Data Structures and Algorithms in Java™

Sixth Edition

Michael T. Goodrich
Department of Computer Science
University of California, Irvine

Roberto Tamassia
Department of Computer Science
Brown University

Michael H. Goldwasser
Department of Mathematics and Computer Science
Saint Louis University

Study Guide: Hints to Exercises

WILEY

Chapter 2 Object-Oriented Design

Hints

Reinforcement

R-2.1) Think of applications that could cause a death if a computer failed.

R-2.2) Consider an application that is expected to change over time, because of changing economics, politics, or technology.

R-2.3) Consider the File or Window menus.

R-2.4) You can make the change and test the code.

R-2.5) You can make the change and test the code.

R-2.6) Your program should output 42, which Douglas Adams considers to be the answer to the ultimate question of life, universe, and everything.

R-2.7) A long value can be no larger than $2^{63} - 1$.

R-2.8) Code up an example and see what the compiler says.

R-2.9) Think about what happens when a new instance of class Z is created and when methods of class Z are called.

R-2.10) Think about code reuse.

R-2.11) Review the section about casting in an inheritance hierarchy, and recall that an object behaves according to what it actually is, not what it is called.

R-2.12) Review the definition of inheritance diagram, and begin your drawing with Object as the highest box.

R-2.13) Casting in an inheritance relationship can only move up or down the hierarchy.

R-2.14) You don't need to declare the array, just show how to use an exception try-catch block to reference it.

R-2.15) Reread the section on throwing exceptions.

Creativity

C-2.16) Create a separate class for each major behavior.

C-2.17) Try to use variables and conditions that are impossible, but the dependence on their values requires logical reasoning that the compiler writers did not build into their compiler.

C-2.18) You will need to maintain some additional state information.

C-2.19) Keep track of how much has been paid during the current month.

C-2.20) Don't forget you can use getBalance( ) as well.

C-2.21) You need to use the super keyword in *B* and *C*.

C-2.22) Recall the rule about inheritance in Java.

C-2.23) Can you determine a missing entry of a Fibonacci sequence if you are given the number immediate before it and after it?

C-2.24) Use the code from the website as a starting point.

C-2.25) Replace each use of type **long** with the generic parameter type T.

C-2.26) Use the sqrt method in the java.lang.Math class.

C-2.27) Go to the java.com website to review the BigInteger class.

C-2.28) Use three different classes, for each of the actors, and provide methods that perform their various tasks, as well as a simulator engine that performs the periodic operations.

C-2.29) If you have not had calculus, you can look up the formula for the first derivative of a polynomial on the Internet.

Projects

P-2.30) You don't have to use GUI constructs; simple text output is sufficient, say, using X's to indicate the values to print for each bar (and printing them sideways).

P-2.31) When a fish dies, set its array cell back to **null**.

P-2.32) Use random number generation for the strength field.

P-2.33) Create a separate class for each major behavior. Find the available books on the Internet, but be sure they have expired copyrights.

P-2.34) Lookup the formulas for area and perimeter on the Internet.

P-2.35) You need some way of telling when you have seen the same word you have before. Feel free to just search through your array of words to do this here.

P-2.36) While not always optimal, you can design your algorithm so that it always returns the largest coin possible until the value of the change is met.