Contesta las siguientes cuestiones:

**1º) Enumera y define en orden los diferentes pasos del pipeline gráfico desde la subida de datos de CPU-GPU hasta el output merge.**

**CPU:**

Vertex Buffer Object 🡪Geometry 🡪Clipping Rasterization 🡪 Fs 🡪Image

**GPU:**

Vertex Shader (GPU) 🡪Geometry Shader 🡪Rasterization 🡪Fs → Assembly Image Clipping Ps

**2º) ¿Qué matrices intervienen en la transformación de un vértice para pasar de coordenadas locales de modelo a coordenadas homogéneas? Enumera cada transformación y qué matriz interviene en cada una de estas.**

Interviene MVP (Model View Projection), en Model son translación, rotación y escala del objeto, View las coordenadas de Camera y Projection es el tamaño del campo de visión.

**3º)¿Qué es un shader? ¿Qué tipos de shader son los más comunes y para qué sirven? ¿Dónde se compilan y se ejecutan los Shaders?**

Es un programa o proceso que se ejecuta múltiples veces en paralelo en la GPU. Los más comunes són Vertex shader y Pixel shader. Sirven para iluminación, sombreado y también para efectos, animaciones, etc.

**4º) Al cargar una mesh, ¿Qué ocurre con estos vértices, qué implica esta carga? ¿Qué pasamos al pipeline gráfico, cómo sabe el pipeline gráfico diferenciar dentro de todos los vértices qué son las posiciones/normales/texturas?**

Implica que los vértices locales de la mesh cambian a los vértices del mundo, y esto es debido a la Model View Projection.

Pipeline sabe la diferencia porque se le pssamos “Layout Declaration”, eso hace que mediante el buffer de información este sabe el número de bytes que se le asigna a cada uno. Nosotros le pasamos 4, los primeros 3 son los colores y el último es el Alpha, que es la transparencia.

**5º) Dentro de las 3 escenas diferentes que has realizado podrías enumerar cuántas meshes cargadas se encuentran en la GPU, matrices, uniformes...**

Se cargan 3 meshes, pero solo se renderizan aquellas meshes que estén por pantalla. Solo se muestran aquellas meshes que estén por pantalla.