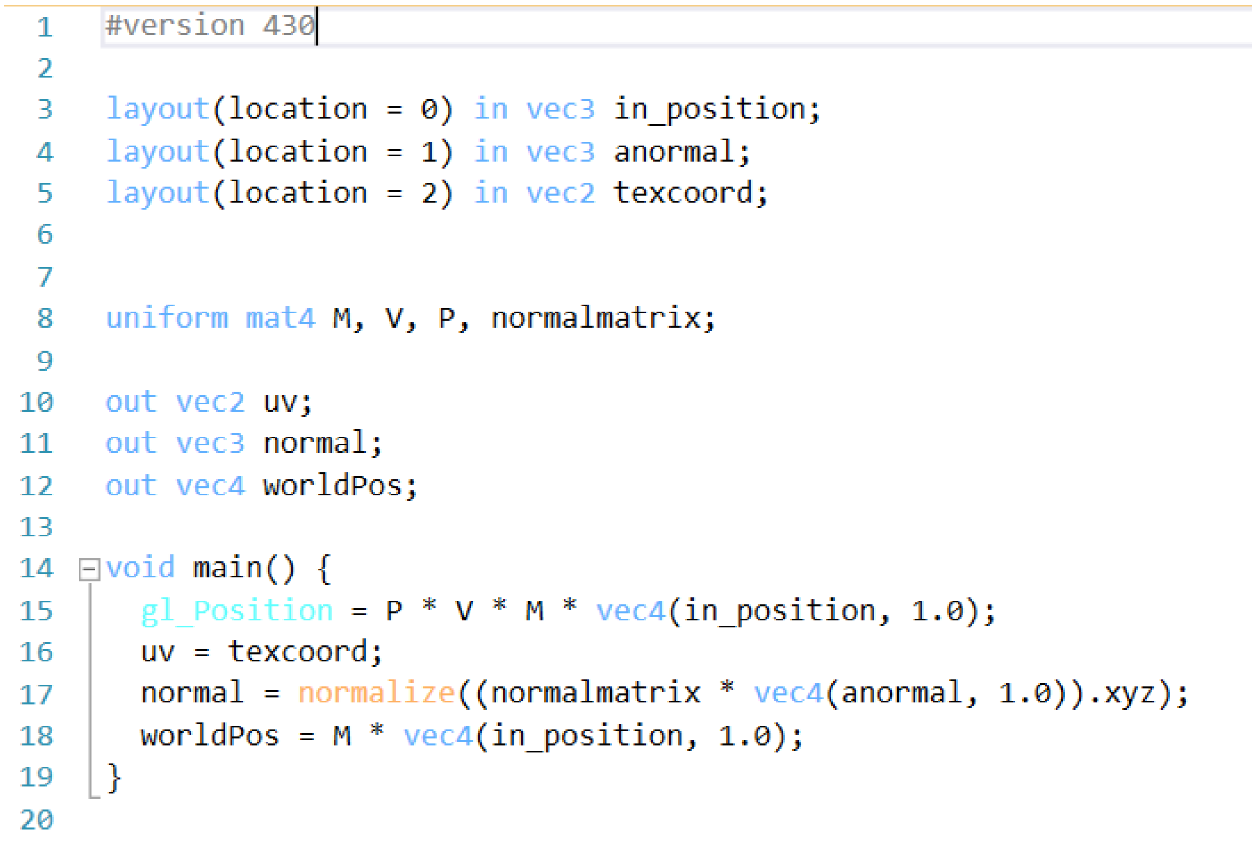
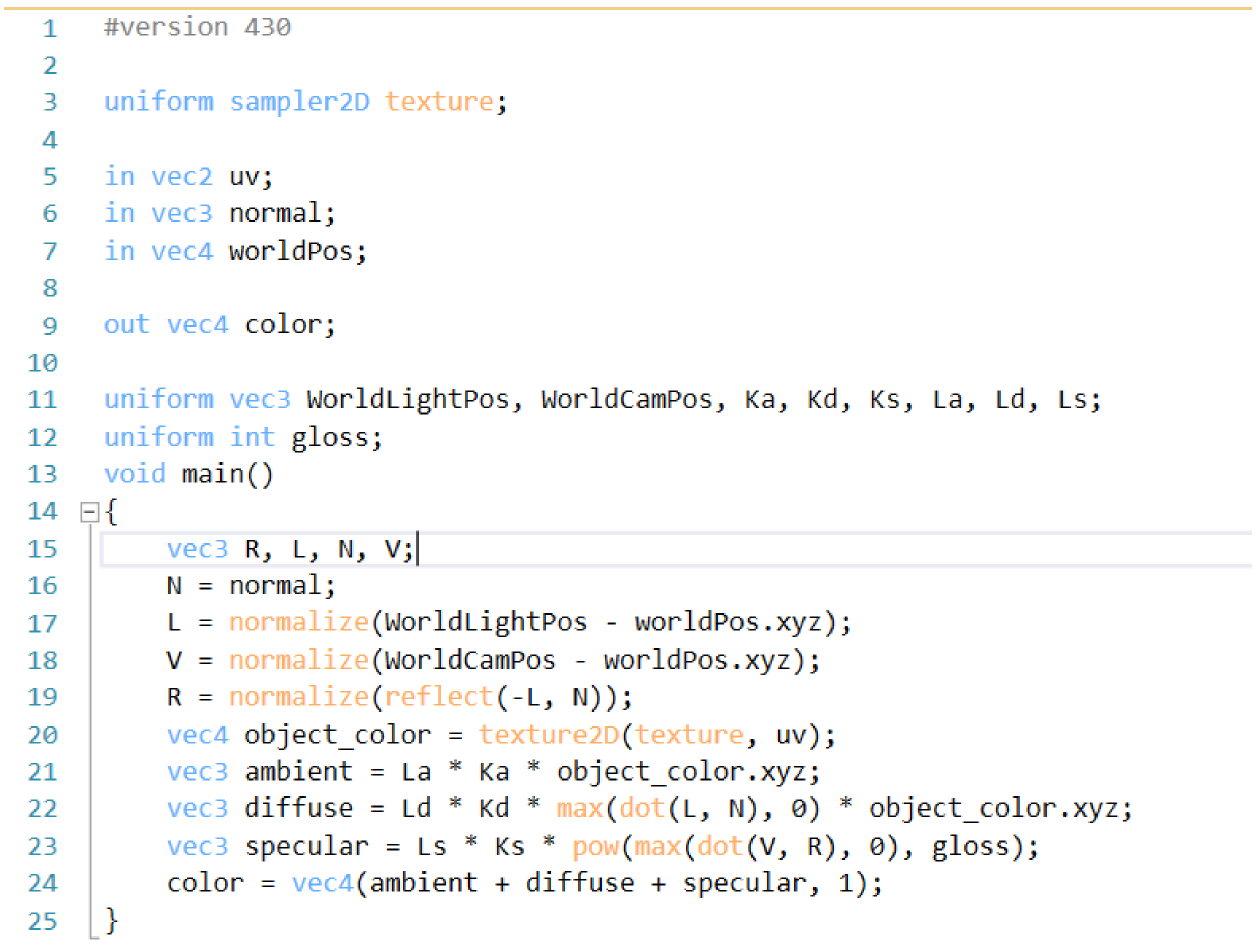
0616018 林哲宇

.vert 全部都是一樣的，就是為了算出 normal，其中 normalmatrix 是把 model 做 inverse 再做 transpose 算出來的



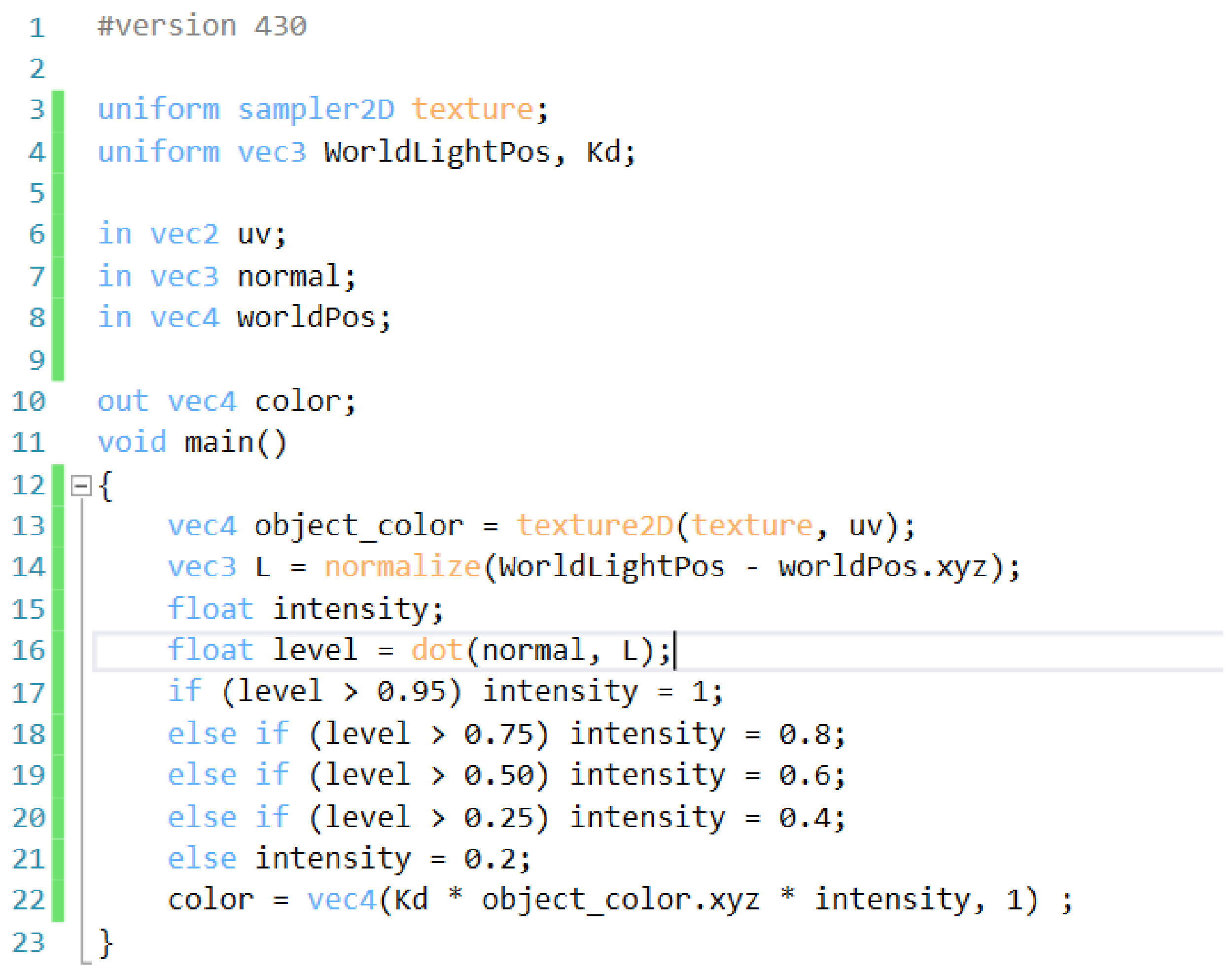
```glsl
#version 430

layout(location = 0) in vec3 in_position;
layout(location = 1) in vec3 anormal;
layout(location = 2) in vec2 texcoord;


uniform mat4 M, V, P, normalmatrix;

out vec2 uv;
out vec3 normal;
out vec4 worldPos;

void main() {
    gl_Position = P * V * M * vec4(in_position, 1.0);
    uv = texcoord;
    normal = normalize((normalmatrix * vec4(anormal, 1.0)).xyz);
    worldPos = M * vec4(in_position, 1.0);
}
```

