Introduction to Computer Graphics
Midterm Examination

Name:_____

Honor Pledge: This is a closed-book, closed-notes, independent exam. Please sign the honor pledge: On my honor as a student, I have neither given nor received information on this exam. Signed, _____

Display Technology

1) What is a scanline?

A row of pixels on a raster display. Somewhat true to call a row of pixels in the frame buffer a scanline.

2) Frame buffers are described as having a certain depth. To what does the depth refer?

The number of bits used to store the color for each pixel.

3) What is stored in the frame buffer when a color map lookup table is used?

A number that corresponds to an index into the color map lookup table. The lookup table is an array of color descriptions (RGB). For example, index 0 could refer to the color BLACK, while the index 1 could refer to the color SKY_BLUE. The colors in the color map are reconfigurable.

4) What is a shadow mask?

A shadow mask is a perforated metal sheet that between the electron gun and the phospher coating on the screen. The shadow mask serves to sharpen the RGB beams that are emitted from the guns.

Mathematical Foundations

5) Given two points, (a, b) and (c, d), provide the following equations of a line:

Slope-intercept:
y = mx + intercept
m = (d-b) / (c-a)
intercept: Substitute a for x and b for y (or c for x and d for y)
b = [(d –b) / (c – a)] * a + intercept
intercept = b – a * [(d-b) / (c-a)]

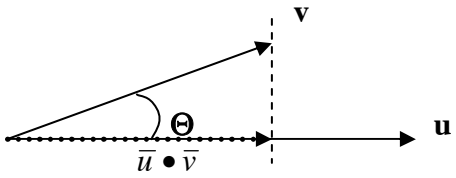
Parametric:
x = a + t (c-a)
y = b + t (d-b)

6) Provide a geometric (use a picture) and an algebraic definition of the dot product of two vectors, $[u_x, u_y, u_z]$ and $[v_x, v_y, v_z]$

Algebraic:

$$\begin{aligned} \bar{u} \bullet \bar{v} &= |\bar{u}||\bar{v}|\cos(\Theta) \\ &= u_x v_x + u_y v_y + u_z v_z \end{aligned}$$

Geometric: Two vectors define a plane
Here, we've aligned the defined plane with the paper



7) What is an orthonormal matrix?

A matrix is orthonormal if:
The row and column vectors have unit length
The row and column vectors are pairwise orthogonal (their dot product is 0)

8) Compute the inverse of the following orthonormal matrix:

$$\begin{bmatrix} 0.71 & 0.0 & 0.71 \\ 0.0 & 1.0 & 0.0 \\ -0.71 & 0.0 & 0.71 \end{bmatrix}^{-1} = \begin{bmatrix} 0.71 & 0.0 & -0.71 \\ 0.0 & 1.0 & 0.0 \\ 0.71 & 0.0 & 0.71 \end{bmatrix}$$

OpenGL

9) Use glPushMatrix(), glPopMatrix(), glRotatef(), glTranslatef(), and glutWireCube() to draw the hour marks on the face of a clock. The hour marks should look be unit cubes of size 1.0 and their centers should be 3.0 units from the center.

```
for (j=0; j<12; j++) {
      glPushMatrix ();
            glRotatef(30.0*j, 0.0, 0.0, 1.0);
            glTranslatef (3.0, 0.0, 0.0);
            glutWireCube (1.0);

      glPopMatrix ( );
}
```

10) Given a variable, v:

Float v[3];
v[0] = 1.0; v[1] = 0.0; v[2] = 0.0;

Complete the following two GL functions that will use v as a parameter:

glVector3f_____);
glVector3fv(_____);

glVector3f (v[0], v[1], v[2]);
glVector3fv (v);

Rasterization

11) Given two integer endpoints for a line, and the following rasterization algorithm:

```
public void lineSimple(int x0, int y0,
                       int x1, int y1, Color color)
{
    int pix = color.getRGB();
    int dx = x1 - x0;
    int dy = y1 - y0;

    raster.setPixel(pix, x0, y0);
    if (dx != 0) {
        float m = (float) dy / (float) dx;
        float b = y0 - m*x0;
        dx = (x1 > x0) ? 1 : -1;
        while (x0 != x1) {
            x0 += dx;
            y0 = Math.round(m*x0 + b);
            raster.setPixel(pix, x0, y0);
        }
    }
}
```

a) Describe a case where the algorithm will poorly render a line between the two points.

