# CS4815 Week06 Lab Exercise

**Lab Objective:** We will complete the lab started in Week05 by

- looking at the use of the output primitive, `GL_LINE_STRIP`, and how C++ code can be executed within `glBegin()` – `glEnd()`, the bounds of the definition of primitives
- adding to the chart drawing program for plotting some silly data, and
- further modifying its behaviour to react to menu and keyboard events
- implementing a zoom function for the plots

Here's a **quick summary** of the tasks:

❶ Copy the source code for this week's lab from the class directory `~cs4815/labs/week06` Do not do this until you read the caveat below.

❷ Modify the program so that it switches between *three* chart-drawing modes as specified by a menu selection or a keystroke

❸ Fix the `lineGraph()` code provided to you

❹ Implement a zoom/unzoom option

## In Detail

❶ This week's lab is a continuation of last week's so you should be careful that you don't overwrite all of your work from last week. Create a new subdirectory, like always, in `~/cs4815/labs/week06`. Now, rather than copying from the class directory, you should copy the two source files from your `week05` subdirectory.

There is just one function of use to you in the class directory this week. This is to plot the data as a line chart, as a third plotting option. The code for this function can be found in `~cs4815/labs/week06/linechart.cc` and you should put this code in your `chart.cc` file. To keep things simple and to keep the compiler happy place it just above the function `pieChart()`.

❷ You should now modify your menu and keyboard handling routines so that they respond to plotting the data in any of the three formats you now have code for: `b`(ar), `p`(ie) and `l`(ine). The keyboard commands to switch between plotting formats are obviously `b`, `p` and `l`.

❸ Just like last time there is a problem with the positioning of the plot. Fix this by adjusting the constants.

While you're snooping around `lineChart()` notice how the primitive definition `GL_LINE_STRIP` can have a for-loop embedded within to permit all 11 line segments (12 points) to be specified as one "thing".

❹ The final task for this week is to implement a zoom function (keyboard `z`) that "magnifies" part of the plot and its inverse unzoom (keyboard `Z`) that backs off from the plot. In order to keep this simple I do not want to get in to panning around the image in order to view other parts that get lost in the zoom. It will be sufficient to show me that you can change the scale of the plot, even if part of it gets lost. Hitting 'z' repeatedly should have the effect of zooming further and further closer; likewise with 'Z' and unzooming.

There is actually very little code to write in order to implement this feature. Look back at previous sample programs and see what is the function that gives us a "window" into the scene. Look up the documentation to see what the parameters mean and now adjust these parameters by means of the `z` / `Z` keys.

I found the discussion here very helpful with implementing the zooming.

Note that this should be independent of whatever plotting format is active currently. You should not go tampering with the code for any of the three formats in order to implement this feature.

This lab, a combination of Week05 and Week06, will be due at the start of next week's labs, as per the usual deadline.

The command to hand it in is:

```
handin -m cs4815 -p w06
```