CPSC 314
Assignment 0

due: Wednesday, January 18, 2006 (in class)
Worth 7% of your final grade.

Purpose: (a) to review your math skills; (b) to get you thinking about how matrices can be used to implement geometric transformations, such as translations, rotations, and scales; (c) to get you to compile a simple OpenGL program and understand it.

Answer the questions in the spaces provided on the question sheets. If you run out of room for an answer, continue on the back of the page.

Name: _____

Student Number: _____

Question 1	/ 19
Question 2	/ 5
Question 3	/ 4
Question 4	/ 13
Question 5	/ 4
Question 6	/ 6
TOTAL	/ 51

1. Dot Products and Cross Products

$a = \begin{bmatrix} 4 \\ 1 \\ 2 \end{bmatrix}$, $b = \begin{bmatrix} -2 \\ 4 \\ 0 \end{bmatrix}$, $c = 3$.

(a) (11 points) For each of the following expressions, either compute the result or declare that it is a nonsense expression.

$a \cdot b$

ab^T

$b^T a$

$a^T b$

ab

$a \times b$

$b \times a$

$a \times a$

ac

$ca \times b$

$a^T cb$

(b) (1 point) Which of the above algebraic expressions are equivalent to $b \cdot a$?

(c) (2 points) Give a non-zero vector c such that $a \cdot c = 0$. Is your answer unique ?

(d) (1 point) Compute a normalized version of the vector a, i.e., compute $\hat{a} = \frac{a}{|a|}$.

(e) (1 point) Suppose that three 3D vectors a, b, c all have unit length and they are all orthogonal to each other, i.e., $a \cdot b = 0$, $a \cdot c = 0$, $b \cdot c = 0$. If you know that $a \times b = c$, then determine an expression for $a \times c$. Assume a right-handed coordinate system.

(f) (1 point)

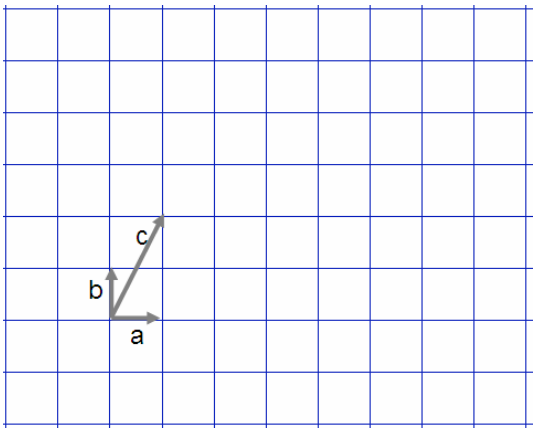
$$A = \begin{bmatrix} 2 & 0 & -2 \\ 0 & 2 & 4 \\ 0 & 0 & 1 \end{bmatrix}, B = \begin{bmatrix} 0 & 1 & -1 \\ -1 & 0 & -1 \\ 0 & 0 & 1 \end{bmatrix}.$$

Compute $C = AB$.

(g) (1 point) Express $(AB)^T$ in terms of A^T and B^T.

(h) (1 point) Given a simplified algebraic expression for $A^T A B B^{-1} A^{-1}$.

2. Vectors



(a) (1 point) Draw the vector e, where $e = 3b - 2a$.

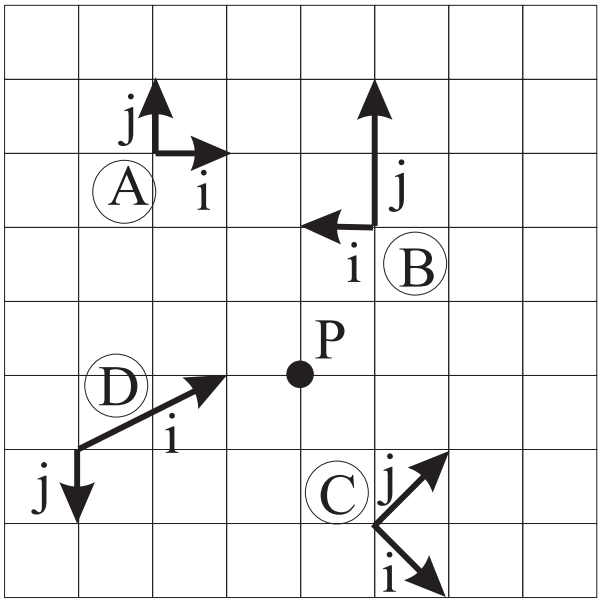
(b) (1 point) Draw the vector f, where $f = 3a + 2b - 3c + c - a + b$.

(c) (1 point) Express c in terms of a and b.

(d) (1 point) Express a in terms of b and c.

(e) (1 point) Express b in terms of a and c.

3. (4 points) Points Expressed in Multiple Coordinate Frames



Specify the coordinates of point P with respect to coordinate frames A, B, C, and D.

