

Name: \_\_\_\_\_

**Midterm Exam**

Introduction to Computer Graphics

CS 445/645

October 15, 2002

Professor Brogan

Time Limit: 1 hour and 15 minutes

Reference Materials: Closed book, closed notes, no calculator

Please write and sign the honor pledge:

Notational convention: bold lower case means vector and bold upper case means matrix. (so  $q$  is a scalar,  $\mathbf{q}$  is a vector, and  $\mathbf{Q}$  is a matrix)

All coordinate systems will be assumed right-handed.

Note that some questions take longer than others and some questions are worth more than others. Use your time wisely.

Because you're not allowed to use calculators, you should feel free to leave your answers in intermediate forms: unmultiplied, including trig functions, including square roots, cross products etc.

**Display Technology**

1. (2 points) What critical piece of hardware finally became sufficiently affordable that vector displays became undesirable?
2. (2 points) What is the name of this new display process that supplanted vector displays?

3. (2 points) Did the new system make it easier or more difficult to render a line segment? Why?

Did the system make it easier or more difficult to render solid shapes? Why?

4. (2 points) How many flashes of light (flickers) per second do you see on the screen of an American movie theater?

How many distinct images can be displayed per second?

5. (2 points) How many fields per second do you see on an American television set?

### **Mathematical Foundations**

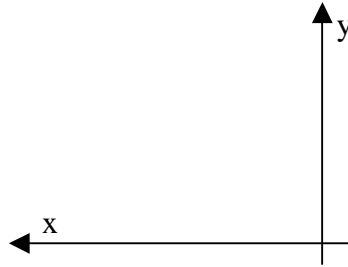
6. (4 points) Let  $\mathbf{v} = [3, 5, 2]$

What is the length of  $\mathbf{v}$ ?

What angle does  $\mathbf{v}$  make with the x axis =  $[1, 0, 0]$

7. (6 points) Use any representation to build the equation of a line from  $\mathbf{p}_1 = [1, 2]$  to  $\mathbf{p}_2 = [3, -2]$ .

8. (2 points) Provided the definition of the positive x and y axes below, would the z axis of a right hand coordinate system come out of the page *towards* you or point *behind* the page (circle one)?

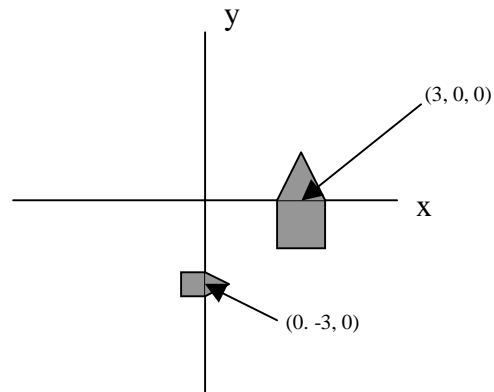


9. (4 points) What does it mean if two vectors are orthogonal? How can you determine if two vectors are orthogonal?
10. (4 points) Is the set of vectors  $\{[1, 1, 0], [2, 3, 0], [0, 1, 1], [1, 1, 1]\}$  a basis of the 3-D Cartesian coordinate space? Either explain why it is not, or provide the coordinates of  $(4, 2, 0)$ .

### **Transformations**

11. (6 points) For the eye position  $\mathbf{e} = [0, 2, 0]$ , a gaze vector  $\mathbf{g} = [0, -1, 0]$ , and a view-up vector  $\mathbf{t} = [1, 1, 0]$ , what is the camera transformation matrix?

12. (6 points) Calculate a chain of 4 x 4 matrices that, when post-multiplied by the vertices of the house, will translate and rotate the house from (3, 0, 0) to (0, -3, 0). The transformation must also scale the size of the house by half.

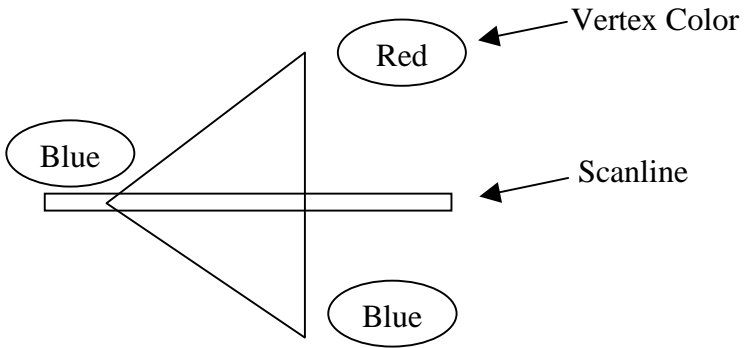


### Rasterization

13. (6 points) In Bresenham's algorithm, we saw the efficiency of incremental evaluation. For the function,  $f(x) = y = x^2 - 3$ , compute the incremental evaluation function.

