NORMALES / DIFFUSE LIGHTING

- Partiendo del ejemplo del triangulo simple convertir nuestra malla X,Y,Z en una malla con normales: X,Y,Z,Xn,Yn,Zn

Podéis prescindir de usar arrays floats puros y usar estructuras en su lugar

```
struct MeshVtx
{
  float x,y,z;
  float xn,yn,zn;
};
```

- Adaptar la carga de la malla al nuevo tipo. Lo único necesario es adaptar el glVertexAttribPointer existente y añadir uno nuevo para el atributo de las normales:

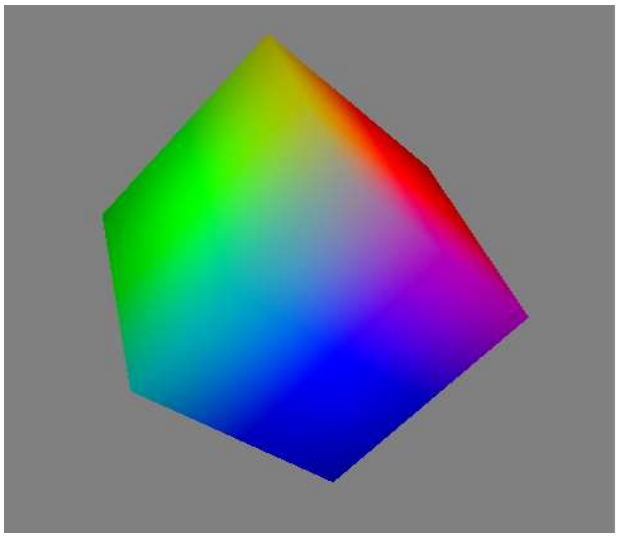
```
glVertexAttribPointer(1,...
glEnableVertexAttribArray(1);
```

- Adaptar el vertex shader para que lea las normales de la malla y las pase al fragment shader
  Para ello se debe:
  - Crear una variable "in" que use "layout" con el atributo 1
  - Crear una variable "out" con la normal

- Modificar el fragment shader para asegurarnos de que hemos recibido la normal, mostrándolo como color del triángulo; para ello creamos una variable "in" de la normal, usando el mismo tipo y nombre de "out" del vertex shader y lo asignamos al FragColor, adaptando el tipo (vec3 -> vec4).

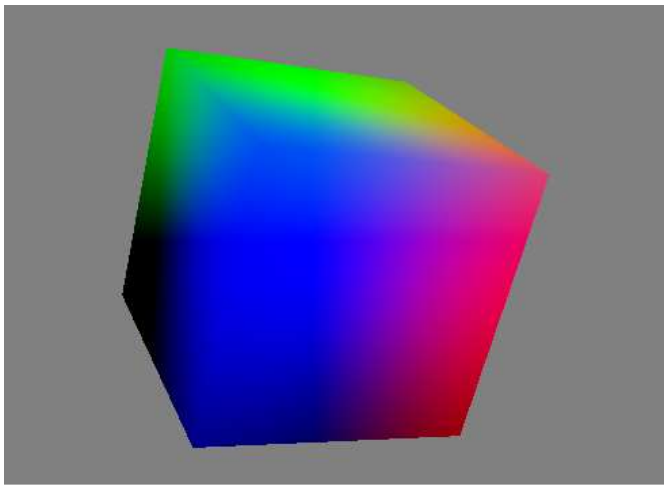
-Una vez tenemos un color correspondiente a la normal que hemos insertado en el vértice, podemos crear un objeto básico interesante para iluminar.
Declaramos un cubo con el mismo formato, vértice+normal

```
{
 {1.0f,1.0f,1.0f,0.577f,0.577f,0.577f},
 {-1.0f,1.0f,1.0f,-0.577f,0.577f,0.577f},
 …
```

