November 8, 2017 Name _____

CS445 Exam 2 Solutions

Fall 2017

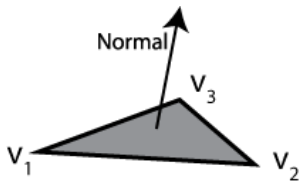
1.	(max = 16)	4.	(max = 10)	7.	(max = 12)
2.	(max = 17)	5.	(max = 6)	8.	(max = 12)
3.	(max = 11)	6.	(max = 16)		
Final Score:			(max=100)		

Please try to write legibly. The instructor's complete failure to decipher what you write will be considered an incorrect answer.

1. (16 pts) Normals:
   a) (4 pts) Given the triangle below with vertices v_0, v_1, and v_2, how do you calculate the upward facing normal as a function of the vertices?

$a = v_2 - v_1$ normal = $\frac{a \; X \; b}{\|a \; X \; b\|}$

$b = v_3 - v_1$

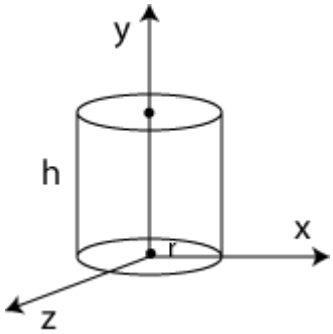


   b) (6 pts) Given the cylinder shown below, what is the *outward* facing normal for a vertex v=(x,y,z,1) on the

      i) Top: (0, 1, 0, 0)

      ii) Bottom (0, -1, 0, 0)

      iii) Side (x,0,z,0) / || (x,0,z,0) ||

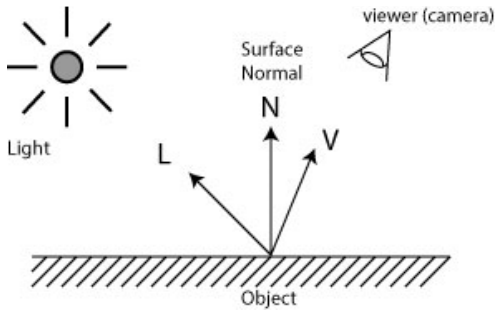


   c) (6 pts) Transformations: In webGL, vertex transforms can safely be used for transforming the normals in which of the following cases (circle any or all that apply)

      i) All rotations.
      ii) Only rotations about the coordinate axes.
      iii) All scale transforms.
      iv) Only uniform scales transforms.
      v) Only non-uniform scales transforms.
      vi) All translations.
      vii) Only translations along a coordinate axis.

2. (1 pt each, 17 pts total) Circle True (T) or False (F):

For the lighting questions below, assume the picture:



a) **T or F:** When a completely transparent object is rendered, the z buffer is never modified.

b) **T or F:** Both vertex and fragment shaders have direct access to uniform variables.

c) **T or F:** Only the fragment shader has direct access to the attribute variables.

d) **T or F:** Clipping occurs between the vertex and fragment shaders.

e) **T or F:** Surface normals are typically stored in uniform variables.

f) **T or F:** In Phong lighting, ambient light depends on the light location.

g) **T or F:** In Gouraud shading, light values are calculated at the vertices and then interpolated across the fragments.

h) **T or F:** The intensity of diffuse light doesn't depend on the location of the viewer.

i) **T or F:** The intensity of the specular light is strongest when the angle between the light and the viewer is small.

j) **T or F:** Aliasing commonly occurs when there is minification.

k) **T or F:** In the shader pipeline, the perspective division is performed in the fragment shader.

l) **T or F:** Anti-aliasing techniques work by increasing the resolution of the screen as needed.

m) **T or F:** Pixelation commonly occurs when there is minification.

n) **T or F:** A texel is the unit corresponding to a pixel in screen coordinates.

o) **T or F:** Bump mapping adds the appearance of surface detail by both modifying the normals and adding extra geometry.

p) **T or F:** A uniform variable called a sampler2D is used to store a 2D texture.

q) **T or F:** If M is the matrix that transforms a vertex, then $(M^T)^{-1}$ is how the corresponding normal transforms.

