Opening Exercise

Use a Random object to create an object that simulates a standard six-sided die. A Die should have a roll() method.

```java
import java.util.Random;

public class Die {
    
    private static Random generator = new Random();

    // FILL IN THE BLANK

}
Exercise: Improving Our Solution

How can we modify our `Die` class to support dice of any number of sides?

```java
public class Die {
    private static Random generator = new Random();

    public Die() {
    }

    public int roll() {
        int randomValue = generator.nextInt(6);
        return randomValue + 1;
    }
}
```
Using Java Objects

We have always created and use objects through Interactions pane.

How can we run Java programs from outside of Dr. Java?
A main() Method

```
public static void main( String[] args )
{
    Die d1 = new Die();
    Die d2 = new Die();
    System.out.println(d1.roll() + d2.roll());
}
```

The code in the method body is just like the code we write to create and use objects in the Interactions pane. It must create objects to use — it cannot call the methods directly!
How Do We Learn to Use a New Class?

Random is a pretty cool class. How can we learn to do more with it?

Read the source code.

Look at its Javadoc.
Javadoc as a Tool

At a command-line prompt:

mac os x > javadoc DiffSound.java

generates the file: DiffSound.html

mac os x > javadoc *.java

generates hyperlinked documentation for all the Java source files in the current directory.
What is Javadoc?

a program that writes a file

...so what is a file?
A File is ...

a place
for your stuff
Data Compression and Files

This is what I showed to demonstrate compression:

![Diagram showing a source file being compressed by a program to produce a compressed file.]

But this isn't actually how our data compression for Sound worked.
Our Sound Compression

It did this:

To save our DiffSound objects, we need to write them to a file.
How to Write to a File

A simple way of writing a file is:

1. Create a file object.
2. Write data to it using its write() and newLine() methods.
3. Close the file.

#2 and #3 are just like what we've done in the past. #1 requires a "trust me" moment or two — for now.
Our Compressed File

... is bigger than the original!

Why?
Our Compressed File

FileWriters are for creating *text* files,
and text is an inefficient encoding!

This semester, we will spend our time dealing with text as our third medium.

To save our data in a more compact format, we will have to learn about another kind of file object — in CS II or by our own research!
How Do We Learn to Use a New Class?

BufferedWriter is a pretty cool class. How can we learn to do more with it?

Read the source code.

Look at its Javadoc.