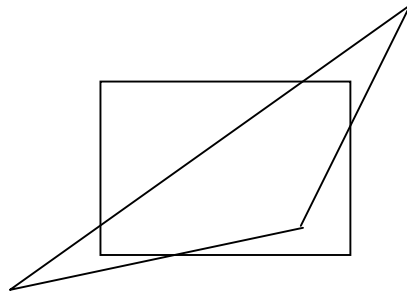
14. (4 points) For the following triangle, select the color the indicated scanline would be (described from left to right).

- a) Entirely blue
- b) Entirely red
- c) Blue to purple
- d) Blue to purple to red
- e) Red to Blue
- f) Red to Purple



15. (4 points) When using a parity counter to rasterize general polygons, provide an example of a type of vertex that can cause rendering errors unless treated carefully and consistently.

16. (2 points) Clip the following triangle to the viewport. Draw an X through any deleted vertices and use circles to indicate any new vertices you create.



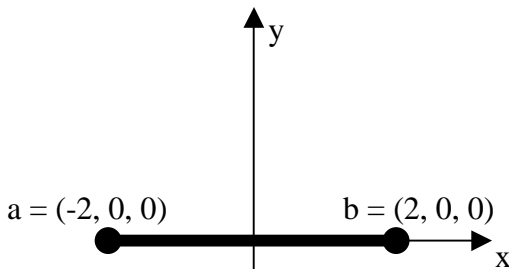
17. (4 points) When performing Cohen-Sutherland line clipping, how do we use the outcodes to check for trivial rejection (what exact operation and comparison would you use)?

## OpenGL

18. (4 points) What does `glPushMatrix()` do and why would one use it?

19. (2 points) Does OpenGL work more reliably with convex or concave polygons?

20-a. (6 points) For assignment 1, you had to use transformation matrices to move a firetruck's ladder. You are provided with a function called `drawLadder()` that places the ladder in the following position when called with a MODELVIEW matrix equal to identity.



You must write the sequence of OpenGL transformations that will:

- 1) Move endpoint  $a$  to the coordinate  $(10, 10, 0)$
- 2) Raise the ladder (so endpoint  $b$  is above  $a$ ) by 30 degrees
- 3) Rotate the ladder about the  $y$  axis by 45 degrees so endpoint  $b$  comes out of the page towards the reader

20-b. (6 points) How would you augment your code to render a second, identical, segment that is collinear with, but extends, the first segment you rendered. Either write your new sequence of OpenGL transformations in the space provide below or *clearly* identify your changes in your answer above. Assume the second ladder segment has two endpoints labeled  $c$  and  $d$ . You want the location of endpoint  $c$  to equal the location of endpoint  $b$  and endpoint  $d$  should be eight units from coordinate  $(10, 10, 0)$ .

21. (4 points) What is a viewing frustum?

22. (4 points) Why is the triangle strip a more desirable geometric primitive than a list of triangles?

23. (4 points) Given the following components from a 3-D scene:

**a**: an axis of rotation

**v**: a point to be rotated

**M<sub>1</sub>**: a matrix that causes a rotation about **a** by 1.0 degrees

**M<sub>2</sub>**: a matrix that causes a rotation about **a** by 50 degrees

Which statement best compares the accuracy of the transformed point, **v'**, computed by *Operation A* versus *Operation B*?

- a) Operation A is more accurate
- b) Operation B is more accurate
- c) The two Operations are equally accurate

Operation A

$$\mathbf{v}' = \mathbf{M}_2\mathbf{v}$$

OR

Operation B

```

MultiplyProduct = identity;
for (i=0; i<50; i++) {
  MultiplyProduct = M1M';
  M' = MultiplyProduct;
}
v'=M'v

```

**Viewing in 3-D**

24. (4 points) What type of camera is created by moving the center of projection (COP) infinitely far from the projection plane?

25. (4 points) Write the perspective projection matrix. Multiply it by the given homogeneous point to demonstrate how it generates pixel coordinates that reflect perspective foreshortening.

$$\begin{bmatrix} \phantom{0} \\ \phantom{0} \\ \phantom{0} \\ \phantom{0} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ W \end{bmatrix} = \begin{matrix} \text{Transformed 4-D} \\ \text{Point} \end{matrix} = [ \phantom{0}, \phantom{0}, \phantom{0}, \phantom{0} ] \text{ becomes } \begin{matrix} \text{Pixel} \\ \text{Value} \end{matrix} [ \phantom{0}, \phantom{0} ]$$