Programming Assignment on Chapter 8

•• E8.1 Implement the Coin class described in Section 8.2. Modify the CashRegister class so that coins can be added to the cash register, by supplying a method
void receivePayment(int coinCount, Coin coinType)
The caller needs to invoke this method multiple times, once for each type of coin that is present in the payment.

•• E8.2 Modify the giveChange method of the CashRegister class so that it returns the number of coins of a particular type to return:
int giveChange(Coin coinType)
The caller needs to invoke this method for each coin type, in decreasing value.

• E8.3 Real cash registers can handle both bills and coins. Design a single class that expresses the commonality of these concepts. Redesign the CashRegister class and provide a method for entering payments that are described by your class. Your primary challenge is to come up with a good name for this class.

•• E8.4 As pointed out in Section 8.2.3, the Scanner.next method is a mutator that returns a value. Implement a class Reader that reads from System.in and does not suffer from that shortcoming. Provide methods
public boolean hasMoreElements() // Checks whether there is another element
public String getCurrent() // Yields the current element without consuming it
public void next() // Consumes the current element

•• E8.5 Reimplement the BankAccount class so that it is immutable. The deposit and withdraw methods need to return new BankAccount objects with the appropriate balance.

• E8.6 Reimplement the Day class of Worked Example 2.1 to be mutable. Change the methods addDays, nextDay, and previousDay to mutate the implicit parameter and to return void. Also change the demonstration program.

•• E8.7 Write static methods
• public static double cubeVolume(double h)
• public static double cubeSurface(double h)
• public static double sphereVolume(double r)
• public static double sphereSurface(double r)
• public static double cylinderVolume(double r, double h)
• public static double cylinderSurface(double r, double h)
• public static double coneVolume(double r, double h)
• public static double coneSurface(double r, double h)
that compute the volume and surface area of a cube with height h, sphere with radius r, a cylinder with circular base with radius r and height h, and a cone with circular base with radius r and height h. Place them into a class Geom etry. Then write a program that prompts the user for the values of r and h, calls the six methods, and prints the results.

•• E8.8 Solve Exercise •• E8.7 by implementing classes Cube, Sphere, Cylinder, and Cone. Which approach is more object-oriented?

••• E8.9 Modify the application of How To 7.1 so that it can deal with multiple students. First, ask the user for all student names. Then read in the scores for all quizzes, prompting for the score of each student. Finally, print the names of all students and their final scores. Use a single class and only static methods.

••• E8.10 Repeat Exercise ••• E8.9, using multiple classes. Provide a GradeBook class that collects objects of type Student.

••• E8.11 Write methods
public static double perimeter(Ellipse2D.Double e);
public static double area(Ellipse2D.Double e);
that compute the area and the perimeter of the ellipse e. Add these methods to a class Geometry. The challenging part of this assignment is to find and implement an accurate formula for the perimeter. Why does it make sense to use a static method in this case?

•• E8.12 Write methods
public static double angle(Point2D.Double p, Point2D.Double q)
public static double slope(Point2D.Double p, Point2D.Double q)
that compute the angle between the x-axis and the line joining two points, measured in degrees, and the slope of that line. Add the methods to the class Geometry. Supply suitable preconditions. Why does it make sense to use a static method in this case?

••• E8.13 Write methods
public static boolean isInside(Point2D.Double p, Ellipse2D.Double e)
public static boolean isOnBoundary(Point2D.Double p, Ellipse2D.Double e)
that test whether a point is inside or on the boundary of an ellipse. Add the methods to the class Geometry.

•• E8.14 Using the Picture class from Worked Example 6.2, write a method
public static Picture superimpose(Picture pic1, Picture pic2)
that superimposes two pictures, yielding a picture whose width and height are the larger of the widths and heights of pic1 and pic2. In the area where both pictures have pixels, average the colors.

•• E8.15 Using the Picture class from Worked Example 6.2, write a method
public static Picture greenScreen(Picture pic1, Picture pic2)
that superimposes two pictures, yielding a picture whose width and height are the larger of the widths and heights of pic1 and pic2. In the area where both pictures have pixels, use pic1, except when its pixels are green, in which case, you use pic2.

• E8.16 Write a method
public static int readInt(
Scanner in, String prompt, String error, int min, int max)
that displays the prompt string, reads an integer, and tests whether it is between the minimum and maxi mum. If not, print an error message and repeat reading the input. Add the method to a class Input.

•• E8.17 Consider the following algorithm for computing xn for an integer n. If n < 0, xn is 1/x–n. If n is positive and even, then xn = (xn/2)2. If n is positive and odd, then xn = xn–1

Å~ x. Implement a static method double intPower(double x, int n) that uses this algorithm. Add it to a class called Numeric.

•• E8.18 Improve the Die class of Chapter 6. Turn the generator variable into a static variable so that all dice share a single random number generator.

•• E8.19 Implement Coin and CashRegister classes as described in Exercise •• E8.1. Place the classes into a package called money. Keep the CashRegisterTester class in the default package.

• E8.20 Place a BankAccount class in a package whose name is derived from your e-mail address, as described in Section 8.6. Keep the BankAccountTester class in the default package.

•• E8.25 Add a method ArrayList<Double> getStatement() to the BankAccount class that returns a list of all deposits and withdrawals as positive or negative values. Also add a method
void clearStatement() that resets the statement.

•• E8.26 Implement a class LoginForm that simulates a login form that you find on many web pages. Supply methods
public void input(String text)
public void click(String button)
public boolean loggedIn()
The first input is the user name, the second input is the password. The click method can be called with arguments "Submit" and "Reset". Once a user has been successfully logged in, by supplying the user name, password, and clicking on the submit button, the loggedIn method returns true and further input has no effect. When a user tries to log in with an invalid user name and password, the form is reset. Supply a constructor with the expected user name and password.

•• E8.27 Implement a class Robot that simulates a robot wandering on an infinite plane. The robot is located at a point with integer coordinates and faces north, east, south, or west. Supply methods
public void turnLeft()
public void turnRight()
public void move()
public Point getLocation()
public String getDirection()
The turnLeft and turnRight methods change the direction but not the location. The move method moves the robot by one unit in the direction it is facing. The get Direction method returns a string "N", "E", "S", or "W".