4. Geometric Transformations

In computer graphics, we often choose to represent points in *homogeneous coordinates* of the form:

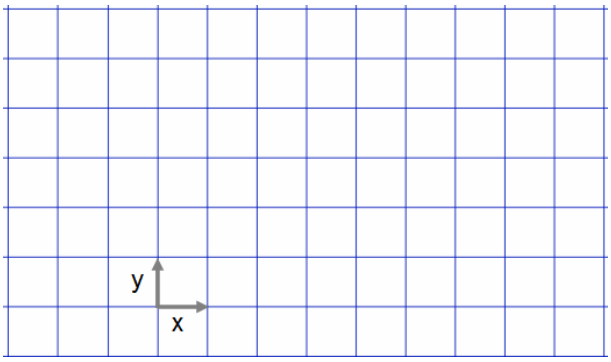
$P \begin{bmatrix} x \\ y \\ h \end{bmatrix}$ in 2D, or

$P \begin{bmatrix} x \\ y \\ z \\ h \end{bmatrix}$ in 3D, where $h = 1$ for points. This can be viewed as a trick that lets us to represent translations using matrices.

(a) (4 points) The following 5 points represent a simple geometric model of the letter 'P'. Draw the untransformed 'P' by plotting the (x,y) values of the given points $P_1 ... P_5$ and drawing a line that connects them in the given order.

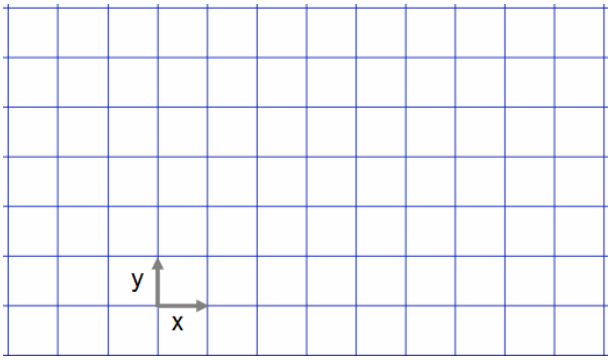
Now multiply each of the points by the transformation matrix T, i.e., $P'_n = TP_n$. Draw the resulting transformed 'P'. Describe the effect of the transformation T in terms of translation, rotation and scaling.

$$P_1 \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, P_2 \begin{bmatrix} 0 \\ 2 \\ 1 \end{bmatrix}, P_3 \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}, P_4 \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}, P_5 \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}, T = \begin{bmatrix} 0 & -2 & 6 \\ 2 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix}.$$



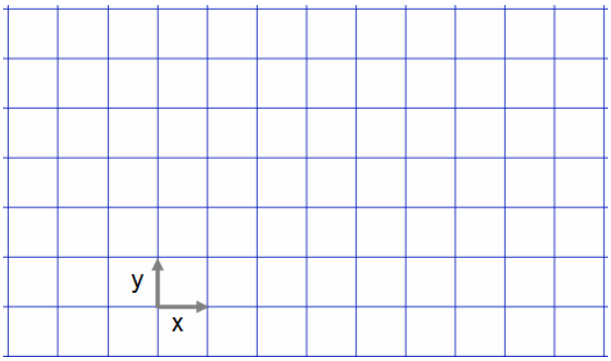
(b) (3 points) In the same way, determine what the following transformation matrices do.

$$T = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



(c) (3 points)

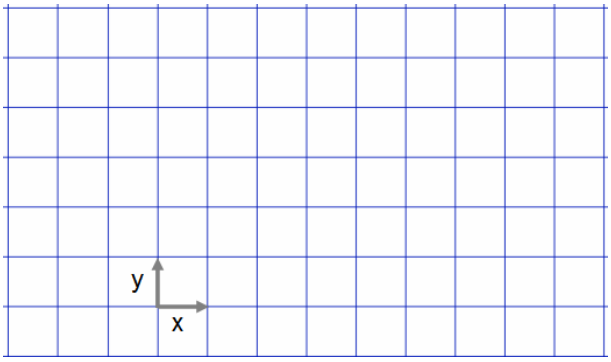
$$T = \begin{bmatrix} -1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



(d) (3 points)

$$T = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

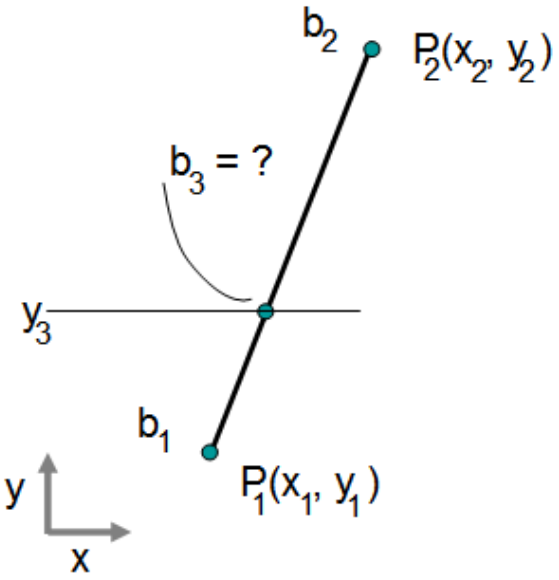
For this example, you might try drawing a line from each unprojected point to its corresponding projected point in order to determine what the transformation does.



