A11b – per Pixel and Vertex shading

In this assignment, you have to write the vertex and fragment shaders to support four different types of shading models: per vertex, per pixel, light per vertex and BRDF per pixel, light + ambient + diffuse + emission per vertex and specular per pixel. The application contained in `index.html`, and shaders code should be written in file `shaders.js`. The code, written in GLSL, can use the following variables and uniforms to find the elements necessary to compute the final color:

```
uniform vec3 eyePos;                // viewer position
uniform vec3 LAPos;                 // light position
uniform vec4 LAlightColor;          // light color
uniform vec4 ambientLightColor;     // ambient light color
uniform vec4 diffuseColor;          // diffuse light color
uniform vec4 specularColor;         // specular color
uniform float SpecShine;            // specular shine
uniform vec4 ambientMatColor;       // ambient mat colro
uniform vec4 emitColor;             // emit colorfloat LBConeOut;
```

Vertex position and normal vector directions are found in:

```
in vec3 in_pos;
in vec3 in_norm;
```

Final color is returned in:

```
out vec4 color;
```

The following GLSL standard procedure can be helpful in solving this exercise:

normalize()
cos()
radians()
pow()
dot()
length()
clamp()
reflect()
max()

whose reference can be found at pages 7 and 8 of the official WebGL reference card from Kronos Group (`webgl20-reference-guide.pdf`, enclosed in this ZIP file for convenience).

The first two light models have been implemented to guide you in writing the code