CIS 410/510: Project #7
Due February 22, 2020
(which means submitted by 6am on February 23, 2020)
Worth 6 points toward your grade

Assignment:

1) Download skeleton code proj7.vtk
2) Download data file noise.vtk
3) Compile and run and make sure it runs correctly

You will do a series of seven tasks to learn VTK. Each time you complete a task, you should make a copy of your source. You will be handing in your source code for each task. They should be named proj7A.cxx, proj7B.cxx, proj7C.cxx, ... proj7G.cxx. What to upload? A tarball with 8 files: proj7A.cxx-proj7G.cxx + one screenshot showing proj7F.cxx working.

What are the seven tasks?:

A) Make the visualization window be 768x768 when the program is first invoked.
B) Add a contour filter to the pipeline, with isovalues at 2.4 & 4.
C) Remove the contour filter and instead slice by Z=0
D) Modify the vtkLookupTable so that it smoothly interpolates from blue at the minimum to red at the maximum, with purple in between.
E) Modify your program to have both a contour and a slice filter. You will need to have two networks.
F) Add two renderers. Place the slice in the left renderer and the contour in the right.
G) Modify your program to iterate over isovalues. Start with 1.0 and add 0.01 all the way up to 6 (a total of 500 isovalues). Render each new value. This will make an animation.

Hints:

First, the execution model appears to have changed in the last year. I found I needed to add many "Update()" calls to make modules update. My advice is to call "Update()" whenever you feel you have a filter that is ready to execute. (This is different than how I lectured.)

=== 7C ===
There is no module for slicing. Instead, you use the module "vtkCutter" and then set up a vtkPlane and assign it as the vtkCutter's "CutFunction".

=== 7D ===
The vtkLookupTable has 256 entries. You want to set the color for each of these 256 entries, manually interpolating from blue to red.

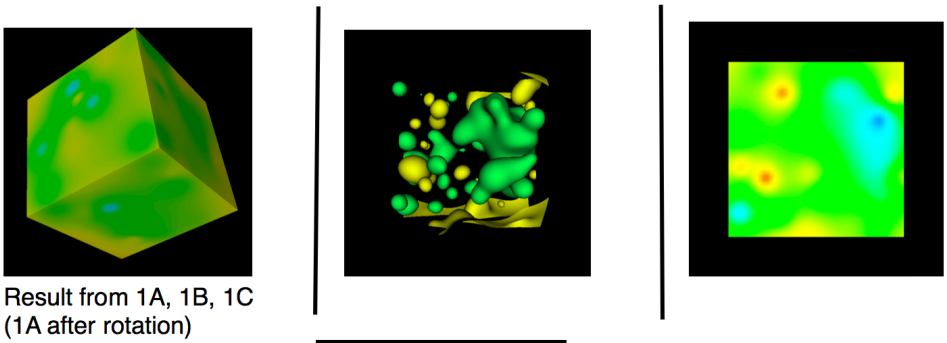
=== 7E ===
The output from the slice and the output from the cutter will each need to have its own mapper and actor (so two mappers and two actors).

=== 7F ===
You will need two vtkRenderers and to add them to the vtkRenderWindow. Also, you'll need a SetViewport command.

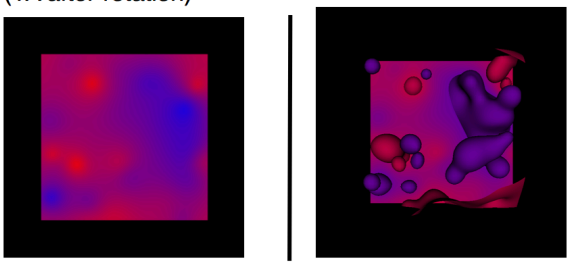
=== 7G ===
Never call vtkRenderWindowInteractor::Start().
Your program should just end after the 500 iterations.
I found that the camera wasn't setting itself correctly. Each time I updated the isovalue, I did:

```
cf->Update();
ren2->GetActiveCamera()->ShallowCopy(ren->GetActiveCamera());
```

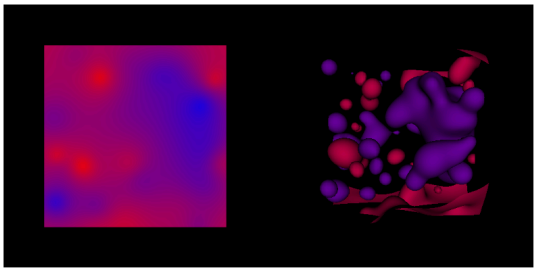
Note: this screenshot refers to the parts as 1A, 1B, etc., instead of 7A, 7B, etc.



Result from 1A, 1B, 1C
(1A after rotation)



Result from 1D, 1E



Result from 1F

Result from 1G: image on right has single isovalue that animates.