Parameter passing made easy

- *Everything is passed by value*
- *The result of the program is always as if a copy is made of the parameter, and that copy is passed to the routine.*
- This is true of *references*. They act pretty much like pointers passed by value in C.
- This is all you have to remember

Parameter passing in Java

- In Java objects are *always* accessed via references
- Java reference: These references are different from, and simpler than, C++ references
  - Unlike a C++ reference, a reference can be reassigned
  - Thus, unlike a C++ reference, the type of the object referred to by the reference changes
  - Unlike C++, references cannot be to a primitive type

```
public class B {

   public int age;

   public B(int a) {
      age = a;
   }

   public void print(String s) {
      System.out.println(s+", "+"B object "+age);
   }
}
public class D extends B {

   public int weight;

   public D(int a, int w) {
      super(a);
      weight = w;
   }

   public void print(String s) {
      System.out.println(s+" "+"D object "+age+", "+weight);
   }
}
```

Examples

```
public class T {

   /* basic reference operations */

  public static void main(String[] args) {
     B b1 = new B(50);
     b1.print("b1{50} ");
     D d = new D(51,100);
     d.print("d{51, 100}");
     B b2 = (B) d;
     b2.print("b2{51,100} which is d");
     b2 = b1;
     b2.print("b2{50} which again b1 again");
     b1.print("b1{50} before xchange");
     d.print("d{51,100} before xchange");
     xchangeWrong(b1, (B) d);
     b1.print("b1{50} after xchange");
     d.print("d{51,100} after xchange");
  }
```

```
  public static void xchangeWrong(B bee1, B bee2) {
        B b = bee1;
        bee1.print("bee1 before xchange");
        bee2.print("bee2 before xchange");
        bee1 = bee2;
        bee2 = b;
        bee1.print("bee1 after xchange");
        bee2.print("bee2 after xchange");
     }
  }
```

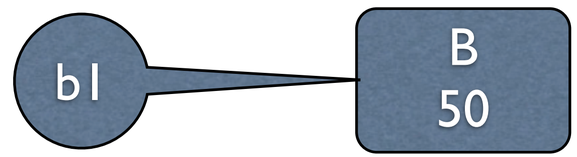
Example continued

```
public class T {

   /* basic reference operations */

  public static void main(String[] args) {
     B b1 = new B(50);
     b1.print("b1{50} ");
     D d = new D(51,100);
     d.print("d{51, 100}");
     B b2 = (B) d;
     b2.print("b2{51,100} which is d");
     b2 = b1;
     b2.print("b2{50} which again b1 again");
     b1.print("b1{50} before xchange");
     d.print("d{51,100} before xchange");
     xchangeWrong(b1, (B) d);
     b1.print("b1{50} after xchange");
     d.print("d{51,100} after xchange");
  }
```

```
  public static void xchangeWrong(B bee1, B bee2) {
        B b = bee1;
        bee1.print("bee1 before xchange");
        bee2.print("bee2 before xchange");
        bee1 = bee2;
        bee2 = b;
        bee1.print("bee1 after xchange");
        bee2.print("bee2 after xchange");
     }
  }
```

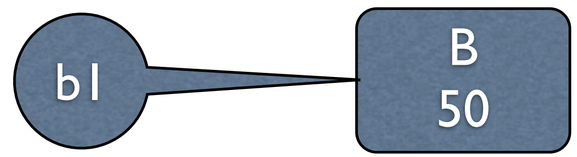


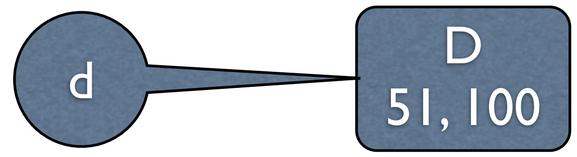
```
public class T {

    /* basic reference operations */

   public static void main(String[] args) {
      B b1 = new B(50);
      b1.print("b1{50} ");
      D d = new D(51,100);
      d.print("d{51, 100}");
      B b2 = (B) d;
      b2.print("b2{51,100} which is d");
      b2 = b1;
      b2.print("b2{50} which again b1 again");
      b1.print("b1{50} before xchange");
      d.print("d{51,100} before xchange");
      xchangeWrong(b1, (B) d);
      b1.print("b1{51} after xchange");
      d.print("d{51,100} after xchange");
   }
```

```
   public static void xchangeWrong(B bee1, B bee2) {
         B b = bee1;
         bee1.print("bee1 before xchange");
         bee2.print("bee2 before xchange");
         bee1 = bee2;
         bee2 = b;
         bee1.print("bee1 after xchange");
         bee2.print("bee2 after xchange");
      }
   }
```



