Assignment #3
CS174A – Introduction to Computer Graphics - Fall 2014

DUE: November 7, 2014 by 11:59PM

YOU MUST UPLOAD THE FOLLOWING IN SOME TYPE OF CONTAINER (TAR,ZIP,etc)
- A README DESCRIBING WHAT YOU DID AND DID NOT DO
- THE SOURCE CODE FOR YOU SUBMISSION – INCUDE EVERYTHING WE NEED TO COMPILE AND RUN YOUR SUBMISSION

100 POINTS AVAILABLE / PARTIAL CREDIT MAY BE AWARDED

The following is **required** for the assignment and provides 100 points of basic credit:

1. Get a simple WebGL capable window to display without error – 5 points.

2. Develop a function that generates the geometry for a sphere. Use a parameter to control the number of vertices that form the sphere. You can use the sphere generator in the book or find an alternative technique on the Internet – 10 points.

3. Extend your sphere function to, along with the vertices, generate normals for shading. A parameter should specify whether normals for flat or smooth (Phong type) shading are created – 10 points.

4. Create a small solar system. One sun in the center with four orbiting planets. You choose a location (other than the world coordinate system origin) and diameter of the sun, the radius of each planet's orbit (keep them in the same plane), the diameter of each of the four planets and, finally, how fast each planet orbits around the sun – but each should move at a different speed – 20 points.

5. Implement the sun as a point light source. You have already selected the diameter of the sun, however, if the sun is large the color should be warmer (reddish), if smaller the color should be colder (blueish). Each planet has a different appearance. First, there is an icy white, faceted, diamond-like planet (medium-low complexity sphere, flat shaded, specular highlight). Second, there is a swampy, watery green planet. (medium-low complexity sphere, Gouraud shaded, specular highlight). Third, there is a planet covered in clam smooth water (high complexity, Phong shaded, specular highlight) and finally a mud covered planet, brownish-orange with a dull appearance (medium-high complexity, no specular highlight) – 20 points.

6. Implement (re-use) your keyboard based navigation system from Assignment #2 to allow a user to fly around your solar system. Define and document a key to reset the view so that the entire solar system is visible – that view should be from above looking down at a 30 degree angle. This same view position should be used when your application starts. – 10 points

7. Implement the Phong shading model to appropriately shade each of the planets as outlined above. It is up to you to figure out where the various parts of the shading model are best implemented (i.e. in the application, vertex shader or fragment

shader). For instance does it make sense to do some part calculation on the CPU or GPU? Are some calculations performed per object, per primitive or per fragment and how might that affect how you implement a solution? – 25 points.

Extra Credit

1. Add a moon orbiting around one of your planets. You decide on the moon's orbital speed, diameter, color and appearance – 15 points.

2. Define and document a key that will allow the eye point (camera) to attach/detach to one of the orbiting planets. While attached allow only the heading to be changed so you can look around at the other planets moving. You can place the eye point at any point near the planet (fixed in orbit) – 20 points.