CSC 471 midterm 1 – Winter 2017

Name:____

READ ME FIRST

- Work individually! You may reference your course notes and use a calculator
- Don’t spend too much time on any one problem. This exam should take 80 minutes.
- Be neat
- Show how you got your answers!
- When in doubt, write down your assumptions
- You are allowed to use a calculator

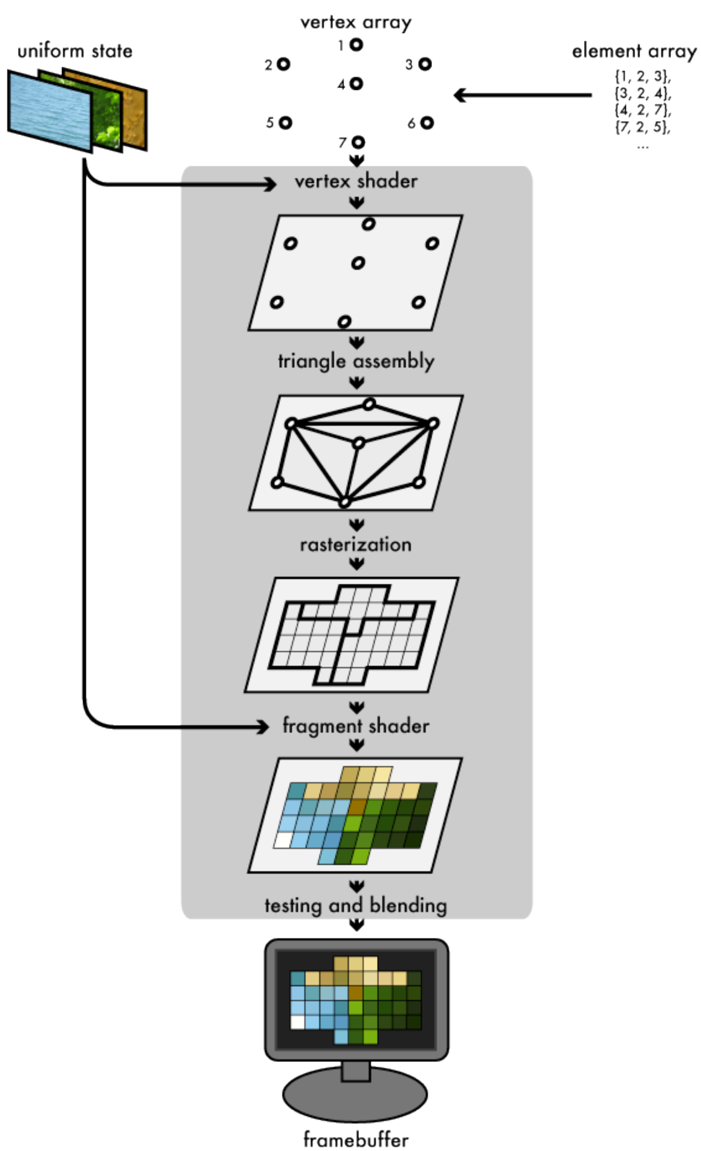
1	20 pts	Short answer	
2	10 pts	Vectors	
extra credit	2 pts		
3	30 pts	2D transform matrices	
4	15 pts	Transforms	
5	15 pts	More Transforms	
6	10 pts	Rasterization	
	100 pts	Grand total	

Short answer/ true & false questions (20 pts)

a) (1 pt) In a very general sense, the GPU can be viewed as a SIMD machine that allows a program to run the same 'vertex shader' program on multiple different vertices in parallel and then run a 'fragment shader' program on multiple fragments in parallel, thus speeding up the process of rendering computer graphics

True **False**

(b-f) Refer to the following figure and fill in the missing information – short answers (2 pts each):

<table>
<tr><td rowspan="5">
</td><td>(b) Describe what data is typically transferred from the CPU to the GPU for graphics applications:</td></tr>
<tr><td>c) Describe what computational task a vertex shader typically does?</td></tr>
<tr><td>d) Describe specifically what happens mathematically to all data passed from a vertex shader (out) into a fragment shader (in):</td></tr>
<tr><td>e) Describe what computational task a fragment shader typically does?</td></tr>
<tr><td>f) What are the value ranges of the data at the end of the graphic's pipeline?</td></tr>
</table>

g) (9 pts) If you wanted the iris of a CG creatures eyeball to track the mouse movement (i.e. appear to follow where the user currently has the mouse located), but always draw inside the creatures eye, which is defined by a sphere with the following equation:

$$f(x,y) = (x - x_c)^2 + (y - y_c)^2 + (z - z_c)^2 - r^2$$

with {xc, yc, zc} = {-1, 3, 1} and a radius of 3, what should the iris' {ix, iy, iz} location be (for an iris of radius 1), if you are given mouse coordinates transformed into world coordinates as follows: {mx, my} = {5, 2} – assume the z value should be the same as the eye's z values?(show your work with math)

