COLBY COLLEGE
Department of Mathematics
and Computer Science

CS115 **Allen B. Downey**
Spring 1998

Assignment 5: Moiré patterns

Due: Wednesday 11 March

The purpose of this assignment is to use loops to automate repetitious computations.

Setup

1. Use the Roaster wizard to create a new project named Moire.π, and a new Applet class named Moire. Make the window size 400 x 400. Make sure that you put the new project in a folder by itself, preferably on the desktop. Also, once you have chosen a name for the project and the class, stick with it. Many of you have discovered that it is difficult to put a project back together if its files are scattered or if you try to change the names of things.

2. Write a method called `paintCircle` with the following prototype:

   ```
   public void paintCircle (Graphics g, int diameter, int width, int height)
   ```

   `width` and `height` are the width and height of the screen, as determined (in `paint`) by invoking `bounds`. `paintCircle` should draw a circle with the given diameter, in the center of the screen.

3. Test your program by invoking it several times from `paint`, with various diameters.

Draw a target

1. Write a method called `paintTarget` with the following prototype:

   ```
   public void paintTarget (Graphics g, int width, int height)
   ```

   Once again, `width` and `height` are the width and height of the screen. `paintTarget` should invoke `paintCircle` four times, creating circles with diameters 100, 200, 300 and 400.

2. In `paint`, don't invoke `paintCircle` directly. Invoke `paintTarget` instead.

Draw a better target

1. Change `paintTarget` so that instead of drawing 4 circles it draws 8, equally spaced.
2. Now make it draw 16.
3. Now make it draw 32. No, don't. At this point you should be thinking that this is a collosal waste of time. There must be a better way. And, of course, there is.

Iterate

1. Add variables to `paintTarget` named `smallest`, `biggest` and `skip`. `smallest` will be the diameter of the smallest circle, `biggest` will be the diameter of the biggest, and `skip` will be the distance between the circles. To start, make them 100, 400 and 50, respectively.
2. Add a variable named `diameter` and initialize it to `smallest`. Write a while loop that draws a circle with diameter `diameter`, increments `diameter` by `skip`, and continues until `diameter` is greater than `biggest`.
3. Test your program with various values of `smallest`, `biggest` and `skip`.

Draw a very fine target

1. An interesting thing starts to happen as `skip` gets small. Run your program with `smallest` = 2, `biggest` = 400, and `skip` = 2.

   Did you get what you expected? If you did, you know too much and you should go outside and play. For the rest of us, the unexpected coolness on the screen is called a Moiré pattern, and it is the result of a type of roundoff error called aliasing. For more information on Moiré patterns, see the following web pages:

   ```
   http://www.thepalace.com/jbum/pixmagic/galmoire.html
   http://www.exploratorium.edu/xref/exhibits/moire_patterns.html
   ```

2. You can make your target even cooler by changing the color using `setColor`.

Draw two overlapping targets

1. Probably the coolest Moiré pattern comes from two overlapping targets, with one shifted down and to the right one pixel. Within the same loop you used in the previous part, add a couple of lines that change the color and draw a second circle, with x and y coordinates that differ by one from the first circle's.

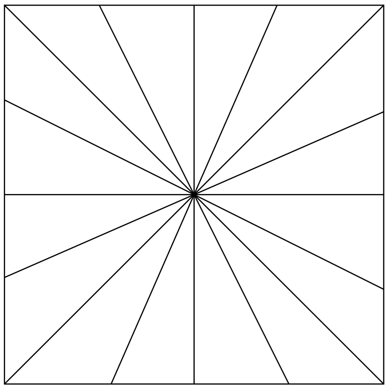
   HINT: inside the loop you will invoke setColor twice and drawOval twice.

Draw a radial Moiré pattern

1. Write a method with the following prototype:

   ```
   public void paintRadial (Graphics g, int width, int height)
   ```

2. This method should draw a pattern like the figure on the next page, except that the distance between the radial lines should be smaller. Small enough that you start to see Moiré patterns, that is.



And now for something completely different...

This part of the assignment is from the book, page 184.

1. Create a new project named Loop, and make it an Application.
2. Evaluate the expression

$$\sum_{i=1}^{10,000} \frac{1}{i} \tag{1}$$

   with two different `for` loops; first use

```
for (int i = 1; i <= 10000; i++)
```

   and then use

```
for (int i = 10000; i >= 1; i--)
```

3. Explain why the printed values are not equal. What is the more accurate value?
4. If you do not understand what is happening, try doing the computation on paper, adding only the first ten values, and using two-digit decimal math. In other words, round everything off to two significant digits at each step of the calculation. For example, $1 + 1/2 + 1/3 = 1.0 + 0.5 + .33 = 1.8$

What to turn in

1. Print your target-drawing code and write your name in the upper-right corner.
2. Print Loop.java and write your explanation on the back. Staple the two pages together.