11.6 Exercises

Exercises

1. Which of the following are examples of subtype polymorphism?
   - [ ] Having a `sqrt(int)` and `sqrt(double)` method in the same class.
   - [ ] Having a `Dog` override the `makeSound` method that it inherits from `Animal`.
   - [ ] Creating a class that implements the built-in `Comparable` interface.
2. How would you compare two strings alphabetically in Java?
3. Suppose we correctly define a `Comparator` class called `SixComparator` for integers that compares them based on the number of `6` s they contain. What will the code `(new SixComparator()).compare(12345678, 45666678)` return?
4. What is the main difference between the `Comparable` and `Comparator` interfaces?

Solutions

Problem 1

Having a `Dog` override the `makeSound` method that it inherits from `Animal`.

Overriding `makeSound` allows us to have different implementations of the same method via subtypes.

Creating a class that implements the built-in `Comparable` interface.

Implementing the Comparable interface involves overriding Comparables' methods, allowing for differing behavior of the same method across types.

Problem 2

`s1.compareTo(s2)`

Problem 3

It will return some negative number, since `12345678` has less `6`'s than `45666678`. Note that there is no guarantee of what the negative number's value is, only that it is less than 0.

Problem 4

An object that implements `Comparable` can compare another object to itself, whereas a Comparator compares two objects other than itself.

A good way to remember this is that a `Comparable` has an inherent property of being *able to be compared*, while a `Comparator` is an external source of truth.

Previous
11.5 Chapter Summary

Next
12. Inheritance IV: Iterators, Object Methods

Last updated 1 year ago