

## Program 7

CS 264  
Fall 2006  
Craig A. Rich

In this program, we will write Jasmin source code that sorts three-dimensional boxes by volume. Implement an object class named `Box` whose instances represent boxes having height, width and length. The `Box` class should implement the following Java class specification:

```
public class Box implements Comparable {  
  
    private long height, width, length;  
  
    public Box(long height, long width, long length) {...}  
    public long volume() {...}  
    public int compareTo(Object box) {...}  
  
    public static void main(String[] arguments) {...}  
}
```

1 Edit a Jasmin source code file named `Box.j` that implements the `Box` specification, subject to the following implementation notes:

- The class variables represent the height, width and length of a box.
- The class variables are set using the `Box` constructor, which is named `<init>` in Jasmin source code. Note that the `Box` constructor must first invoke the constructor of its parent class `Object`.
- The `volume` method returns the volume (height times width times length) of a box.
- The `compareTo` method returns -1, 0, or 1, depending on whether the volume of this box is less than, equal to, or greater than the volume of the argument box, respectively. Note that the `compareTo` method implements the only method specified in the `Comparable` interface. This guarantees that an array of `Boxes` can be conveniently sorted using the built-in static method `Arrays.sort(Object[] array)` in the package `java.util`, which applies to any array of `Comparable` objects. Also note that to obtain the volume of the argument box, it must first be cast from type `Object` to type `Box` using the `checkcast` instruction.
- The `main` method reads from the standard input stream and prints to the standard output stream. The input contains a nonnegative integer  $n \geq 0$  representing the number of boxes, followed by  $3n$  positive integers representing the height, width and length (in that order) of the  $n$  boxes. `main` should construct  $n$  boxes representing those in the input, store references to the boxes in an array of length  $n$ , sort the array (using the built-in method `Arrays.sort`), then print the volumes of the  $n$  boxes in increasing order.
- Use the access permissions and properties (e.g. **public**, **private**, **static**) and the argument and return types shown in the specification.

- 2 Assemble and run the program on the input file named `Box.in` shown in the sample input below, and capture the output in a file named `Box.out`. Turn in printed copies of the Jasmin source code file `Box.j` and the output file `Box.out`.

```
% java Box < Box.in > Box.out
```

*Sample Input*

```
5
145 472 812
827 133 549
381 371 900
271 389 128
718 217 491
```

*Sample Output*

```
13493632
55573280
60385059
76500746
127215900
```