2 Self-Referential Data; Designing Methods — Part 1

We continue with the the theme of the photo images. Our collections of images surely contains more than one picture. We will first define a list of images.

Once we have the data that represents our pictures, we would like to be able to ask questions about the data. To do so, we design methods in the classes that represent our data.

2.1 Self-Referential Data

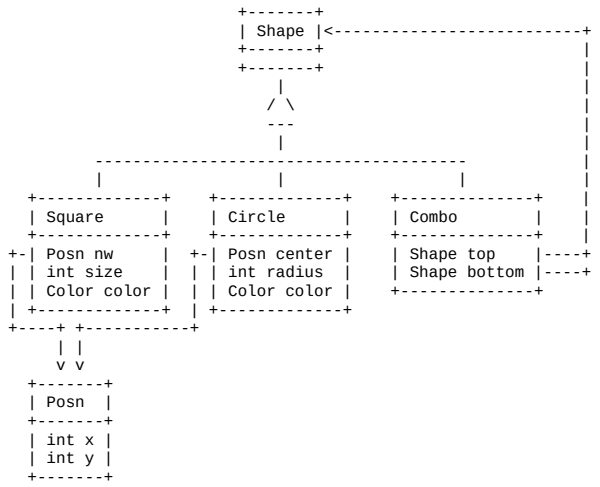
1. The HtDP data definition for a list of photos is:

   ```
   ;; A ListOfPhotos is one of
   ;; — empty
   ;; — (cons Photo ListOfPhotos)
   ```

   Design the Java classes to represent a list of *Photo*s. Remember to make examples of data.

   Note: This is the last time we will remind you to make examples of data.

2. The textbook uses classes that represent geometric shapes to illustrate the design of classes. Define Java classes that correspond to the following class diagram.



2.2 Methods for Simple Classes and Classes with Containment

For this series of exercises use the classes from Lab 1 that represented photo images and included the date and time information.

Below is an example of the design of a method that computes the number of pixels in a photo image:

- Step 1: Problem analysis and data definition.

  The method deals with *Photo*s and so it needs to be defined in the class *Photo*. Each instance of a *Photo* has all the information we need to solve the problem - we do not need any additional data to be given. The result is an integer.

  We will use the following data in our examples. For your work add at least one more instance of each class.

```
// Examples for the class ClockTime
ClockTime ct1 = new ClockTime(9, 50, "pm");
ClockTime ct2 = new ClockTime(11, 30, "am");
ClockTime ct3 = new ClockTime(9, 50, "am");


// Examples for the class Date
Date d1 = new Date(2006, 9, 23);
Date d2 = new Date(2006, 11, 7);
Date d3 = new Date(2006, 9, 25);


// Examples for the class Photo
Photo river = new Photo("River", "jpeg", 3456, 2304, 3614571,
                        this.d1, this.ct3);
Photo mountain = new Photo("Mountain", "jpeg", 2448, 3264, 1276114,
                           this.d2, this.ct2);
Photo people = new Photo("People", "gif", 545, 641, 13760,
                         this.d2, this.ct1);
Photo icon = new Photo("PLTicon", "bmp", 16, 16, 1334,
                       this.d1, this.ct2);
```

- Step 2: The purpose statement and the header.

```
// to compute the number of pixels in this photo
int pixels(){...}
```

- Step 3: Examples.

```
people.pixels() ---> 349345
icon.pixels() ---> 256
```

- Step 4: The template.

  int pixels(){
  ... *this.name* ... −−− *String*
  ... *this.kind* ... −−− *String*
  ... *this.width* ... −−− *int*
  ... *this.height* ... −−− *int*
  ... *this.bytes* ... −−− *int*
  ... *this.date* ... −−− *Date*
  ... *this.time* ... −−− *ClockTime*

  We will only need *this.width* and *this.height*.

- Step 5: The method body.

  ```
  // to compute the number of pixels in this photo
  int pixels(){
     return this.width * this.height;
  }
  ```

- Step 6: Tests.

  ProfessorJ provides a special way of running the tests. A *check* expression

  check *test method invocation* expect *expected test result*

  produces the test result as a *boolean* value and all test results are reported in a separate display. The following code:

  ```
  // Tests for the method pixels:
  boolean testPixels = (check this.people.pixels() expect 349345) &&
                       (check this.icon.pixels() expect 256);
  ```

  shows the tests for our method.

1. Design the method that determines how long it will take to download this image, if we know the number of bytes we can download in one second.

2. Design the method that determines whether this picture was taken before another picture.

   Remember: One task, one method. Make each class responsible for its data.

3. In the class ClockTime design the method that advances the time by the given number of minutes. Use helper methods as needed.

There will be no quiz today. However, we give you a sample of what the quiz would have been. Take it home and make sure you can do the quiz in less than 10 minutes.

2.3 Methods for Unions

For this part use the data for camera shots (photos or videos) from the previous lab.

1. Design the method that determines how long it will take to download this shot, if we know the number of bytes we can download in one second.

2. Design the method that determines whether this shot will download faster than another one at the given download speed.