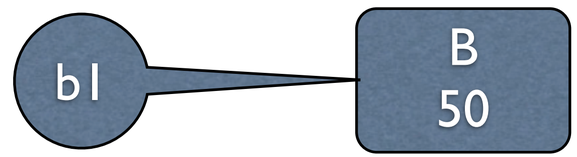


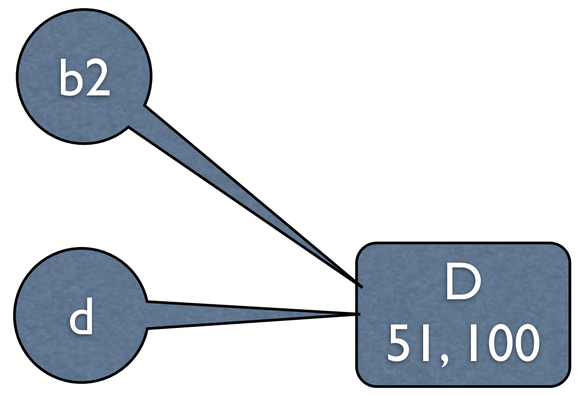
```
public class T {

   /* basic reference operations */

  public static void main(String[] args) {
     B b1 = new B(50);
     b1.print("b1{50} ");
     D d = new D(51,100);
     d.print("d{51, 100}");
     B b2 = (B) d;
     b2.print("b2{51,100} which is d");
     b2 = b1;
     b2.print("b2{50} which again b1 again");
     b1.print("b1{50} before xchange");
     d.print("d{51,100} before xchange");
     xchangeWrong(b1, (B) d);
     b1.print("b1{51} after xchange");
     d.print("d{51,100} after xchange");
  }
```

```
  public static void xchangeWrong(B bee1, B bee2) {
        B b = bee1;
        bee1.print("bee1 before xchange");
        bee2.print("bee2 before xchange");
        bee1 = bee2;
        bee2 = b;
        bee1.print("bee1 after xchange");
        bee2.print("bee2 after xchange");
     }
  }
```



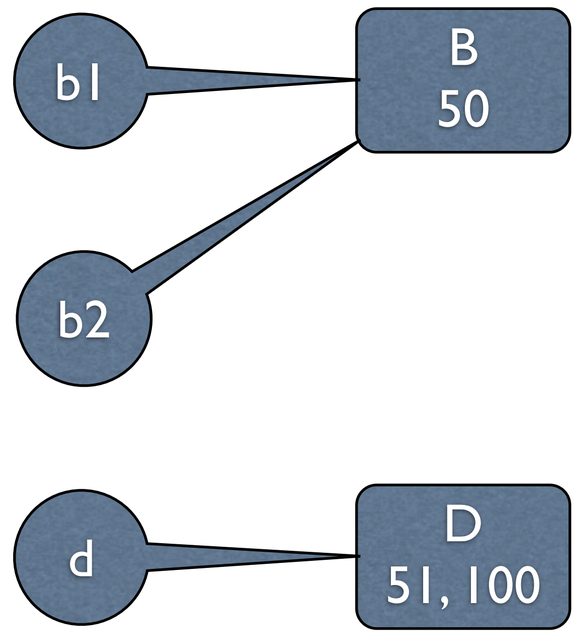


```
public class T {

   /* basic reference operations */

  public static void main(String[] args) {
     B b1 = new B(50);
     b1.print("b1{50} ");
     D d = new D(51,100);
     d.print("d{51, 100}");
     B b2 = (B) d;
     b2.print("b2{51,100} which is d");
     b2 = b1;
     b2.print("b2{50} which again b1 again");
     b1.print("b1{50} before xchange");
     d.print("d{51,100} before xchange");
     xchangeWrong(b1, (B) d);
     b1.print("b1{51} after xchange");
     d.print("d{51,100} after xchange");
  }
```

```
  public static void xchangeWrong(B bee1, B bee2) {
       B b = bee1;
       bee1.print("bee1 before xchange");
       bee2.print("bee2 before xchange");
       bee1 = bee2;
       bee2 = b;
       bee1.print("bee1 after xchange");
       bee2.print("bee2 after xchange");
    }
  }
```

