        background.fill();
    }

    private void addDots(int numDots) {




    }

}
```

for-loop syntax:

```
for(initialization; condition; increment) {
      //code to be repeated
}
```

initialization

condition

code to be repeated

increment

(4) Fill in the code for the for-loop so that the loop body executes 10 times

```
for(                                      ) {
}
```

ANSWER:

```
for(                                      ) {
}
```

(5) Insert a loop into the addDots method whose loop body will execute numDots times.

```
    private void addDots(int numDots) {
         GreenfootImage background = getBackground();






    }
```

Chapter 7 Worksheet

(6) Start to fill in the body of the for-loop so that each time the loop executes a random value is chosen for an x coordinate and a random value is chosen for a y coordinate.

```
    private void addDots(int numDots) {
         GreenfootImage background = getBackground();
         for(int count = 0; count < numDots; count = count + 1) {







         }
    }
```

(7) Add code to the for-loop body so that the drawing color is set to blue and an oval is drawn at the x and y coordinate chosen previously with size 2x2.

```
    private void addDots(int numDots) {
         GreenfootImage background = getBackground();
         for(int count = 0; count < numDots; count = count + 1) {
            int x = Greenfoot.getRandomNumber(getWidth());
            int y = Greenfoot.getRandomNumber(getHeight());







         }
    }
```

(8) What changes are needed to get randomly colored dots?

```
private void addDots(int numDots) {
   GreenfootImage background = getBackground();

   for(int count = 0; count < numDots; count = count + 1) {
      int x = Greenfoot.getRandomNumber(getWidth());
      int y = Greenfoot.getRandomNumber(getHeight());




      background.setColor(new java.awt.Color(0, 0, Greenfoot.getRandomNumber(256)));

      background.fillOval(x, y, 3, 3);
   }
}
```

(9) What changes are needed to get randomly sized dots?

```
private void addDots(int numDots) {
   GreenfootImage background = getBackground();

   for(int count = 0; count < numDots; count = count + 1) {
      int x = Greenfoot.getRandomNumber(getWidth());
      int y = Greenfoot.getRandomNumber(getHeight());

      background.setColor(new java.awt.Color(0, 0, Greenfoot.getRandomNumber(256)));




      background.fillOval(x, y, 3, 3);
   }
}
```

(10) Write the code in the Square class that would set the initial image of the Square actor to be a filled-in square of size 5x5.

```
public class Square extends Actor {

   public Square() {








   }
}
```

(11) Write the code in the Square so that when act is called, the square will grow in size by 1 pixel in each dimension.

```
public class Square extends Actor {

   public Square() {
      GreenfootImage image = new GreenfootImage(5,5);
      image.setColor(java.awt.Color.GREEN);
      image.fill();
      setImage(image);
   }

   public void act() {





   }
}
```

(12) Modify the code in act so that when the square reaches a certain size, it will disappear.

```
public class Square extends Actor {

   public Square() {
      GreenfootImage image = new GreenfootImage(5,5);
      image.setColor(java.awt.Color.GREEN);
      image.fill();
      setImage(image);
   }

   public void act() {
      GreenfootImage image = getImage();
      image.scale(image.getWidth()+1, image.getHeight()+1);
      setImage(image);





   }
}
```

(13) Modify the code again, so that the maximum size of the square (for disappearing) is passed in as a parameter to the constructor of the square.

```
public class Square extends Actor {



   public Square(                        ) {
      GreenfootImage image = new GreenfootImage(5,5);
      image.setColor(java.awt.Color.GREEN);
      image.fill();
      setImage(image);
   }

   public void act() {
      GreenfootImage image = getImage();
      image.scale(image.getWidth()+1, image.getHeight()+1);
      setImage(image);



      if(image.getWidth() >= 100) {
         getWorld().removeObject(this);
      }
   }
}
```

(14) What changes are needed to make the image grow and shrink?

```
public class Square extends Actor {

   private int maxSize;




   public Square(int max) {
      maxSize = max;
      GreenfootImage image = new GreenfootImage(5,5);
      image.setColor(java.awt.Color.GREEN);
      image.fill();
      setImage(image);




   }

   public void act() {
      updateImage();
   }
   private void updateImage() {

      GreenfootImage image = getImage();




   }
}
```