Phong shading 的 .frag，照著 pseudo code，把該算的 N, V, L, R 算出來後代進去就好了，而算法是參考 https://learnopengl-cn.github.io/02%20Lighting/02%20Basic%20Lighting/



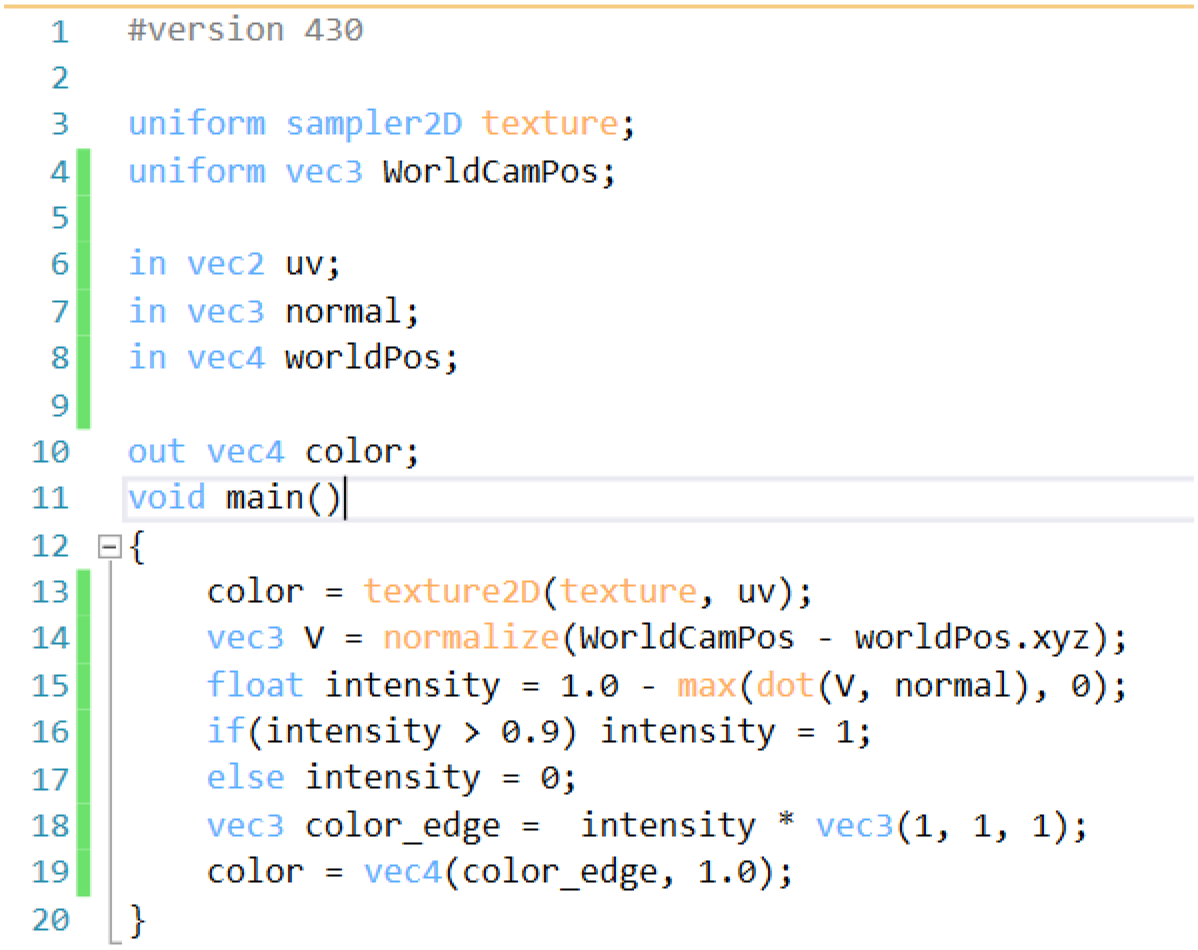
```glsl
#version 430

uniform sampler2D texture;

in vec2 uv;
in vec3 normal;
in vec4 worldPos;

out vec4 color;

uniform vec3 WorldLightPos, WorldCamPos, Ka, Kd, Ks, La, Ld, Ls;
uniform int gloss;
void main()
{
    vec3 R, L, N, V;
    N = normal;
    L = normalize(WorldLightPos - worldPos.xyz);
    V = normalize(WorldCamPos - worldPos.xyz);
    R = normalize(reflect(-L, N));
    vec4 object_color = texture2D(texture, uv);
    vec3 ambient = La * Ka * object_color.xyz;
    vec3 diffuse = Ld * Kd * max(dot(L, N), 0) * object_color.xyz;
    vec3 specular = Ls * Ks * pow(max(dot(V, R), 0), gloss);
    color = vec4(ambient + diffuse + specular, 1);
}
```

Level 就是把法向量 normal 和光源 L 內積，就是說兩個向量如果越接近平行，則強度越大



```
#version 430

uniform sampler2D texture;
uniform vec3 WorldLightPos, Kd;

in vec2 uv;
in vec3 normal;
in vec4 worldPos;

out vec4 color;
void main()
{
    vec4 object_color = texture2D(texture, uv);
    vec3 L = normalize(WorldLightPos - worldPos.xyz);
    float intensity;
    float level = dot(normal, L);
    if (level > 0.95) intensity = 1;
    else if (level > 0.75) intensity = 0.8;
    else if (level > 0.50) intensity = 0.6;
    else if (level > 0.25) intensity = 0.4;
    else intensity = 0.2;
    color = vec4(Kd * object_color.xyz * intensity, 1) ;
}
```

Edge Effect 就是要描繪出物體的外緣，所以也就是說看到的部份會跟視點到物體的向量接近垂直，所以將 1 減掉 View 和 normal 內積，如果值越大代表越接近垂直。因此設一個 threshold 0.9，如果算出的值大於 0.9，則給予它顏色。



```
#version 430

uniform sampler2D texture;
uniform vec3 WorldCamPos;

in vec2 uv;
in vec3 normal;
in vec4 worldPos;

out vec4 color;
void main()
{
    color = texture2D(texture, uv);
    vec3 V = normalize(WorldCamPos - worldPos.xyz);
    float intensity = 1.0 - max(dot(V, normal), 0);
    if(intensity > 0.9) intensity = 1;
    else intensity = 0;
    vec3 color_edge =  intensity * vec3(1, 1, 1);
    color = vec4(color_edge, 1.0);
}
```