**Exercises:**

**Object-Oriented Programming:**

**Advanced Capabilities**

Make a new Eclipse project called oop-advanced-project or similar.

**1.** Make a CarSale class to represent the sale of a car. The class should represent the car name

(model), the list price, the discount in percent, and the final cost (calculated from the list price after

the discount is applied). To keep your code shorter, you can have getter but not setter methods for

each of those properties. Also put in a useful toString method. Make a few instances and print them

out.

Note: if you don’t understand @Override, just skip it for now. In the next lecture, we will explain it

more and see a situation where using it is critical. Also, remember that once you have instance vari-

ables, Eclipse can create the getters, setters (not needed here), and constructor for you. Use the

Source menu to tell Eclipse to insert this code.

**2.** Make a PaperclipSale class to represent the sale of a set of boxes of certain types of paper clips.

The class should represent the color of the clips, the per-box price, the number of those boxes being

sold, and the final cost. Also put in a useful toString method. Make a few instances and print them

out.

**3.** Make a static method called cheapest that, given an array of*mixed* CarSale and PaperclipSale

objects, will return the item with the lowest cost. (Return null if given an empty array. Don’t worry

about the possibility that the array might have two entries with the same cost.) Where is the best

place to put this method? Test the method.

Question to ponder: why was it important that your classes had meaningful toString methods?

**4.** Make a static method called totalCost that, given an array of mixed CarSale and PaperclipSale

objects, will return the total cost of all elements in the array. (Return 0 if given an empty array.) Test

the method.

**5.** Make a Coin enum with instances named HEADS and TAILS.

**6.** Make a static flip method that returns Coin.HEADS and Coin.TAILS with equal probability. Where

is the best place to put this method? Make a test case where you call flip multiple times and print

out the result each time.