{ix, iy, iz} =

2) Vectors (10 pts)

Given the following vectors: $\mathbf{v}^T$ =[7, -6, 5] and $\mathbf{u}^T$ = [-1, -2, 5]

Compute**:**

1) **(2 pts) v+u**

2) **(2 pts) v·u**

3) **(2 pts) If w = v+u,** What is the length of the vector **w?**

4) **(4 pts)** Write the normalized form of **w (from the part 3)** (i.e. write **w** as a unit length vector).

5) **(2 pt extra credit):** draw the vector **w** (accurately depicting length (ratio) and direction) as some part of a creature (make it clear which part of the creature is the vector) – you may define the units (i.e. inches, feet, etc.)

3) 2D transform matrices (30 pts)

Given the following 2D transform matrices:

$$m_0 = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad m_1 = \begin{bmatrix} 0.5 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad m_2 = \begin{bmatrix} .707 & -.707 & 0 \\ .707 & .707 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad m_3 = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

a) Name what type of 2D transformation is associated with each matrix and say something about the magnitude of the transform for x or y. **(4 pts total)**

m0:

m1:

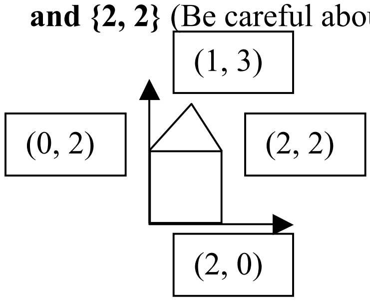
m2:

m3:

b) If these are 2D transforms, why are they 3x3 matrices? (Write 1-2 sentences) **(2 pts)**

c) Carefully compute m3*m1 (that is write out the composite matrix) (4 pts):

d))**(13 pts total)**
(4 pts) **Draw** the result of applying the composite matrix (from part (c) – i.e. m3*m1) to the following figure (draw the entire house transformed). (3 pts each) **Include coordinate labels for your completed drawing for the updated points {0, 2}, {1, 3} and {2, 2}** (Be careful about how you represent the 2D points as vectors of length 3

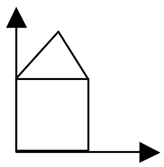


(3 pts) {0, 2} :

(3 pts) {1, 3} :

(3 pts) {2, 2} :

e) Now, only **draw** the result of applying three transforms: m1*m0 to the same figure (feel free to compute the composite matrix if that helps you, but it is not required). Be sure that your drawing includes a representation of the axes to clarify the house' exact final position: **(7 pts)**



4) Transforms (15 pts)

Assuming you have the following functions:

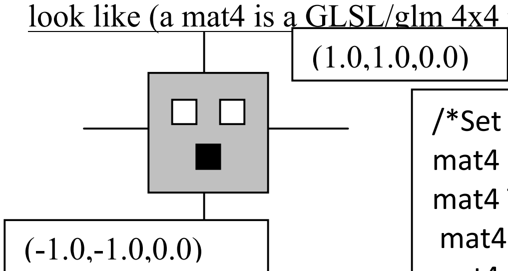
mat4 scale(float sx, float sy, float sz) {… } : returns a scale matrix

mat4 rotate(float angle, float ax, float ay, float ax) {…} : returns a rotation matrix by the given angle and axis [ax, ay, az]

