3 Designing Methods

Portfolio Problems

Work out as complete programs the following exercises from the textbook. You need not work out all the methods, but make sure you stop only when you see that you really understand the design process.

Problems:

1. Problem 10.2 on page 97
2. Problem 10.5 on page 105
3. Problem 14.1 on page 140
4. Problem 14.7 on page 144
5. Problem 15.10 on page 176

Pair Programming Assignment

3.1 Problem

This problem refers to the classes that represent shapes from Assignment 2.

A. Convert the data definitions for the classes that represent shapes so that they use the `IColor` class hierarchy to represent the colors.

B. Define a method `toPosn` that produces an instance of the class `Posn` from this `CartPt` cartesian point.

C. Design the method `centerOf()` that produces the `CartPt` center of this shape as follows:

- for a circle or a disk the center is already given as one of its fields
- the center of the rectangle is halfway between the left and the right edge, and halfway between the top and the botton edge
- the center of the line is halfway between the two ends of the line
- the center of a combo shape if halfway between the centers of the top and the bottom shape

D. Design the method `contains` that determines whether this shape contains the given `CartPt` point.

   Note: To do this correctly for the class `Line` requires a bit of geometrical computation. You are allowed to substiute, in this case, a method that produces `false` for all inputs.

E. Define the method `distanceTo` that computes the distance between the centers of two shapes.

F. Define the method `drawShape` that draws this shape in the given `Canvas`.

3.2 Problem

This problem builds on the data representation for cities and on the functions you have designed for the first assignment. We change the data slightly — all latitudes and longitudes are rounded to the nearest integer and will be represented as an `int` type of data.

A. We would like to draw the cities on a map of the USA. Suppose our Canvas is 300 pixels wide and 300 pixels tall. We want to convert the latitude and longitude representation of the location into a `Posn` on this map. Define the method `toPosn` that produces a `Posn` that represents `this` location on the given Canvas.

   Remember that the coordinates for drawing on the `Canvas` have the origin `(0, 0)` in the top left corner.

   Assume that the latitude and longitude lines are parallel to the Canvas boundaries (*parallel projection*). Assume that

   - the left edge of the Canvas is the 125 degrees of longitude (in the Western hemisphere)
     Note: By mistake we omitted the western states in Assignment 1
   - the right edge is 65 degrees of longitude (in the Western hemisphere)
   - the top of the Canvas is 50 degrees of latitude (in the Northern hemisphere)
   - the bottom is 20 degrees of latitude (in the Northern hemisphere)

   Though it may not be politically correct in this election year, we focus only on the contiguous continental states (omitting the beautiful states of Alaska and Hawaii).

B. Define the method `distance` for the class `City` that computes the distance (in miles) between two cities. Estimates tell us that one degree of longitude (at our latitude) is approximately 55 miles and one degree of latitude is approximately 70 miles. (Feel free to make a more accurate estimates. If you do so, include a comment explaining how you arrived at your estimates.)

   Note: Make sure you follow the *one task — one method* rule and delegate tasks to the classes that should be responsible for handling them!

C. Define the method `totalDistance` that computes the length of a trip on which we visit all cities in a list of cities in the order in which they appear in the list.

D. Design the method `drawCity` that will draw the city as a small red dot at the location given by its `toCartPt` method.

E. In the class `City` design the method `lineTo` that draw a black line from this city to the given one.

F. Design the method `drawPath` for the class hierarchy that represents a list of cities that draws a path traversed as we travel through the cities in this list.

3.3 Problem

Creative Project

This week you will program some of the game actions and interactions.

A. For each object that appears at a new location as the time ticks, or which changes its appearance as the time ticks (and which is represented by the class `MyObject`) design the method `myObjectOnTick` that produces a new instance of `MyObject` as it should appear after one tick of the clock.

B. For each object that appears at a new location in response to some key event, or which changes its appearance in response to some key event (and which is represented by the class `MyObject`) design the method `myObjectOnTick` that produces a new instance of `MyObject` as it should appear after one tick of the clock.

C. For each object used in your game (which is represented by the class `MyObject`) design the method `drawMyObject` that draws this object on the given `Camvas`.

D. Design the method `drawMyWorld` that draws the entire game scene for your game on the given `Canvas`.

E. Show examples of a drawing of your *world* at the start of the game, at some end of your game, and at some points in the middle.

Note: Make sure your code is neat — short methods that do one task only. **Design helper methods** when needed, delegate to the appropriate classes the work they are responsible for. Next week we will finish the game and your grade there will depend greatly on the overall design of the game.

Do not forget tests for all methods. *This is non-negotiable, as the lack of simple unit tests has been shown to cause millions of dollars of damage, as well as loss of life.*

Add the lines

```
import draw.*;
import geometry.*;
import colors.*;
```

at the beginning of every program that deals with drawing on the `Canvas`, uses the color data in the `IColors` collection, or refers to instances of the `Posn` class.