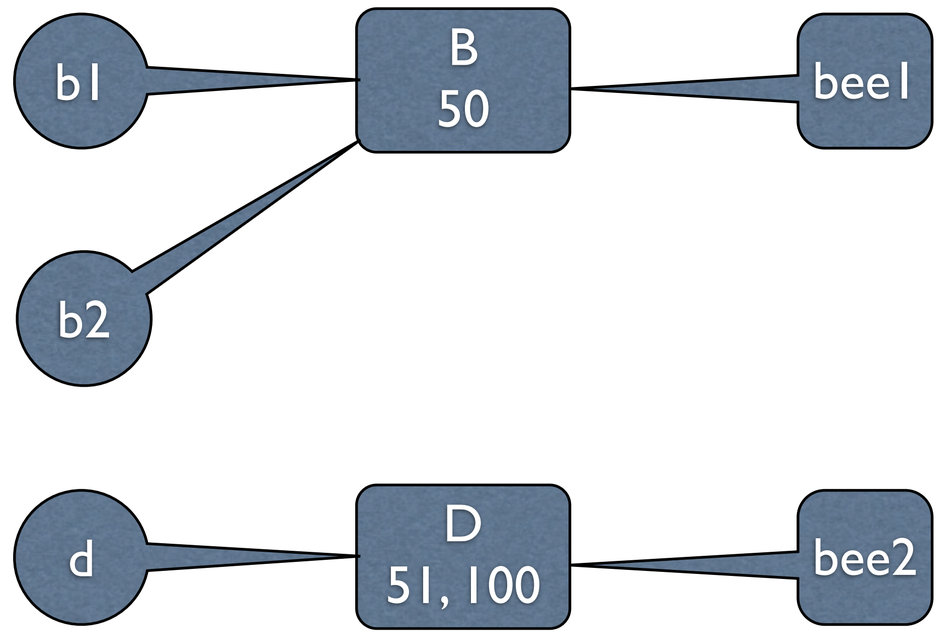


```
public class T {

   /* basic reference operations */

  public static void main(String[] args) {
     B b1 = new B(50);
     b1.print("b1{50} ");
     D d = new D(50,100);
     d.print("d{51, 100}");
     B b2 = (B) d;
     b2.print("b2{51,100} which is d");
     b2 = b1;
     b2.print("b2{50} which is b1 again");
     b1.print("b1{50} before xchange");
     d.print("d{51,100} before xchange");
     xchangeWrong(b1, (B) d);
     b1.print("b1{51} after xchange");
     d.print("d{51,100} after xchange");
  }
```

```
  public static void xchangeWrong(B bee1, B bee2) {
       B b = bee1;
       bee1.print("bee1 before xchange");
       bee2.print("bee2 before xchange");
       bee1 = bee2;
       bee2 = b;
       bee1.print("bee1 after xchange");
       bee2.print("bee2 after xchange");
    }
}
```

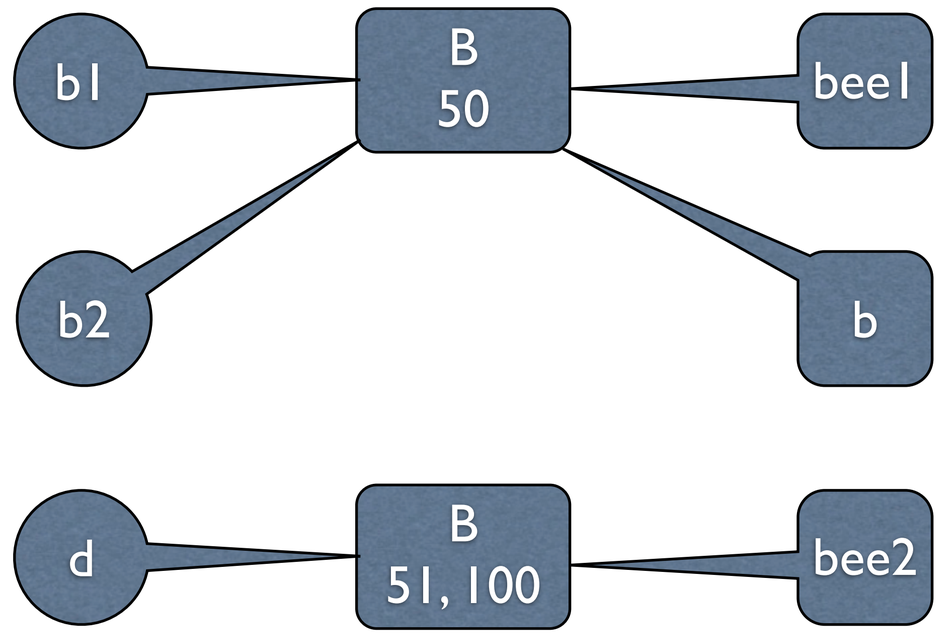


```
public class T {

   /* basic reference operations */

  public static void main(String[] args) {
     B b1 = new B(50);
     b1.print("b1{50} ");
     D d = new D(50,100);
     d.print("d{51, 100}");
     B b2 = (B) d;
     b2.print("b2{51,100} which is d");
     b2 = b1;
     b2.print("b2{50} which is b1 again");
     b1.print("b1{50} before xchange");
     d.print("d{51,100} before xchange");
     xchangeWrong(b1, (B) d);
     b1.print("b1{51} after xchange");
     d.print("d{51,100} after xchange");
  }
```

```
  public static void xchangeWrong(B bee1, B bee2) {
       B b = bee1;
       bee1.print("bee1 before xchange");
       bee2.print("bee2 before xchange");
       bee1 = bee2;
       bee2 = b;
       bee1.print("bee1 after xchange");
       bee2.print("bee2 after xchange");
    }
}
```