3. (11 pts total) **Shaders and the Shader Pipeline**: *Please answer in complete sentences.*

a) (4 pts) What is a fragment?

A fragment is a "preliminary" pixel. The final color of a pixel in the display buffer is a function of the many fragments that overlap at that pixel's location. Each of these overlapping fragments is the result of the rasterization of some triangle that covers that pixel.

b) (4 pts) What is the difference between a uniform variable and attribute variable?

An attribute variable is associated with each vertex. The uniform variable is applied to each object (or draw command)

c) (3 pts) List 3 operations that webGL performs *between* the vertex and fragment shader?

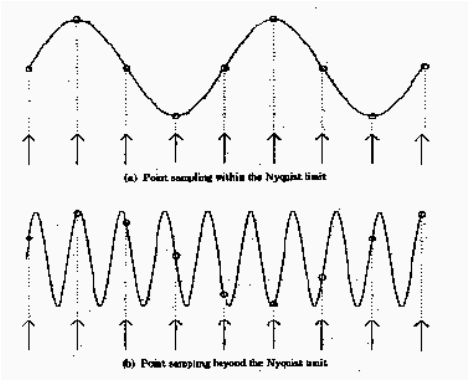
After the vertex shader and before the fragment shader, opengl does the following:

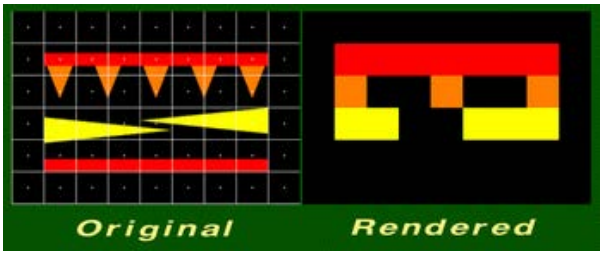
(1) Perspective division
(2) Primitive assembly
(3) Backface culling
(4) Viewport transform
(5) Rasterization
(6) Clipping

4. (5 pts each, 10 pts total) Aliasing and Anti-aliasing:

   a) What is aliasing? Include an example/picture from either signal processing or graphics.

   When a continuous or analog signal (e.g. image or signal) is sampled at a frequency that is too low, it results in an incorrect reconstruction of the image or signal. In the case of an image, details can be lost. In the case of a signal, the frequency of the constructed signal is different (lower) than the frequency of the actual signal.

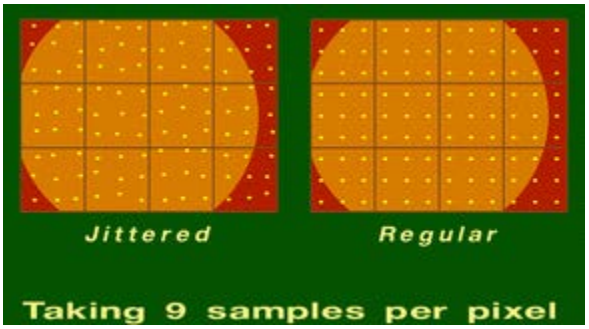




Above: signal processing
Left: image processing

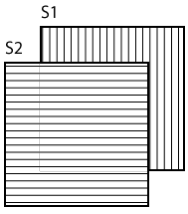
   b) What is anti-aliasing? Describe at least one anti-aliasing technique.

   Anti-aliasing techniques are methods that try to reduce the effects of aliasing. Mipmapping is used in webgl where a texture is stored at multiple levels of resolution. Subsampling is also used where the color of a pixel is sampled at several positions in the pixel (e.g. see image on right) and the results are averaged.

