Assignment 2

Due. Oct. 14 (Tuesday), 23:59.

1. Write a program that asks the user for the radius of a circle and then computes the area of that circle (A) using the formula

$$A = \pi r^2$$

   Note that there is no "raise to power" operator in Java. Given the arithmetic operators you know Java has, how can you write an expression that achieves the desired result?

2. Write a program `FahrenheitToCelsius.java` that reads in a temperature in degrees Fahrenheit and returns the corresponding temperature in degrees Celsius. The conversion formula is

$$C = \frac{5}{9}(F - 32)$$

   The following is a sample run of the program:

```
This program converts Fahrenheit to Celsius.
Enter Fahrenheit temperature: 212
Celsius equivalent = 100.0
```

   If you write the program carelessly, the answer always comes out 0. What bug causes this behavior?

3. Write a dialog version of the above program and call it `FToCDialog.java`. The following is a dialog version of the console program `Add2Integers.java` (Figure 2-2, page 30). As you can see, it is almost identical to the console version, only that it is a subclass of **DialogProgram**. Figure 2-6, page 39 shows the output from the following dialog program. The style of interaction is quite different from the console program.

   The classes **ConsoleProgram** and **DialogProgram** define their own version of **println** and **readInt** so that they operate in the style appropriate to that class. Redefining an existing method so that it behaves differently from the implementation defined by its superclass is called **overriding** that method.

```
/*
 * File: Add2IntDialog.java
 * ------------------------
 * This program adds two integers and print their sum
 * using dialog boxes.
 *
 */

import acm.program.*;

public class Add2IntDialog extends DialogProgram {
```

```
   public void run() {
      println("This program adds two integers.");
      int n1 = readInt("Enter n1: ");
      int n2 = readInt("Enter n2: ");
      int total = n1 + n2;
      println("The total is " + total + ".");
   }
}
```

4. The `TicTacToeBoard` program shown in the text book Figure 2-12, p. 50, generates a tic-tac-toe board.

```
/*
 * File: TicTacToeBoard.java
 * -------------------------
 * This program draws a Tic-Tac-Toe board as an illustration
 * of the Gline class. This version uses explicit coordinate
 * values which makes the program difficult to extend or
 * maintain. In Chapter 3, you will learn how to use constants
 * and expressions to calculate these coordinate values.
 */

import acm.graphics.*;
import acm.program.*;

public class TicTacToeBoard extends GraphicsProgram {

   public void run() {
      add(new GLine(30, 60, 120, 60));
      add(new GLine(30, 90, 120, 90));
      add(new GLine(60, 30, 60, 120));
      add(new GLine(90, 30, 90, 120));
   }
}
```

The program, however, uses poor programming style because the coordinates for each of the lines are specified explicitly in the code. Rewrite this program so that it centers the board in the window and uses a single constant called `BOARD_SIZE` to define the height and width of the figure.