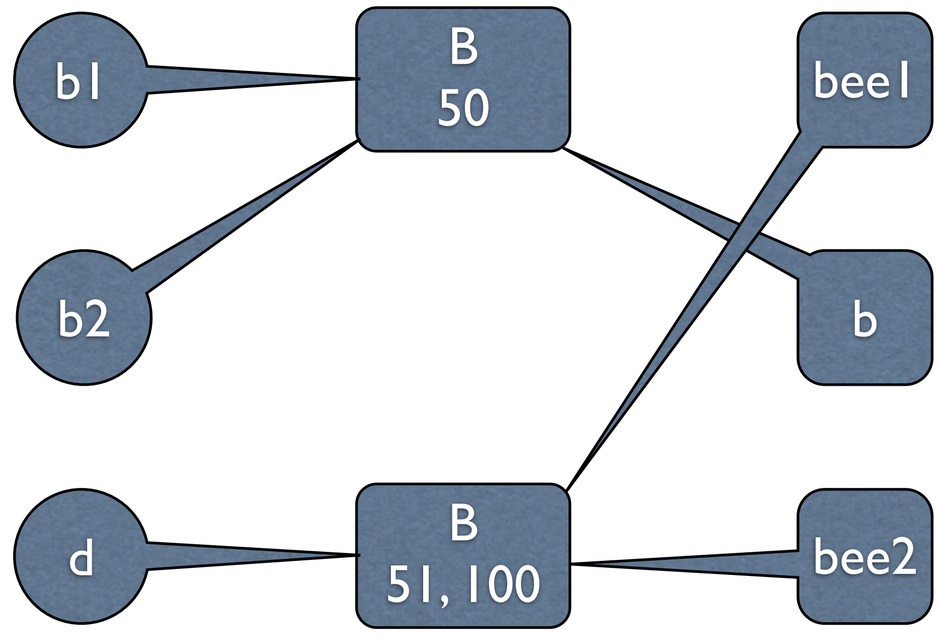


```
public class T {

   /* basic reference operations */

  public static void main(String[] args) {
     B b1 = new B(50);
     b1.print("b1{50} ");
     D d = new D(50,100);
     d.print("d{51, 100}");
     B b2 = (B) d;
     b2.print("b2{51,100} which is d");
     b2 = b1;
     b2.print("b2{50} which is b1 again");
     b1.print("b1{50} before xchange");
     d.print("d{51,100} before xchange");
     xchangeWrong(b1, (B) d);
     b1.print("b1{51} after xchange");
     d.print("d{51,100} after xchange");
  }
```

```
  public static void xchangeWrong(B bee1, B bee2) {
       B b = bee1;
       bee1.print("bee1 before xchange");
       bee2.print("bee2 before xchange");
       bee1 = bee2;
       bee2 = b;
       bee1.print("bee1 after xchange");
       bee2.print("bee2 after xchange");
    }
}
```