When the slope of the line is greater than 1.0.

12) One triangle rasterization technique is called ***Edge Equations.*** List three optimizations implemented in the following two code segments:

```
findBoundingBox(&xmin, &xmax, &ymin, &ymax);
setupEdges (&a0,&b0,&c0,&a1,&b1,&c1,&a2,&b2,&c2);

for (int y = yMin; y <= yMax; y++) {
     for (int x = xMin; x <= xMax; x++) {
          float e0 = a0*x + b0*y + c0;
          float e1 = a1*x + b1*y + c1;
          float e2 = a2*x + b2*y + c2;
          if (e0 > 0 && e1 > 0 && e2 > 0)
               setPixel(x,y);
     }
}
```

```
findBoundingBox(&xmin, &xmax, &ymin, &ymax);
setupEdges (&a0,&b0,&c0,&a1,&b1,&c1,&a2,&b2,&c2);
float e0 = a0*x + b0*y + c0;
float e1 = a1*x + b1*y + c1;
float e2 = a2*x + b2*y + c2;
int xflag = 0;
for (int x = xMin; x <= xMax; x++) {
     if (e0|e1|e2 > 0) {
          setPixel(x,y);
          xflag++;
     } else if (xflag != 0) break;
          e0 += a0; e1 += a1; e2 += a2;
}
```

1) Removed explicit line equation calculation inside the loop. Replaced with discrete difference equation. Removes multiplies

2) Condition check simplified to bitwise OR and one zero-comparison

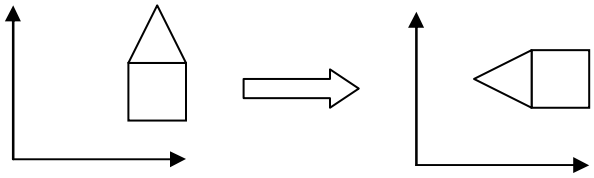
3) xflag allows for early termination when end of line is reached
4) Nested `for` loops reduced to one `for` loop

Geometric Transformations

16) Why do we use a 4x4 homogenous coordinate transformation matrix when describing translations, rotations, and scales, etc.?

A 4x4 homogeneous transformation matrix allows rotation and translation transformations to be unified in one matrix representation. This facilitates concatenation of rotations, scales, translations, etc. Use of the homogeneous coordinate in the lower-right corner of the 4x4 matrix facilitates perspective and parallel projection calculations.

18) Provide a 3x3 matrix that will compute the new vertices of a planar house centered at **[10, 5]** after a rotation of 90 degrees about its center:



First, translate house to (0,0). Then rotate 90 degrees. Then translate house back to (10,5).

$$\begin{bmatrix} 1 & 0 & 10 \\ 0 & 1 & 5 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(90) & -\sin(90) & 0 \\ \sin(90) & \cos(90) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -10 \\ 0 & 1 & -5 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & -1 & 15 \\ 1 & 0 & -5 \\ 0 & 0 & 1 \end{bmatrix}$$

18) (Worth double points) Calculate a 4x4 matrix that rotates points about the vector A = [1, 1, 1] by 90 degrees. Show your work.

Rotate about y-axis –45 degrees to bring vector into YZ plane
Rotate about x-axis 45 degrees to bring vector onto z-axis
Rotate about z-axis by 90 degrees
Unwrap x-rotation
Unwrap y-rotation

Note: Right-handed coordinate system specifies that positive rotations are Counter-Clockwise (CCW) when looking from the positive axis towards the origin

$$\begin{bmatrix} \cos(45) & 0 & \sin(45) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(45) & 0 & \cos(45) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(-45) & -\sin(-45) & 0 \\ 0 & \sin(-45) & \cos(-45) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(90) & -\sin(90) & 0 & 0 \\ \sin(90) & \cos(90) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(45) & -\sin(45) & 0 \\ 0 & \sin(45) & \cos(45) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(-45) & 0 & \sin(-45) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(-45) & 0 & \cos(-45) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Viewing in 3-D

19) What is the viewing frustum?

