Chris Piech Section #2

CS 106A April 18, 2018

Section Handout #2—Simple Java

Portions of this handout by Eric Roberts and Julia Daniel

**1. Mystery Calculation**

The following is code for an interactive console program that performs a type of calculation that is probably familiar. Examine the code. What is the role of the **SENTINEL** value? How do each of the four variables – **a**, **b**, **x**, and **y** – change over time? Overall, what common task does this program do?

**/\***

**\* File: MysteryCalculation.java**

**\* -----------------------------**

**\* It’s your job to figure out what this program does!**

**\*/**

**import acm.program.\*;**

**public class MysteryCalculation extends ConsoleProgram {**

**/\* What does this constant do? \*/**

**private static final int SENTINEL = -1;**

**public void run() {**

**int a = readInt("Enter a value for a: ");**

**int b = readInt("Enter a value for b: ");**

**int x = readInt("Enter a value for x: ");**

**while (x != SENTINEL) {**

**int y = a \* x + b;**

**println("Result for x = " + x + " is " + y);**

**x = readInt("Enter a value for x: ");**

**}**

**}**

**}**

2. The Fibonacci sequence

In the 13th century, the Italian mathematician Leonardo Fibonacci—as a way to explain the geometric growth of a population of rabbits—devised a mathematical sequence that now bears his name. The first two terms in this sequence, **Fib**(0) and **Fib**(1), are 0 and 1, and every subsequent term is the sum of the preceding two. Thus, the first several terms in the Fibonacci sequence look like this:

**Fib**(0) = 0

**Fib**(1) = 1

**Fib**(2) = 1 (0 + 1)

**Fib**(3) = 2 (1 + 1)

**Fib**(4) = 3 (1 + 2)

**Fib**(5) = 5 (2 + 3)

Write a program that displays the terms in the Fibonacci sequence, starting with **Fib**(0) and continuing as long as the terms are less than or equal to 10,000. Thus, your program should produce the following sample run:



This program should continue as long as the value of the term is less than or equal to the maximum value. To do this, you should use a **while** loop, presumably with a header line that looks like this:

**while (term <= MAX\_TERM\_VALUE)**

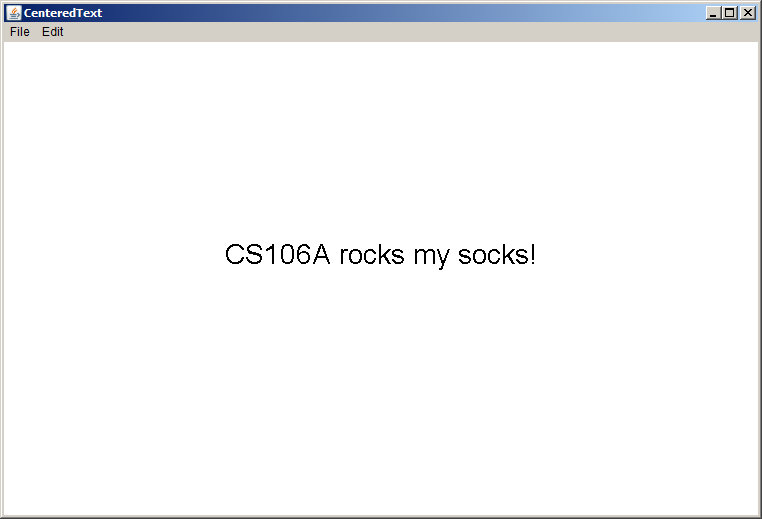
Note that the maximum term value is specified using a named constant. Your program should work properly regardless of the value of **MAX\_TERM\_VALUE**.

**3. Drawing Centered Text**

Your job is to write a **GraphicsProgram** that displays the text message (i.e., **GLabel**):

**CS106A rocks my socks!**

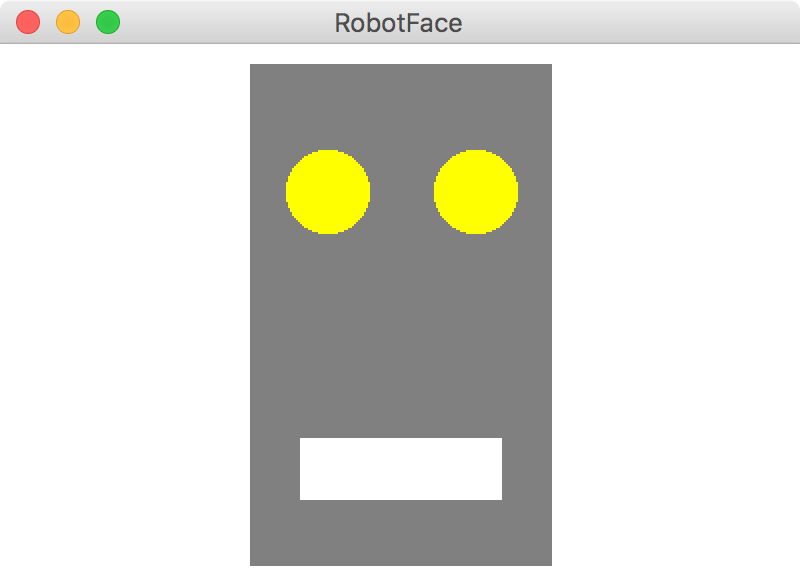
The text should be displayed in SansSerif 28-point font, and centered horizontally and vertically in the middle of the graphics window, looking something like this:



You can find the width of a label by calling label.getWidth() and the height it extends above the baseline by calling label.getAscent(). If you want to center a label, you need to shift its origin by half of these distances in each direction.

*Bonus:* If you wanted to add 10 labels to the window, all with the same font, size, and horizontal centering but with different y-coordinates, how might you organize your code?4. Drawing a face

Your job is to draw a robot-looking face like the one shown in the following sample run:



This simple face consists of four parts—a head, two eyes, and a mouth—which are arranged as follows:

• *The head.* The head is a big rectangle whose dimensions are given by the named constants **HEAD\_WIDTH** and **HEAD\_HEIGHT**. The head is gray.

• *The eyes.* The eyes should be circles whose radius in pixels is given by the named constant **EYE\_RADIUS**. The centers of the eyes should be set horizontally a quarter of the width of the head in from either edge, and one quarter of the distance down from the top of the head. The eyes are yellow.

• *The mouth.* The mouth should be centered with respect to the head in the *x*-dimension and one quarter of the distance up from the bottom of the head in the *y*-dimension. The dimensions of the mouth are given by the named constants **MOUTH\_WIDTH** and **MOUTH\_HEIGHT**. The mouth is white.

Finally, the robot face should be centered in the graphics window.