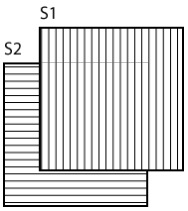


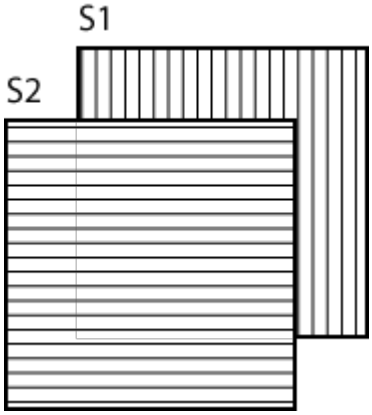
5. (3 pts each, 6 pts total) Suppose we have 2 square surfaces each with a different striped, opaque texture. S2 is in front of S1. The image below shows these surfaces as rendered with z-buffering turned on. What would the rendered image look like if z-buffering is turned off and where the render order is

   a) S1 is rendered first, then S2



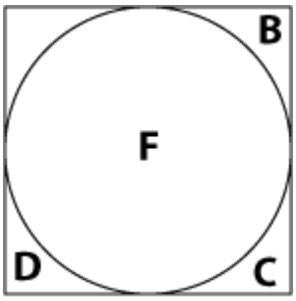
   b) S2 is rendered first, then S1.



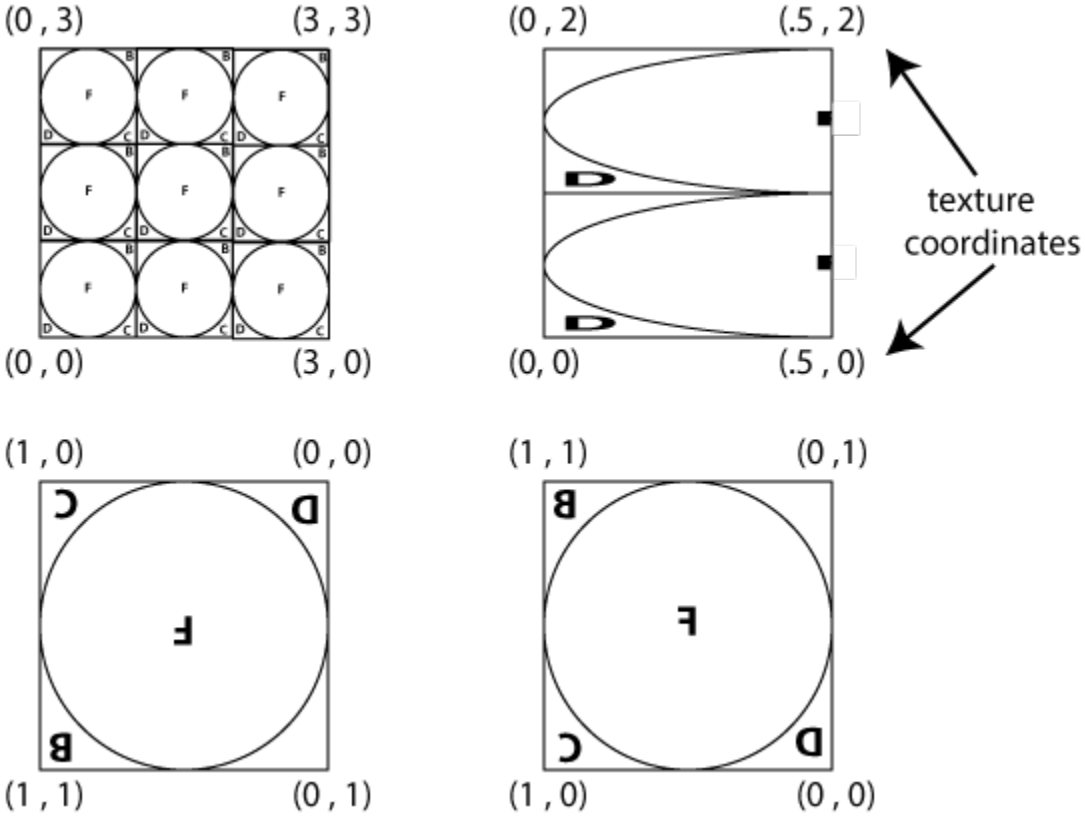


6. (4 pts each, 16 pts total) **Texture Coordinates**: Suppose you want to use the image below as a texture on a square (e.g. a quad modeled as 2 triangles, although this doesn't matter for this problem).

image to be used as a texture object:



For each of the following possible texture coordinates, draw how the texture will look on the square. *Assume that the texture wrap parameter is set to repeat in both dimensions*.

