Lab #3:

In vertex shader :

a. Declare light direction as uniform vector of size 3 (it is going to be in world space):

```
uniform vec3 lightDir;
```

1. *Pass uniform variable through your openGL Code*

1. *Compute the diffuse term in vertex shader:*

```
void main()
{
vec3 normal, nlightDir;
vec4 diffuse;
float NdotL;
            /*first transform the normal into eye space and normalize the
            result*/
      normal =
             /* now normalize the light's direction. Note that according to
             the OpenGL specification, the light is stored in eye space. Also
             since we're talking about a directional light, the position
             field is actually direction*/
      nlightDir =
             /* compute the cos of the angle between the normal and lights
             direction. The light is directional so the direction is constant
             for every vertex. Since these two are normalized the cosine is
             the dot product. We also need to clamp the result to the [0,1]
             range. */
      NdotL =
            /* Compute the diffuse term */
            diffuse =
            gl_FrontColor =
            gl_Position = ftransform();
}
```

1. *In fragment shader*

```
void main()
{
gl_FragColor = gl_Color;
}
```

1. *Use* `gl_LightSource[0].position` *instead of* `lightDir`
2. *Calculate the ambient terms*

```
…
/*declare ambient and globalAmbient variables*/
/* Compute the ambient and globalAmbient terms */
ambient =
globalAmbient =
/* modify FrontColor both diffuse and ambient components*/
gl_FrontColor =
```

1. *Calculate the specular component*

```
/* compute the specular term if NdotL is larger than zero */
if (NdotL > 0.0)
{
            // normalize the half-vector, and then compute the cosine (dot
            product) with the normal
       NdotHV =
       specular =
}
```

1. *Recalculate Calculate the specular component*

```
FrontColor =
```

10. Modify to work with fragment shader

a. *Change* `diffuse,ambient, vec4 diffuse,ambient, normal,lightDir,halfVector` *into varying variables In v*

b. ertex shader do the following

```
void main()
{
/* first transform the normal into eye space and
normalize the result */
normal =
               /* now normalize the light's direction. Note that according to
               the OpenGL specification, the light is stored in eye space.
               Also since we're talking about a directional light, the
               position field is actually direction */
lightDir =
/*Normalize the halfVector to pass it to the fragment shader*/
halfVector
/* Compute the diffuse, ambient and globalAmbient terms */
diffuse =
ambient =
gl_Position = ftransform();
}
```

c. In fragment shader do the following

```
      vec3 n,halfV;
      float NdotL,NdotHV;
      /* The ambient term will always be present */
      vec4 color = ambient;
```

```
                /* a fragment shader can't write a varying variable, hence we
                need a new variable to store the normalized interpolated
                normal */
        n = normalize(normal);
        /* compute the dot product between normal and ldir */
        NdotL =
…
gl_FragColor = color;
Bonus:
Add second light
```