Asn5: 5a

The purpose of the boilerplate code in lines 123-145 is to set up the WebGL program. This includes creating a canvas element, obtaining a WebGL2 rendering context, compiling and linking shaders, and setting up buffers and attributes for sending data to the shaders.

The uniform code in lines 210-217 works with the shaders by setting uniform variables. Uniforms are variables that maintain the same value for all vertices or fragments processed by the shader. In this code, matrices representing the model, view, and projection transformations are sent to the vertex shader using uniform variables. These matrices are then used to transform vertex positions in the vertex shader.

The attribute code in lines 222-228 works with the shaders by specifying how vertex data is laid out in the buffer and how it should be interpreted by the shaders. Attributes are variables that vary per vertex. In this code, the position (p) and texture coordinates (uv) of each vertex are specified as attributes and enabled for vertex shader input.

The vertex shader (defined in vertexShaderSource) is responsible for transforming vertex positions from object space to clip space. It applies transformations using matrices for model, view, and projection, then outputs the transformed vertex position (gl_Position). Additionally, it passes texture coordinates from the vertex shader to the fragment shader (texCoord).

The fragment shader (defined in fragmentShaderSource) is responsible for determining the color of each fragment (pixels) of the rendered object. In this code, it samples a texture (texture1) using texture coordinates received from the vertex shader (texCoord) and outputs the sampled color (fragColor).