CS315 Programming Project 1: Convex Hull

For this project, you will expand a skeleton application written with the Java Swing Graphical User Interface libraries to find the convex hull of a set of points on a drawing area representing Euclidean 2-space.

SwingShell and CanvasPanel

You will find the source files for these two classes on the class assignments web page. SwingShell is a very basic Swing application that will serve as a starting point for your program. It is a top level JFrame containing a panel for control buttons and a CanvasPanel instance that serves as a drawing area. The constructor builds the appearance of the application. SwingShell implements the ActionListener interface, and thus can respond to events such as mouse clicks. These are automatically passed to the actionPerformed method for processing.

CanvasPanel is essentially a drawing area that also listens for mouse events. Whenever the mouse is clicked, CanvasPanel’s mouseClicked() method is automatically called. The mouseClicked() method learns the coordinates of the click and adds the point to a list of points that it is responsible for drawing on the screen. CanvasPanel also has a paintComponent() method that is responsible for painting the component’s appearance whenever necessary. It draws each point in the list on the canvas. The appearance of this application after several points have been entered is shown below. Note that paintComponent() should never be called directly. If you want to cause the panel to redraw, call repaint() instead.
ConvexHull

Using these files as a starting point, you are to write a program called ConvexHull, which starts with the set of points entered on CanvasPanel, computes the convex hull for the set, and displays it on the panel. You may choose either the Gift Wrapping algorithm or the Graham Scan algorithm to compute the convex hull. Note that you must rename the SwingShell class to ConvexHull! Also remember to put your name in a comment at the top of the files.

Requirements:

• Your application should have two buttons, one named “Convex hull” and the other named “Clear.” When the “Convex hull” button is clicked, your program should examine the points entered on the CanvasPanel, compute their convex hull, and cause the CanvasPanel to redraw with the convex hull superimposed over the points. The “Clear” button should erase the CanvasPanel.
• Adding more points after the convex hull is calculated and then clicking “Convex Hull” again should update the diagram to reflect the new convex hull.
• You should add a “Quit” button that does the obvious thing.

Here is an example of the appearance of the finished application. Note that your program only has to have the “Convex hull” button rather than the two buttons shown here.
**Deliverables**

Your program is due **Friday, February 3 at 5:00 pm**. Your files should be contained in a single JAR archive. In the JAR should be all files I will need to compile and run your program. The command to create a JAR archive looks like this:

```
jar cvf <archive-name> <list of files>
```

So, for example

```
jar cvf ConvexHull.jar ConvexHull.java CanvasPanel.java
```

will create an archive file called ConvexHull.jar with ConvexHull.java and CanvasPanel.java as the contents. Once you have created your archive file, email it to me as an attachment.

**Extra Credit**

You may have an alternative algorithm that you came up with as part of our class discussion. I will grant extra credit for working, well documented alternatives. Add a separate button with “Extra Credit” as the label to indicate that you have done this.