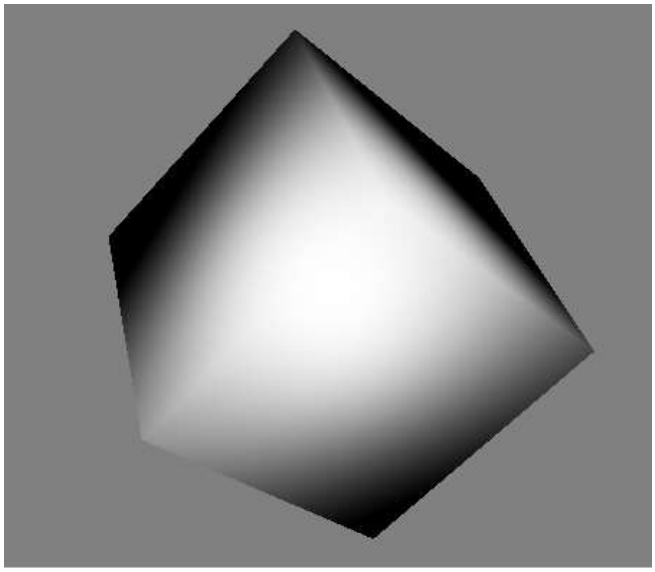


Nos debería aparecer un cubo con las normales interpoladas
Una vez nos aseguramos que las normales responden a lo esperado, implementamos una luz difusa con una fuente de luz direccional

- En el vertex shader, giramos la normal llevando a espacio de mundo (usando la submatriz 3x3). Visualizando el resultado, deberíamos observar como las normales se mantienen de cara adaptándose al giro del cubo (los colores se deslizan sobre la superficie).



- En el fragment shader:
    - Renormalizamos el vector normal (que puede acortarse debido a la interpolación entre vértices)
    - Hacemos un producto escalar con el vector de luz direccional (por ej. 0,0,1). Es una buena práctica evitar colores negativos limitando el resultado del “dot product” a 0.0
    - Sacamos el resultado por FragColor para ver la iluminación

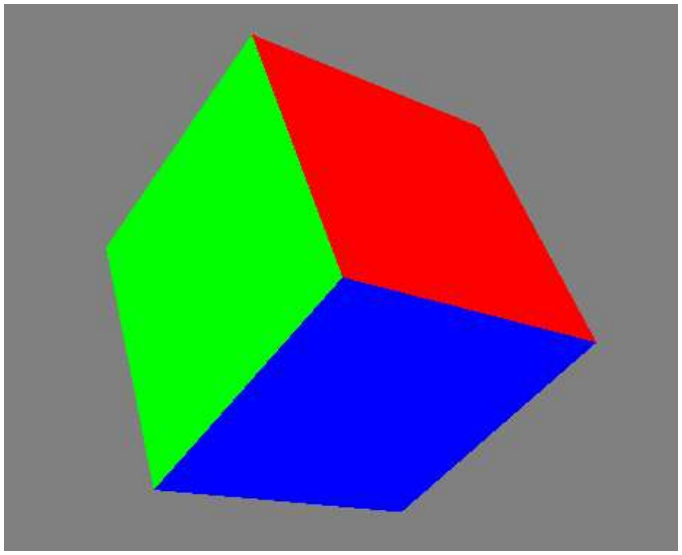


Alternativamente, se puede pasar el vector de luz a espacio de objeto en el programa principal y nos ahorrariamos la transformación de normal por vértice.

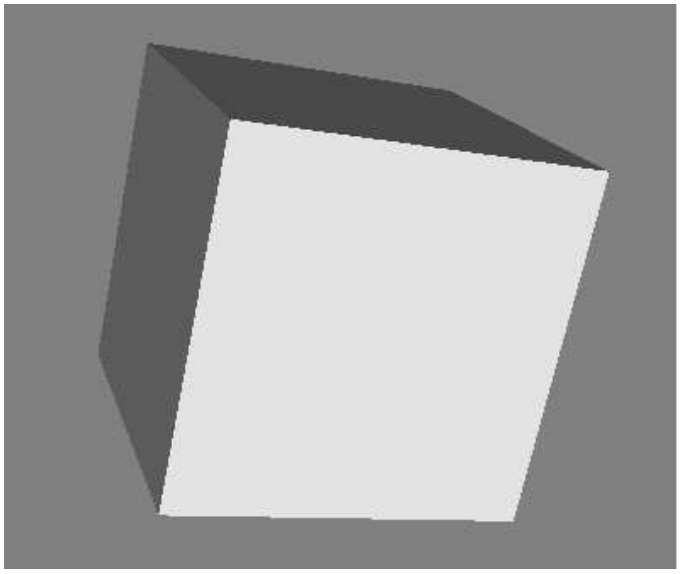
Con esta implementación observamos el glitch de la interpolación de las normales, que es excesiva para un objeto con ángulos tan separados entre las caras. Nuestro próximo objetivo será separar las normales.