5. Linear Interpolation

(a) (2 points) On March 15, the temperature was $5°C$. On March 27, the temperature was $22°C$. What was the temperature on March 23, assuming a linear variation of temperature over time ?

(b) (2 points) As shown below, a line is to be drawn from pixel $P_1(x_1, y_1)$ to $P_2(x_2, y_2)$. Suppose that the brightness of any point along the line is described by a real-valued number, $b \in [0, 1]$, where $b = 0$ represents black, and $b = 1$ represents white, and in between are the various shades of grey. Suppose that we know the brightness of the line at its endpoints, i.e., we know b_1 and b_2, and that the brightness to vary linearly in between the endpoints. What is the brightness of the line at a given value of y_3 ?



6. (6 points) Coding

   This is an exercise that gets you started using OpenGL. OpenGL is not installed on all the CS undergrad machines. The code has been tested on the Linux machines in the CICSR 011 lab. When compiling the code on your own machine (Windows or Linux), you will need to install the "glut" library. This is generally quite easy and will let you do the remaining assignments on your own machine as well.

   A printed copy of the code is included at the end of this assignment. The TA will help you walk you through the code in your first lab.

   You need to do the following:

   - download and untar a0.tar.gz (see the course website)
   - build the executable using 'make' and run it.
   - make the triangle green at the top and white at the bottom
   - add a second triangle that cuts through the original triangle at right angles in order to get a crude approximation of something that looks a bit like a tree. The original triangle sits in the plane defined by $z = 0$. Your new triangle should sit in the plane defined by $x = 0$.
   - hitting 'q' should quit the program. Calling the function 'exit(0)' will accomplish this.
   - hitting ' ' (i.e., a space) should toggle the animation on and off.
   - make the scene rotate around the x-axis instead of the y-axis

   Hand-in Instructions

   - Hand in a printed copy of your code attached to the rest of your assignment.
   - Create a root directory for this course in your account, called cs314. Later all the assignment handin files should be put in this directory.
   - For assignment 0, create a folder called assn0 under cs314 and put all the source files that you want to handin in it. For this assignment it should only contain the openGLDemo.cpp file.
   - The assignment should be handed in with the exact command:

     ```
     handin cs314 assn0
     ```

     This will handin your entire assn1 directory tree by making a copy of your assn1 directory, and deleting all subdirectories! (If you want to know more about this handin command, use: man handin)

```
#include <stdio.h>
#include <GL/glut.h>
#include <math.h>

float angle = 0;

void DrawWorld()
{
  glMatrixMode( GL_PROJECTION );
  glLoadIdentity();
  glMatrixMode( GL_MODELVIEW );
  glLoadIdentity();

    // set the background colour to be used, then apply it
  glClearColor(0,0,0,0);
  glClear( GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    // rotate around the y-axis by 'angle' degrees
  glRotatef(angle,0,1,0);

    // draw a triangle
  glBegin( GL_TRIANGLES );

  glColor3f(1,0,0);
  glVertex3f( 0.0f, 0.5f, 0.0f );

  glColor3f(0,1,0);
  glVertex3f( -0.5f, -0.5f, 0.0f );

  glColor3f(0,0,1);
  glVertex3f( 0.5f, -0.5f, 0.0f );

  glEnd();

  glColor3f(0.8, 0.8,0.8);      // choose a light-grey color
  glTranslatef(0,0,0.25);        // move torus back slightly
  glutSolidTorus(0.1,0.5,8,20); // draw a solid torus

  glColor3f(0,1,0);           // green
  glTranslatef(1,0,0);        // move one unit to the right
  glutWireTeapot(0.4);        // draw a wireframe teapot of size 0.4

  glColor3f(1,1,0);           // yellow
  glTranslatef(0,0.5,0);      // move up
  glutWireTeapot(0.2);        // draw a wireframe teapot of size 0.2
```

```
  glutSwapBuffers();          // force the draw-buffer to be displayed
}

void Idle()
{
  angle += 0.50;
  glutPostRedisplay();
}

void keyhit(unsigned char key, int x, int y)
{
  switch(key) {
    case 'a':
      printf("you hit the 'a' key\n");
      break;
  }
}

int main(int argc, char **argv)
{
    // create window and rendering context
  glutInit( &argc, argv );
  glutInitDisplayMode( GLUT_RGB | GLUT_DOUBLE | GLUT_DEPTH );
  glutInitWindowSize( 640, 480 );
  glutCreateWindow( "openGLDemo" );
  glutDisplayFunc( DrawWorld );
  glutIdleFunc(Idle);
  glutKeyboardFunc(keyhit);
  glEnable(GL_DEPTH_TEST);

    // pass control over to GLUT
  glutMainLoop();
  return 0;       // never reached
}
```