In terms of work division, this project was done almost entirely together. Tommy and I met in Taylor and knocked out a good majority of the project in a single day. Using the provided Java code as basis and inspiration, we created most of the needed classes and the code to calculate the heights, colors, and normals of the tessellated triangles. Tommy and I discussed how to display the land, and then I implemented it. I also added the water, and “pillar of ground” underneath the terrain. As a final touch to my work I added the menu options for water, LOD, reset, and roughness. As always, we did equal shares of work and used Git to keep track of everything and allow us to work at the same time.

As for what we did in the lab, we heavily used the provided Java code in order to create a polished final product with the little time we had. We used square-diamond algorithm initially because we had it in front of us. Ultimately, we found that it looked really nice. It created realistic looking land forms, and we decided to keep it. We selected our colors in hopes of making realistic looking lands. The top color is a bluish grey, which looks a lot like a mountain top. The center green is a darker green to represent forests, which fades to blue as it gets closer to the ocean, which is a deep blue-green.

The default settings are an LOD of 5 and a roughness of 0.5. These settings yield a video-gamey look, and with water enable, it looks like a tropical coast. Increasing the roughness makes the terrain look more mountainous to an extent, and then the land starts to look more like grass. Increasing the LOD generates more detailed terrain, and when it is set to 7, it really looks like a realistic coastline. Lower LOD generate more and more basic land forms, with less than 4 looking almost like building blocks.

The ground below the terrain was generating by creating a polygon for every two points going along the edge to down below where the camera can see. This gives the effect that the terrain is cut out of the ground. Interestingly, water was done in much the same way. First a flat plan at 0.2 across the whole terrain. This gets most of the look of the water, be doesn’t look good around the edges. To remedy this, for every two points around the edge that are below 0.2, a polygon is created that goes up to 0.2. This again generates the illusion that the terrain was cut out of a larger piece of ground.

Couple of interesting technical things to mention: regular arrays were not usable for our implementation. Because 2D arrays cannot be declared with non-constant variables, we had to use vectors. Ultimately, this is a good thing in terms of programming practice, but we also run into issues with the maximum size, which is why we are limited to an LOD of 7.

In terms of using what I have learned this semester, I can safely say I used it all in the project. This project touches on just about every topic that we have covered in class. We didn’t use GLSL ourselves, but that was definitely room for it in doing lighting. I really liked the final product in this project, it really looks great.

Final thought: I’d prefer to have switched this project with project three. I feel that that project was more open, and allowed more freedom in terms of what we do, and how complicated it could get.