Previous Lecture:
- Review
- Defining a class—constructors

Today's Lecture:
- Method toString()
- Methods with parameters

Assigned reading:
- T Sec 3.1.4, 3.2

Announcements
- Section will be in the lab (UP B7)
- P3 has been posted. Due 3/11 R 3:30pm
- Lecture this Thursday is cancelled.
  "Substitute teacher" is Dr. Java.
  - Activity 4 on Lesson Page 3-6 (why private?)
  - Activities on Lesson Page 2-5 (top-down design)

Constructor invocation
```
new class-name(expression-list)
```
- Above expression yields a reference to a new object of the given class-name
- The defined (or default) constructor is invoked on the new object created by new

Creating an object
```
public class Client {
    public static void main(String[] args) {
        Interval in1;
        in1 = new Interval(0.2, 0.7);
    }
}
```

Method toString()
- Every object has default method toString
- Automatically invoked by print, println
```
Interval a = new Interval(1, 2);
System.out.println(a);
```
- Some default text will be printed unless you define a toString method
Method `toString()`

- Usually defined to give a useful description of an instance of a class.
- E.g., useful description of an instance of `Interval` would be the mathematical notation for an `Interval`, e.g. `[3, 7.5]` for an `Interval` object with base 3 and width 4.5.

Method with input parameter

- Write an instance method `expand(double f)` that expands the `Interval` by a factor of `f`.
- What should be the method header?
- Parameter of primitive type: pass by value
  I.e., value is copied

Method with input parameter

- Write an instance method `isIn(Interval i)` that returns the boolean value `true` if the instance is in `Interval i`. Return `false` otherwise.
- Parameter of non-primitive type: pass by reference
  I.e., Reference is copied; object itself is not copied

```java
/** ="this Interval is in i" */
public boolean isIn(Interval i) {
    return (getBase()>=i.getBase() &&
            getEnd()<=i.getEnd());
}
```

```java
public boolean isIn(Interval i) {
    if (getBase()>=i.getBase() &&
        getEnd()<=i.getEnd() == true)
        return true;
    else
        return false;
}
```

/** ="this Interval is in i" */
```java
public boolean isIn(Interval i) {
    return (getBase()>=i.getBase() &&
            getEnd()<=i.getEnd());
}
```

```java
public boolean isIn(Interval i) {
    boolean in = getBase()>=i.getBase() &&
                 getEnd()<=i.getEnd();
    return in;
}
```