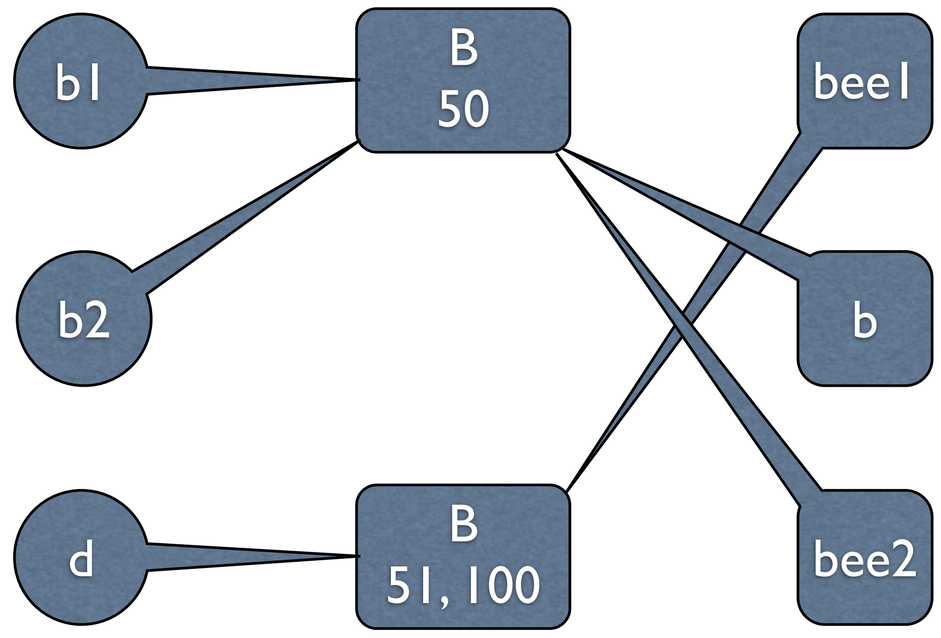


```
public class T {

   /* basic reference operations */

  public static void main(String[] args) {
     B b1 = new B(50);
     b1.print("b1{50} ");
     D d = new D(50,100);
     d.print("d{51, 100}");
     B b2 = (B) d;
     b2.print("b2{51,100} which is d");
     b2 = b1;
     b2.print("b2{50} which is b1 again");
     b1.print("b1{50} before xchange");
     d.print("d{51,100} before xchange");
     xchangeWrong(b1, (B) d);
     b1.print("b1{51} after xchange");
     d.print("d{51,100} after xchange");
  }
```

```
  public static void xchangeWrong(B bee1, B bee2) {
       B b = bee1;
       bee1.print("bee1 before xchange");
       bee2.print("bee2 before xchange");
       bee1 = bee2;
       bee2 = b;
       bee1.print("bee1 after xchange");
       bee2.print("bee2 after xchange");
    }
}
```

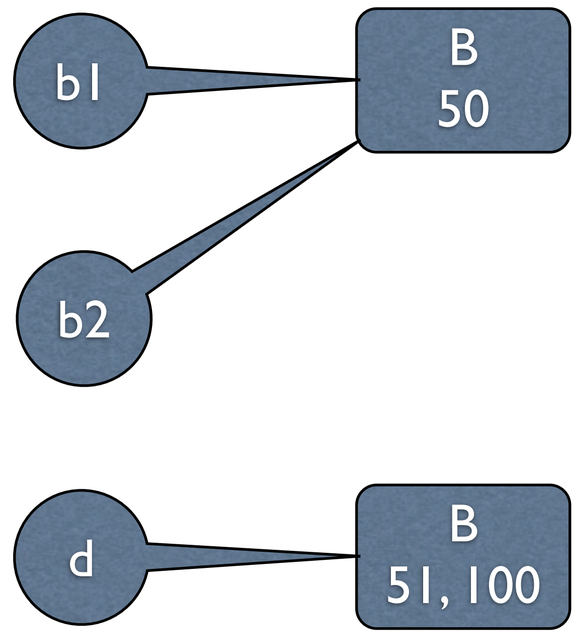


```
public class T {

   /* basic reference operations */

  public static void main(String[] args) {
     B b1 = new B(50);
     b1.print("b1{50} ");
     D d = new D(50,100);
     d.print("d{51, 100}");
     B b2 = (B) d;
     b2.print("b2{51,100} which is d");
     b2 = b1;
     b2.print("b2{50} which is b1 again");
     b1.print("b1{50} before xchange");
     d.print("d{51,100} before xchange");
     xchangeWrong(b1, (B) d);
     b1.print("b1{51} after xchange");
     d.print("d{51,100} after xchange");
  }
```

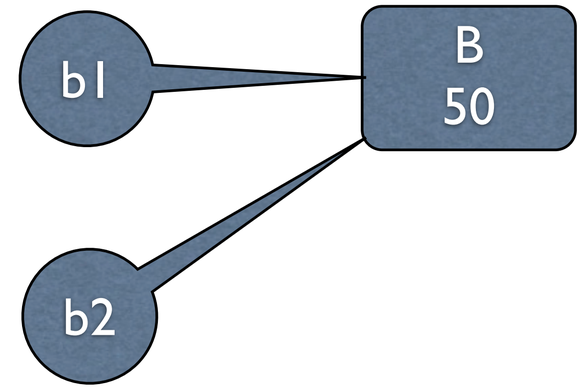
```
  public static void xchangeWrong(B bee1, B bee2) {
       B b = bee1;
       bee1.print("bee1 before xchange");
       bee2.print("bee2 before xchange");
       bee1 = bee2;
       bee2 = b;
       bee1.print("bee1 after xchange");
       bee2.print("bee2 after xchange");
    }
  }
```

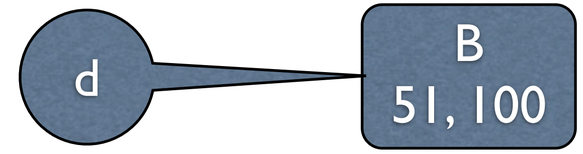


```
public class T {

   /* basic reference operations */

   public static void main(String[] args) {
      B b1 = new B(50);
      b1.print("b1{50} ");
      D d = new D(50,100);
      d.print("d{51, 100}");
      B b2 = (B) d;
      b2.print("b2{51,100} which is d");
      b2 = b1;
      b2.print("b2{50} which is b1 again");
      b1.print("b1{50} before xchange");
      d.print("d{51,100} before xchange");
      xchangeWrong(b1, (B) d);
      b1.print("b1{51} after xchange");
      d.print("d{51,100} after xchange");
   }
```






```
b1{50} , B object 50
d{51, 100} A object 51, 100
b2{51,100} which is d D object 51, 100
b2{50} which is b1 again, B object 50
b1{50} before xchange, B object 50
d{51,100} before xchange D object 51, 100
bee1 before xchange, B object 50
bee2 before xchange D object 51, 100
bee1 after xchange D object 51, 100
bee2 after xchange, B object 50
b1{50} after xchange, B object 50
d{51,100} after xchange D object 51, 100
```

What if you want to change a primitive value?

```
import java.lang.Integer;

public class Int {

   int val;

   public Int( ) {
      val = 0;
   }

   public Int(int i) {
      val = i;
   }

   public int get( ) {
      return val;
   }

   public void set(int i) {
      val = i;
   }

   public String toStr( ) {
      return Integer.toString(val);
   }
}
```

```
public class T {

   /* basic reference operations */

   public static void main(String[] args) {
      int i = 4;
      System.out.println("i: "+i);
      foo(i);
      System.out.println("i: "+i);
      i = fooR(i);
      System.out.println("i returned: "+i);

      Int ii = new Int(4);
      System.out.println("ii: "+ii.toStr( ));
      fool(ii);
      System.out.println("ii: "+ii.toStr( ));
   }

   public static void foo(int i) {
      i++;
   }

   public static void fool(Int j) {
      j.set(j.get()++);
   }

   public static int fooR(int i) {
      return 5;
   }
}
```

```
public class T {

    /* basic reference operations */

   public static void main(String[] args) {
      int i = 4;
      System.out.println("i: "+i);
      foo(i);
      System.out.println("i: "+i);
      i = fooR(i);
      System.out.println("i returned: "+i);

      Int ii = new Int(4);
      System.out.println("ii: "+ii.toStr( ));
      fool(ii);
      System.out.println("ii: "+ii.toStr( ));
   }
```

i: 4

i: 4

i returned: 5

ii: 4

ii: 5

```
public static void foo(int i) {
   i++;
}

public static void fool(Int j) {
   j.set(j.get()++);
}

public static int fooR(int i) {
   return 5;
  }
}
```

To return a value

- return it as a function return value
- pass in a reference by value, and change what the reference is pointing to For primitives this requires wrapping them in an object (as we did with Int).