mat4 translate(float tx, float ty, float tz) {…} : returns a translation matrix

*And assume the operator * is defined for matrix multiplication as expected*

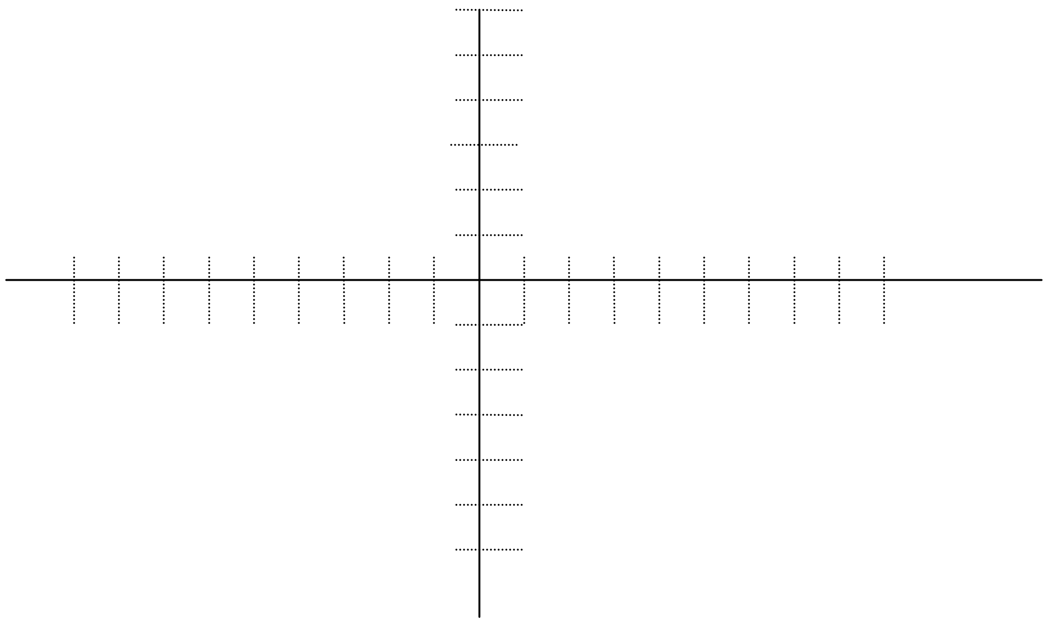
Carefully draw the result of the following OpenGL/GLSL code assuming that the DrawRobotFace() function draws the complete image below (i.e. one grey box with sides of length 2 with three small sub-boxes inside with sides of length 0.5: white eyes and a black mouth). Recall that rotations are specified as counter-clockwise. **Carefully** read all the code below before drawing and be sure that it is clear what the final drawing will look like (a mat4 is a GLSL/glm 4x4 matrix – as expected):




```
/*Set up the first matrix */
mat4 Scale = scale(2, 1, 1);
mat4 Trans = translate( 1, 1, 0);
 mat4 Rot = rotate( -90, 0, 0, 1);
mat4 Model = Scale*Rot*Trans;
/*send matrix to the vertex shader */
glUniformMatrix4fv(prog->getUniform("MV"), 1, GL_FALSE, Model);
 /* Draw */
 DrawRobotFace ();

/*Set up the second matrix */
mat4 Scale = scale( 1, 2, 1);
mat4 Trans = translate( 0, 2, 0);
 mat4 Rot = rotate( 45, 0, 0, 1);
 mat4 Model = Trans*Rot*Scale;
/*send matrix to the vertex shader */
glUniformMatrix4fv(prog->getUniform("MV"), 1, GL_FALSE, Model);
/* Draw */
```