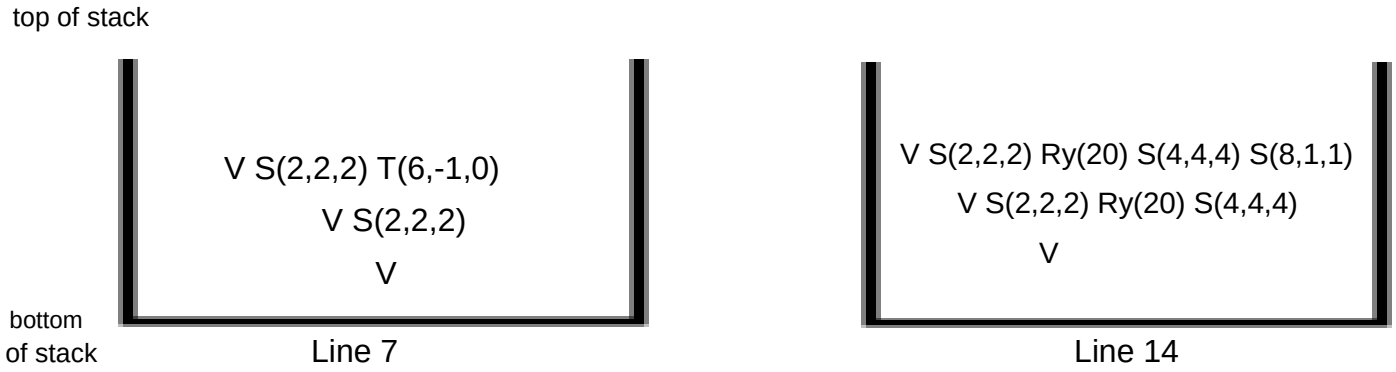


7. (12 pts total) Given the following sequence of commands in webGL:

```
Line  0: stack.clear(); // places identity on stack
Line  1: stack.multiply(V); // V is the view matrix
Line  2: stack.push();
Line  3: stack.multiply(scalem(2, 2, 2));
Line  4: stack.push();
Line  5: stack.multiply(translate(6, -1, 0));
Line  6: gl.uniformMatrix4fv(uModel_view, false, flatten(stack.top()));
Line  7: Shapes.drawPrimitive(Shapes.cube);

Line  8: stack.pop();
Line  9: stack.multiply(rotateY(20));
Line 10: stack.multiply(scalem(4, 4, 4));
Line 11: stack.push();
Line 12: stack.multiply(scalem(8, 1, 1));
Line 13: gl.uniformMatrix4fv(uModel_view, false, flatten(stack.top()));
Line 14: Shapes.drawPrimitive(Shapes.disk);
```

Show what is stored on the matrix *stack* at these lines?

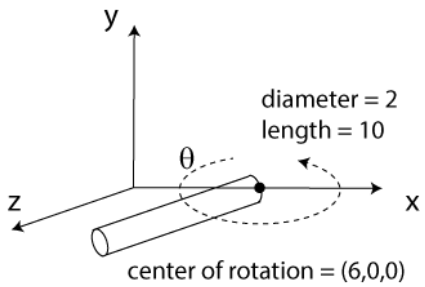


Use the space below as a scratch area if needed.

8. (12 pts total) Scene Graph. Suppose you have a rod that *rotates in the x-z plane*. It rotates by angle θ about one end fixed at the point (6,0,0). The rotation axis is parallel to the y-axis. The rod is a cylinder with length 10 and diameter 2.

Draw the <u>scene graph</u> for this rod: Assume that you have access to a cylinder primitive with diameter 1, height 1, centered at the origin, aligned with the y-axis.

Scale transformations should be indicated as **S(s_x,s_y,s_z)** where you fill in specific values for s_x, s_y, and s_z. Similarly, translations and rotations should have the form **T(t_x,t_y,t_z), Rx(angle), Ry(angle),** and **Rz(angle).**



There are many solutions. Below are 2 possibilities: