Procedure specifications

```
/** Javadoc comment */
<method header> {
    …
}
```

```
/** Set the title to t. */
public void setTitle(String t)
    {...}
```

1. The spec explains what each parameter is for (so, it must mention all of them).

2. The spec is a *command* to do something.

```
setTitle("I want peace");
```
Set the title to "I want peace";

Function specifications

```
/** Return true if a, b, and c are
    in ascending order. */
public static boolean areAscending(
    int a, int b, int c) {
    return a < b && b < c;
}
```

A function call produces a value.

The function spec should say what the function-call value equals.

Constructor specifications

```
/** Constructor: a new instance with
    chapter number n, chapter title
    t, and previous chapter null
 */
public Chapter(int n, String t)
    {...}
```

Good specifications

• *Written before the method body*

• *Accurate and complete*

• *Include preconditions —constraints on the parameters that must be satisfied in a call, constraints that the caller must be aware of.*

```
/** = the square root of r.
    Precondition: r >= 0. */
public double sqrt(double r)
    { … }
```

Changing the spec

1. Change the specification to say what the method will now do.
2. Change the body to keep the specification accurate.

```java
/** = "a, b, and c are in non-
descending order". */
public boolean areAscending(
        int a, int b, int c) {
    return a <= b && b <= c;
}
```