The volume of world space rendered on the screen. This volume is defined by top, bottom, left, right, near, and far clipping planes.

20) (Worth double points) Given a camera position ***P***, a vector normal to the image plane ***N***, and an up vector $\boldsymbol{V_{up}}$, describe how to convert a point W in world coordinates to a point in camera coordinates. Provide your final answer in the form of one (or a product of many) transformation matrix. Hint, the origin in camera coordinates is located at ***P*** and the world coordinate axes must be rotated to align with the camera's coordinate axes.

1) Translate the camera to the origin
2) Rotate the world so (x,y,z) aligns with camera's (u,v,w) axes
   a. The normalized **N** vector defines the w-axis
   b. The normalized cross product of the w-axis and $\mathbf{V}_{up}$ defines the u-axis
   c. The normalized cross product of the v-axis and the w-axis defines the v-axis
3) The transformation matrix is constructed by making each row equal to the u, v, and w axis vectors. The right column is the translation vector. The bottom row is all 0's with a final 1.

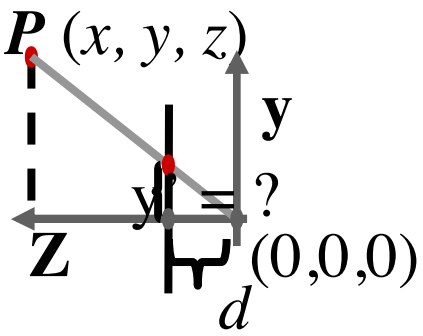
$$W = \frac{N}{\|N\|}$$

$$U = \frac{WxV_{up}}{\|WxV_{up}\|}$$

$$V = VxW$$

$$\begin{bmatrix} U_x & U_y & U_z & -U \bullet P \\ V_x & V_y & V_z & -V \bullet P \\ W_x & W_y & W_z & -W \bullet P \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

21) Calculate y' in the following figure



$$\frac{y}{z} = \frac{y'}{d} \qquad y' = \frac{yd}{z}$$

22) The 4x4 perspective projection matrix, $\mathbf{M}_{per}$, converts points from the camera coordinate system to the view plane coordinate system. It performs the calculation from the previous question. What is the 4x4 matrix, $\mathbf{M}_{per}$?

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/d & 0 \end{bmatrix}$$

Color and Light

23) What is the name of the region of the retina where our vision is the sharpest?

Fovea

24) What color are we the least sensitive to? Red, Green, or Blue

Blue

25) Why do monitors need to perform gamma correction?

The perceived intensity (brightness) of a pixel is not linearly related to gun voltage.

26) If you were an artist, trying to carefully color polygons by hand, which color space would you prefer to work in, RGB or HSV? Why?

I would pick HSV because it is intuitive to select a first select a hue and subsequently pick a saturation and a value. RGB, by contrast, is more difficult to tinker with. Blending between two different colors in RGB space is not as easily accomplished as when blending between two HSV values.

27) Given:

an incoming light intensity, **I**
a surface normal, **N**
a vector from the surface to the light source, **L**
a vector from the surface to the viewer's eye, **V**
specular constant, $\mathbf{k_s}$

define the intensity of the light reaching the viewer as a result of the specular effects of the Phong lighting equation. Your definition may only use these variables.

$I_{viewer} = k_s I_{light} (\cos(\phi))^n$
$I_{viewer} = k_s I_{light} (V \bullet R)^n$
$R = 2 (N \bullet L) N - L$

28) Both Gouraud and Phong Shading interpolate along polygon edges to compute intensities. But the two shading models interpolate different things.

a) What does Gouraud Shading interpolate along edges?

Color / Intensity of the vertices at the edge endpoints.

b) What does Phong Shading interpolate along edges?

Vertex normals of the edge endpoints.

29) True or False? Raytracing produces viewer-independent lighting solutions.

False

30) True or False? Radiosity produces accurate specular highlights

False

31) (Extra Credit) Who is credited with inventing **The Rendering Equation**?

Jim Kajiya