- Preparar un segundo cubo que no interpole las normales. Para ellos, los vértices tendrás que ser replicados cara a cara, de manera que podamos usar las normales independientes de cada cara (1,0,0 / 0,1,0 / 0,0,1 / -1,0,0 / 0,-1,0 /…)

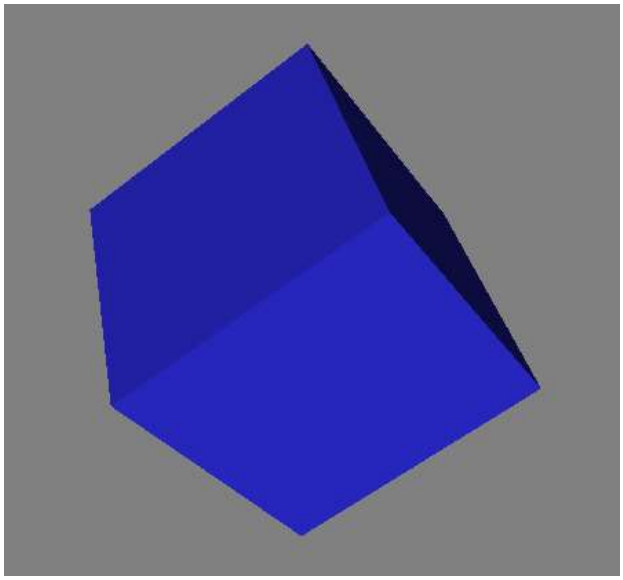
Normales:



Diffuse:



- Por último, usamos “uniform” para pasar al fragment shader un color. Usaremos la iluminación para multiplicar este color



Ahora pasamos a la iluminación difusa con una luz puntual, en vez de la simple direccional.

Para poder calcular la iluminación de un punto del espacio con una fuente de luz puntual, necesitamos la posición de ambos puntos.
Por tanto ya no basta con la normal de la superficie. También necesitamos la posición del punto de la superficie que estamos pintando.

-El vertex shader debe sacar como output la posición. Creamos una variables de salida:

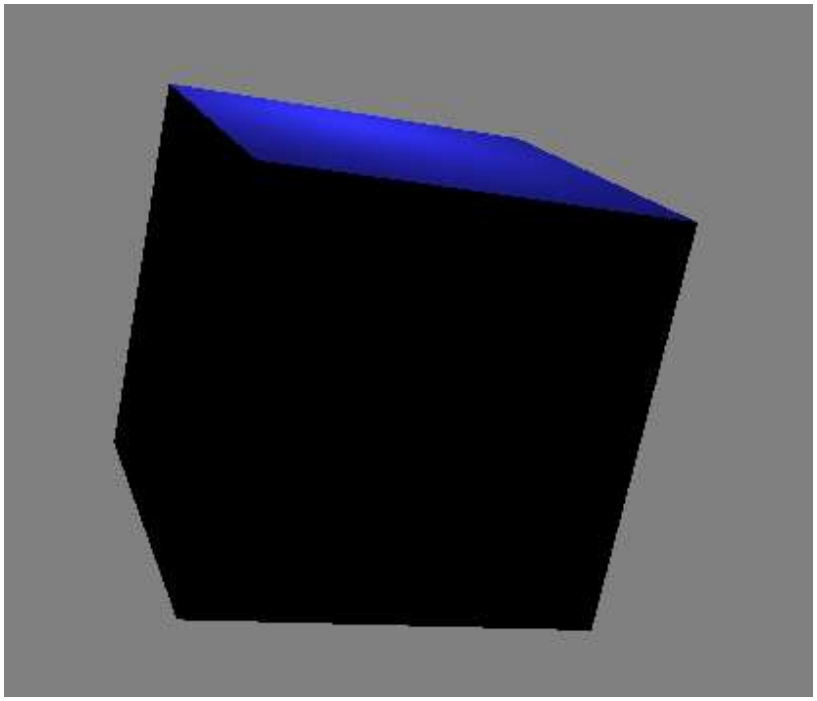
out vec3 FragPos;

…que debe sacar la posición de mundo de los vértices, interpolados para el punto concreto a dibujar:

FragPos = vec3(model * vec4(aPos,1.0));

- El fragment shader debe obtener el vector incidente de luz restando la posición de la superficie (FragPos) de la posición de la luz. Una vez restada, se debe normalizar para poder hacer el producto escalar.

Le pasaremos la posición de la luz usando un uniform. Una luz fácil de comprobar puede ser una frontal o una encima del cubo (ej. 0,2,-7).



-Para comprobar que la iluminación es correcta es buena idea ralentizar/parar el cubo y mover la posición de luz alrededor de él.