CS 273 Laboratory 11: Inheritance

This lab gives you experience with Java class inheritance. It has a total of 100 possible points.

Preliminaries

Create a folder named `lab11` in the `cs273` area of your network drive.

Go to the course website on Learning@UP and download the software for Lab 11 (`lab11.zip`). Unzip the software into the `lab11` folder you just created. This includes several Java files as well as other project-related files.

Open the project and run `FaceFrame`. It should display a window with 12 cartoonish faces that are identical except for size and (horizontal and vertical) scaling. Each is labeled (e.g., "sad", "jealous") that indicates the kind of face it is supposed to be. Right now all the faces are emotionless.

Examine the files on the screen:

- `FaceFrame.java` - an application that creates 12 faces (of 6 varieties) and displays them
- `Face.java` - the definition of a "Face" object. Study all the methods in this class. Note especially that there are methods (e.g., `pixelX`, `pixelY`, `distX`, `distY`) for computing coordinates relative to positions the current object's head (e.g., 20% of the way across the face). These will be handy when drawing facial features.
- `Polygon2.txt` - In addition to being in your `lab11` directory, a hardcopy of this file is at the end of this lab handout. The file is the documentation for a `Polygon2` class which is an extended, custom `Polygon` class, which , in addition to normal `Polygon` functionality, allows the following:
  - polygons that approximate a crescent-moon to be created
  - rotation
  - horizontal and vertical scaling

  Study this documentation so that you can use the `Polygon2` class effectively.

A `Face` object draws a face within a given bounding box. The face is drawn with:

- a pink head
- blue eyes
- a red rectangular mouth
- a brown triangular nose
- a splotch of yellow hair
- pink ears

There are actually a total of six face classes. In the application, two instances of each class are created and displayed:

- `Face`
- `HappyFace`
- `SadFace`
- `EmbarassedFace`
- `JealousFace`
- `GrouchoFace`

The reason that they all display the same face is that all are declared as subclasses of `Face`, and the starter code does not define each type of face to have any additional (or different) behavior.

Examine the *FaceDemo* located at Week 11 of the course website. It shows you approximately what you're going to try to make the `FaceFrame` look like at the end of this lab.

IMPORTANT: you should not modify `Face.java` or `FaceFrame.java` during this lab.

Laboratory

Part 1: Create a happy face

Edit `HappyFace.java` so that a `HappyFace` object acts just like a `Face` object except that the mouth is a smile rather than a simple rectangle. You should complete the `HappyFace` class using the following steps:

1. Create a `protected` method called `createMouth` that takes no parameters and *returns* a `Polygon2` object that is the form of a filled crescent that looks like a smile. (See the documentation for the `Polygon2` class with regard to the constructor and methods to scale and rotate the `Polygon2` object.) The object returned should fit in the face. Recall that the method call `pixelX(20)` returns the x-coordinate location that is 20% from the left edge of the face's bounding box.
2. Override the `drawMouth` method in the `Face` class, by defining a `protected` method called `drawMouth` in the `HappyFace` class that takes a `Graphics` object `g` as a parameter and returns nothing. The `drawMouth` method in the `HappyFace` class should set the color of the graphics pen to `this.mouthColor()`. Then the method should create a `Polygon2` variable and assign it to the result of calling `createMouth()`. Finally, the method should draw the mouth to the `Graphics` object `g` by calling `fillPolygon` with the mouth polygon as the argument.

checkpoint 1 (25 points): Show your lab instructor or assistant that the two happy faces now look as described above.

Part 2: Create a sad face

Edit `SadFace.java` so that it a `SadFace` behaves like a `HappyFace` except that:

- The mouth is a frown
    - Do this by making `SadFace` a subclass (derived class) of `HappyFace`, so you can override the `createMouth` method with one that creates a mouth in which the smile is rotated 180 degrees. To do this properly, you should use the `super` keyword.
- The color of the mouth is black
    - Do this by overriding the `mouthColor` method.
- The nose is a dark red oval
    - Do this by overriding the `drawNose` method.

checkpoint 2 (20 points): Show your lab instructor or assistant that the two sad faces now look as described above.

Part 3: Create a jealous face

Edit `JealousFace.java` so that a `JealousFace` object is drawn just like a `SadFace` object, except that:

- The centers of the eyes are green
- There is no hair

Use **subclassing** and **inheritance** whenever possible.

checkpoint 3 (10 points): Show your lab instructor or assistant that the two jealous faces now look as described above.

Part 4: Create an embarrassed face

Edit `EmbarassedFace.java` so that an `EmbarassedFace` object is drawn just like a `SadFace` object, except that:

- The face is red (i.e., the color of the head is red)
- The mouth is pink
- The white parts of the eyes are about three times as big

Again, use **subclassing** and **inheritance** whenever possible to avoid doing extra work.

checkpoint 4 (10 points): Show your lab instructor or assistant that the two embarrassed faces now look as described above.

If you have completed all above checkpoints, you have a grade of D (65) for this lab.

Part 5: Create a Groucho face

Edit `GrouchoFace.java` so that a `GrouchoFace` object is drawn just like a `HappyFace` object, except that:

- The hair is black
- The face has a wide, black mustache below the nose
    - Do this by overriding the `drawNose` method to include the original nose (call the super version) and then painting a mustache.
- The smile is tilted a few degrees
    - Do this by overriding `createMouth`.

Yet again, use **subclassing** and **inheritance** whenever possible.

<u>checkpoint 5 (15 points):</u> Show your lab instructor or assistant that the two Groucho faces now look as described above.

If you have completed all above checkpoints, you have a grade of B- (80) for this lab.

Part 6: Add glasses and a cigar to the Groucho face

Edit `GrouchoFace.java` so that a `GrouchoFace` includes eyeglasses and a cigar in the mouth.

Of course, you should use **subclassing** and **inheritance** whenever possible.

<u>checkpoint 6 (20 points):</u> Show your lab instructor or assistant that the two Groucho faces now have a cigar and glasses.

If you have completed all above checkpoints, you have a grade of A (100) for this lab.

Part 7: Finish up

Close all windows.

Log off. You're done!