Autoboxing

Up through version 4 of Java, an assignment like

```java
Integer b;
b= 25;
int i;
i= b;
```

were illegal, because the types of the variable and expression did not match—one is an `int` and the other is class `Integer`.

In version 5 of Java, **autoboxing** was introduced, which makes such assignments legal. In the first assignment, `b= 25;` the integer 25 is automatically wrapped in a new instance of class `Integer`; it is as if the programmer had written

```java
b = new Integer(25);
```

In the second assignment, `i= b;` since an `int` value is needed, the int value is automatically extracted from `Integer b`; it is as if the programmer had written

```java
i = b.intValue();
```

Autoboxing is a convenience for the programmer, allowing them to shorten their code.

Autoboxing is done with all primitive types and their corresponding wrapper classes. For example, here is autoboxing of a character:

```java
Character c= 'g';
```

Moreover, the autoboxing can be carried out in places other than assignments.

We give the rule for autoboxing for type `int` and leave to you to write the rule for other primitive types.

**Autoboxing rule for int:** If an expression `e` (say) of type `int` appears in a position in which an object of class `Integer` is expected (or possible), the expression `new Integer(e)` is used in place of `e`.

Here is an example of the use of this rule. The parameter of function isFive,

```java
/** = "obj is 5". */
public static boolean isFive(Integer obj) {
    return obj.equals(new Integer(5));
}
```

is of type `Integer`. Yet, in a call of it, we can put a value of type `int`.

`AutoboxDemo.isFive(5)`

There is an unboxing rule also—a rule for extracting a value from a wrapper class object. Here is the rule for class `Integer`; you can generalize it to the other wrapper classes yourself.

**Autounboxing rule for Integer:** If an expression `e` (say) of type `Integer` appears in a position in which an expression of type `int` is expected, then the expression `e.intValue()` is used in place of `e`.

Here is an example of the use of this rule. The parameter of function isFour,

```java
/** = "c is 4". */
public static boolean isFour(int c) {
    return c == 4;
}
```

We can call `isFour` using the call `AutoboxDemo.isFour(new Integer(4))`, and the `int` value is extracted from the argument expression.