I have created a class in /home/bvz/gui/hw/hw2 called template.java that creates a basic JComponent object, adds it to a JFrame, and makes the JFrame visible. You can modify this class to fit your needs for the following problems. I have not over-specified the problems by telling you what the exact sizes of all the objects and windows are. Unless I specify precise sizes for windows or objects, use your best judgment.

1) Rewrite your thermometer application from hw1 so that it incorporates the following changes:

   a) ThermometerApplication should now take only three arguments, the minimum temperature, the maximum temperature, and the starting temperature. The label should be omitted.

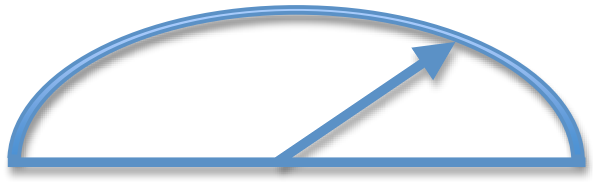
   b) the model uses the observer pattern described in class and the model supports the following methods:

      i) void setValue(int degrees)
      ii) int getMinValue()
      iii) int getMaxValue()
      iv) int getValue()

   c) use an Area object to create the filled bulb and stem portion of the thermometer in the original thermometer view.

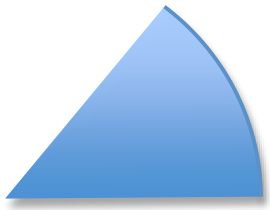
   d) add the following view(s), each in its own window:

      1) A temperature gauge with a chorded half oval and an arrow line that goes from the minimum value (right side of the gauge) to the maximum value (left side of the gauge). It should resemble the gauge shown below. Note that the arrow is filled. Select appropriate values for the width and length of the arrow and use whatever color, line style, and width you prefer. It's okay to use a semi-circle. Should you decide to use an oval, it's easiest to make the angle of the arrowline be proportional to the ratio of the current value to the temperature range.



      2) (**CS594 students only; CS460 students can get 10 points of extra credit by doing this part of the problem**). A filled pie arc

whose arc is equal to the percentage of the current value as a function of the maximum value. For example, if the minimum temperature is 32 and the maximum temperature is 212, then the arc below represents a temperature of roughly 57 degrees, because 57 degrees is roughly 1/8 of the way between 32 and 212, and a 45 degree arc is roughly 1/8 of the way between 0 and 360. Fill the arc with the color blue.



In /home/bvz/gui/hw/hw2 you will find a class named *TemperatureController*. This class creates a dialog box that you can use to control the model (i.e., it will allow you to set the model to a certain temperature). Your main method in ThermometerApplication should create an instance of TemperatureController after creating instances of the model and the views. The constructor for TemperatureController takes two arguments:

1) a pointer to the model object,
2) a pointer to the JFrame that contains the temperature gauge (not the thermometer view, the new temperature gauge you just wrote).

The dialog box will pop up to the right of the window containing the temperature gauge. When you manipulate the slider in this dialog box, TemperatureController updates the model. Your model should in turn notify the views, which should change in response to changes in the temperature.

I have used a rough heuristic to calculate the correct size for the slider, based on the minimum and maximum values of your temperatures. You may need to resize the dialog box if it is to small to accommodate all the labels on the slider.

If you have questions about how your program should look or run, you can try my executable, /home/bvz/gui/hw/hw2/thermometer.jar. It can be run as follows:

```
java –jar thermometer.jar minTemp maxTemp startTemp true/false
```

where minTemp and maxTemp are the minimum and maximum temperatures, startTemp is the starting temperature. A value of true for the fourth command line argument indicates a pie view should be created and a value of false indicates that a pie view should not be created. Your application will not have the fourth argument, because your application with either create a pie view, if you are taking CS594, or not create a pie view, if you are taking CS460. My jar file allows you to test either case. A sample invocation might be:

```
jar –jar thermometer.jar -50 150 75 true
```

2) Write a program named DisplayImage that displays an image and centers a text caption horizontally beneath the image that describes the image.

3) In this problem you are going to display a set of text strings, find the character at a given mouse coordinate and place a cursor before that character.

   Your program should meet the following specifications:

   a) It should be placed in a package named **textselection** and the file containing *main* should be named **selectText.java**. You should have additional files for the model and the view/controller.

   b) It should take a point size, an (x,y) point, and a series of strings as command line arguments. You can create a multiple word string by putting quotes around it. For example:

   ```
   java textselection.selectText 20 30 50 "Vander Zanden" "Hooty And The Blowfish" "Boo Hoo"
   ```

   c) Your application will display each string on a separate line, in a SERIF font, putting 10 pixels between each line.

   d) The window that displays the text strings should be just large enough to accommodate all the text strings with a 10 pixel border of whitespace around the entire window.

   e) Once the text strings are drawn, your code should determine which one of the strings contains the (x,y) point, and then draw a line between that character and the *previous character*.

   f) If the (x,y) coordinate is within five pixels of the rightmost character in any line, then the cursor line should be drawn after the last character in the line.

   g) If the (x,y) coordinate is between lines, above any of the lines, below any of the lines, to the left of any of the lines, or more than 5 pixels to the right of any of the lines, then no line should be drawn.

Hint: The “Drawing Basics” lecture notes contain a pseudo-code algorithm for determining where a cursor should be placed within multiple lines of text.

Problem restriction: You may not use the TextLayout class for this problem.

If you have questions about how your program should look or run, you can try my executable, /home/bvz/gui/hw/hw2/textselection.jar.