CIS 441/541: Project #2A
Due Feb 21st (which means 6am Feb 22nd)
Worth 8% of your grade

Overview:
You will make a program that has one window with two sub-frames. The code to set up the windows, sub-frames and event handling is done for you (via the skeleton program project2A.cxx and usage of the VTK library). Your job is to write OpenGL rendering code in two methods (one for each sub-frame).

IMPORTANT: unfortunately, this project requires VTK 6.3 (not 8.1). I am very sorry for the wasted effort this causes. I have built VTK 6.3 in the Room 100. It also available at: https://www.vtk.org/files/release/6.3/VTK-6.3.0.tar.gz

In sub-frame #1:
You will render proj1e_geometry.vtk using the VTK infrastructure for windowing and events and your own OpenGL code for rendering. The OpenGL code will go in vtk441MapperPart1::RenderPiece. You will be rendering the geometry returned from GetTriangles(). The Triangle class now has a "fieldValue" data member, which ranges between 0 and 1. You will map this to a color using the GetColorMap function. You will also include normal for lighting. GetColorMap returns 256 colors. A fieldValue value of 0 should be mapped to the first color, and a fieldValue value of 1 should be mapped to the 255th color. Each fieldValue should be mapped to the closest color of the 256, but interpolation of colors is not required.

You will also add 12 white lines that form the outline of a cube that has coordinates from -10 to +10 in X/Y/Z.

You must use glColor commands, i.e., not textures.

In sub-frame #2:
Exactly the same as window #1, but use textures. Implement your GL code in vtk441MapperPart2::RenderPiece.

Again, you should add white lines for the cube outline.

Hint: I recommend you "walk before you run" & "take small bites". OpenGL can be very punishing. Get a picture up and then improve on it. Make sure you know how to retreat to your previously working version at every step.

Hint: OpenGL "state thrashing" is common and tricky to debug. Get one window working perfectly. Then make the second one work perfectly. Then try to get them to work together. Things often go wrong, when one program leaves the OpenGL state in a way that doesn't suit another renderer.

Hand-in: your code and a screenshot of your program