Review questions

- Where do you put declarations for instance and static variables?
  - Immediately inside the class definition
    (immediately inside `class Interval { ... }`)?
  - Inside a method definition
    (e.g., inside method `main`)?

- What is the difference between instance and class variables (or methods)?

Review questions, cont’d

- If a variable does not store a value of primitive type, what does it store?

- What is the keyword for “protecting” instance variables? (i.e., not allow access from another class.)

- What is the keyword for instantiating an object?

Constructor invocation

```java
new class-name( expression-list )
```

- The value of above expression is 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

```java
public class Client {
    public static void main(String[] args) {
        Interval i1;
        i1 = new Interval(0.5, 1);
    }
}
```

```java
class Interval {
    private double base; // low end
    private double range; // interval width

    // Constructor
    public Interval(double base, double r) {
        this.base = base;
        range = r;
    }

    public double getEnd() {
        return base + range;
    }
}
```

Method with input parameter

- Write an instance method `expand(double f)` that expands the `Interval` by a factor of `f`.
- Where do you put this method?
- What should be the method header?

```java
public class Client {
    public static void main(String[] args) {
        Interval i1 = new Interval(0.2, 0.7);
        double x = 2;
        i1.expand(x);
        System.out.println(i1.getEnd());
    }
}
```

```
// Expand interval by factor of f
```

```
public Interval(double b, double r) {
    this.base = b;
    this.range = r;
}
```

- Keyword `this` returns a reference to the object itself
- Use keyword `this` only when it is necessary. (It is not necessary in the example above.)
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.

- Be sure to read Sec 5.0

```java
// Check if self is subset of interval i
public class Client {
    public static void main(String[] args) {
        Interval i1 = new Interval(0.2, 0.7);
        Interval i2 = new Interval(Math.random(), 0.2);
        System.out.println(i2.isIn(i1));
    }
}
```

```java
// Check if self is subset of interval i
public boolean isIn(Interval i) {
    return base>=i.base && getEnd()<=i.getEnd();
}
```

```java
// Check if self is subset of interval i
public boolean isIn(Interval i) {
    boolean in = base>=i.base && getEnd()<=i.getEnd();
    return in;
}
```
// Check if self is subset of interval i
public boolean isIn(Interval i) {
    return
    base>=i.base && getEnd()<=i.getEnd();
}

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

Class Variables & Methods
- Shared by all instances of a class
- Only one copy no matter how many objects have been instantiated
- Examples:
  - A variable to keep track of how many Intervals have been created
  - A constant used by the whole class

class Interval {
    private double base; // low end
    private double range; // interval width
    public final static double maxWidth=5;

    public Interval(double b, double r) {
        base = b;
        range =
    }

    // Other methods below ...
}

Class (static) method
- Write a class method
  `overlap(Interval a, Interval b)`
- that returns a new `Interval` if Intervals `a` and `b` overlap. Return `null` otherwise.
- What is the method header?