Assignment 3: Smooth Cone

In this assignment, you'll mainly perform Gouraud shading to get a smoothly shaded cone. You've been given a cone which is shaded only with the ambient reflection model as shown to the left of Figure 1. After applying the diffuse and specular reflection while considering material properties, the properly shaded cone will look similar to the right of Figure 1. By toggling the '**s**' button from the keyboard, you can view the wireframe model of the cone. In this assignment, you need to write codes both in the application and vertex shader.

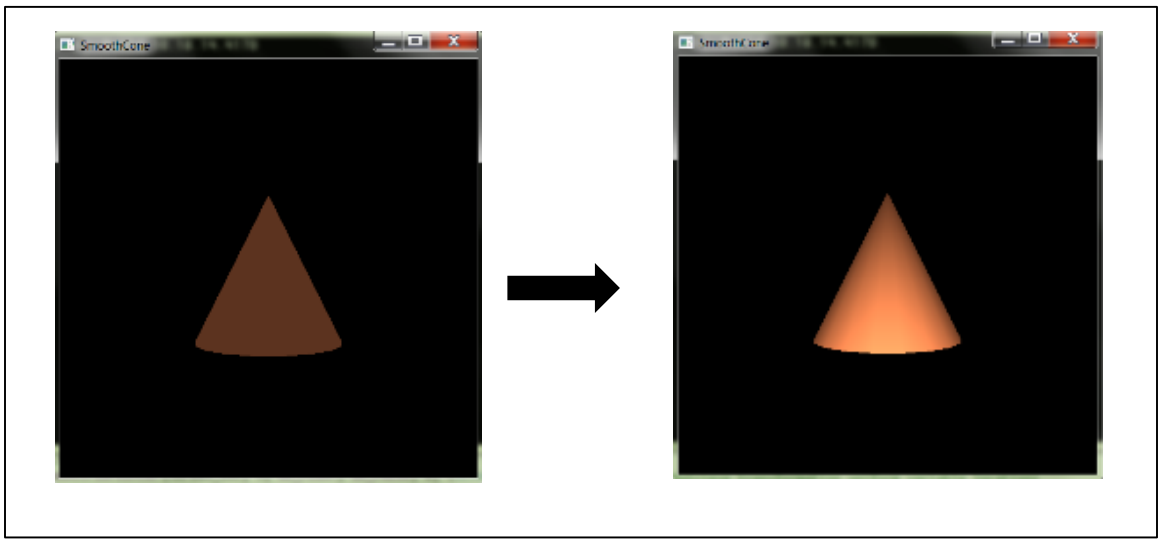

Figure 1: A cone (Left) not properly shaded and (Right) shaded.

In **Application "Smoothcone.cpp"**:

1. You need to implement the following function:

   updateVertexNormals(){};

   At this moment, the vertex normals have not yet been computed. You need to calculate per vertex normal in this function in order to implement Gouraud shading.

2. Next, you need to take a look at 'Initialize()' function. It defines several material and lighting parameters (Figure 2). These are defined as uniform variables in the vertex shader. Some of the variables have been assigned values by obtaining the indices of the corresponding uniform variables defined in the vertex shader. You need to work on the variables which have not yet been assigned values. Take a look at the vertex shader ("smoothshader.vert") to have a clear understanding of the uniform variables associated with the material and lighting properties.

```
// Initialize shader material and lighting parameters

	vec3 material_ambient(0.9, 0.5, 0.3);
	vec3 material_diffuse(0.9, 0.5, 0.3);
	vec3 material_specular(0.8, 0.8, 0.8);


	vec3 light_ambient(0.4, 0.4, 0.4);
	vec3 light_diffuse(1.0, 1.0, 1.0);
	vec3 light_specular(1.0, 1.0, 1.0);

	float material_shininess = 150.0f;
```

Figure 2: Variables associated with the material and lighting properties in Display function.

3. In "Display()" function, a point light has been defined. You need to calculate per vertex light direction in the vertex shader. A number of matrices have been defined. Take a look at the matrices defined as uniform variables in the vertex shader. In application, you need to find the indices of all these uniform variables and assign values to them. You need to define the model view matrix and normal matrix. As already discussed in the lecture (8th week), a normal matrix is usually the upper left 3X3 of the model-view matrix.

In **vertex shader**:

1. You need to do the eye space conversion of the vertices and normals by multiplying with the appropriate matrices. As already mentioned, you need to calculate per vertex light direction. In eye space, the viewer is located at the origin. You need to consider this while calculating the half vector for specular reflection.
2. Calculate the diffuse and specular reflection considering material properties and other reflection coefficients as mentioned in the shader.
3. Calculate the combined light intensity to get the desired result.

Submission:

Submit the assignment in a zipped file via canvas. Name the file as Firstname_Lastname_3.zip (your first name and last name). Deadline is Monday, November 30, 11:59 pm.

This assignment carries a weightage of **30%** of this course.