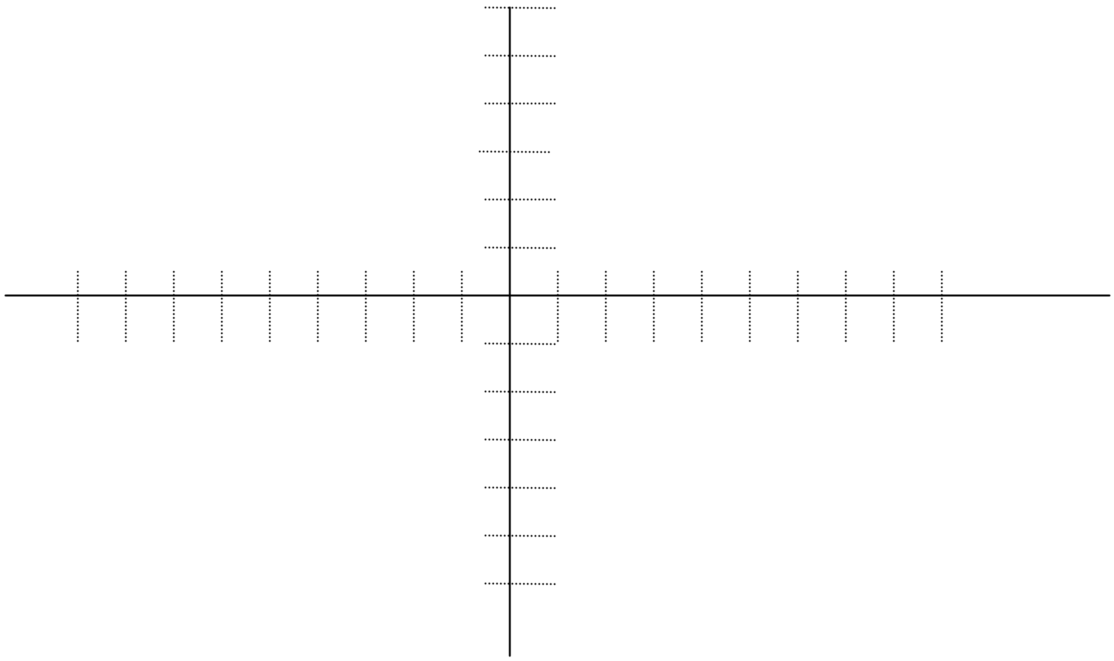
Complete your drawing on the next page



4) More Transforms (15 pts)

Given the following code snippet, that creates a hierarchical model for a robot chest and one arm with lower and upper arm portions (3 shapes total, using drawing primitives exactly like what you used for lab 6 – i.e. a cube that spans {-1, -1, -1} to {1, 1, 1} and the matrix stack provided with lab 6). Correctly draw the current position of the chest and arm (both upper and lower portions) - **Complete your drawing on the next page**

```
MV->pushMatrix();
     MV->loadIdentity();
     MV->translate(Vector3f(0, 0, -5));
     MV->scale(Vector3f(0.75, 0.75, 0.75));
     glUniformMatrix4fv(prog->getUniform("MV"), 1, GL_FALSE, MV);
     shape->draw(prog);
     MV->pushMatrix();
          MV->translate(Vector3f(-1, 1, 0));
          MV->rotate(90, Vector3f(0, 0, 1));
          MV->translate(Vector3f(-.75, 0, 0));
          MV->pushMatrix();
               MV->translate(Vector3f(-.75, 0, 0));
               MV->rotate(-45, Vector3f(0, 0, 1));
               MV->translate(Vector3f(-.75, 0, 0));
               MV->scale(Vector3f(0.75, 0.25, 0.25));
               glUniformMatrix4fv(MV, 1, GL_FALSE, MV); //pseudo code
               shape->draw(prog);
          MV->popMatrix();
          MV->scale(Vector3f(0.75, 0.25, 0.25));
          glUniformMatrix4fv(prog->getUniform("MV"), 1, GL_FALSE, MV);
          shape->draw(prog);
      MV->popMatrix();
 MV->popMatrix();
```

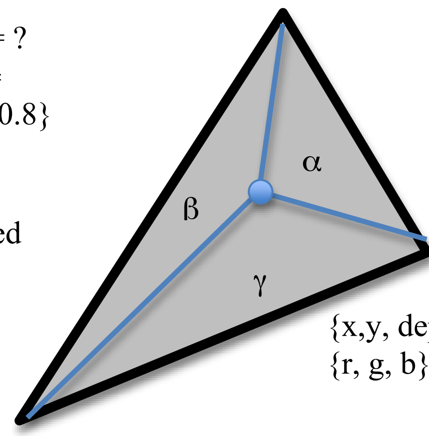


5) Rasterization (10 pts total):

If you have a triangle converted to window coordinates with the following coordinates, (including depths and colors) – **given the associated Barycentric coordinates** (ie do not compute them, use what is given):

{x,y, depth} = {214, 69, 4}
{r, g, b} = {0, 0, 1}

{x, y, z} = ?
{α, β, γ}=
{0.2, 0.0, 0.8}



a) (3 pts) What are the coordinates for the associated interpolated vertex?:

{x,y, depth} = {242, 42, 9}
{r, g, b} = {0, 1, 0}

b) (3 pts) What is the interpolated color?:

{x,y, depth} = {104, 14, 5}
{r, g, b} = {1, 0, 0}

d) (4 pts) Assuming the current value stored in the depth buffer/z-buffer for the associated pixel is 4.3, would the frame buffer/color buffer be updated with the new color? **Assuming the z values specified are distances measured from the camera – thus smaller values are closer to the camera.**