GRAPHICS PROGRAMMING: OpenGL

```
STUDENT REGISTRATION ID (NRP):
NAME:
CLASS:
#include "GL/freeglut.h"
#include "GL/gl.h"
float angle = 0;
void renderFunction()
   glClearColor(0.0, 0.0, 0.0, 0.0);
   glClear(GL_COLOR_BUFFER_BIT);
   angle += 0.0001;
   glRotatef(angle, 0, 0, 1);
   glColor3f(1.0, 1.0, 1.0);
   glOrtho(-1.0, 1.0, -1.0, 1.0, -1.0, 1.0);
   glBegin(GL_POLYGON);
       glVertex2f(-0.5, -0.5);
       glVertex2f(-0.5, 0.5);
       glVertex2f(0.5, 0.5);
       glVertex2f(0.5, -0.5);
   glEnd();
   glFlush();
   glutPostRedisplay();
}
int main(int argc, char** argv)
{
   glutInit(&argc, argv);
   glutInitDisplayMode(GLUT_SINGLE);
   glutInitWindowSize(500,500);
   glutInitWindowPosition(100,100);
   glutCreateWindow("OpenGL - First window demo");
   glutDisplayFunc(renderFunction);
   glutMainLoop();
   return 0;
}
ACTIVITY
Which part of the program is responsible for (put comments in the program):
* Making the background black
* Making the square white
* Drawing the square
* Rotating the square
Which part of the program you don't understand? Write it down.
Can you:
* Make a cube instead of a square?
* Move the cube to the right, and move it back to the left instead of rotate
it?
```

```
STUDENT REGISTRATION ID (NRP): _____
NAME:
CLASS:
// Source: https://www.ntu.edu.sg/home/ehchua/programming/opengl/CG_Examples.html
// Compile: g++ 3d.cpp -o 3d -lglut -lGL -lGLU
#include "GL/glut.h"
                                                            glBegin(GL_TRIANGLES);
GLfloat anglePyramid = 0.0f;
                                                              glColor3f(1.0f, 0.0f, 0.0f);
GLfloat angleCube = 0.0f;
                                                               glVertex3f( 0.0f, 1.0f, 0.0f);
                                                               glColor3f(0.0f, 1.0f, 0.0f);
void initGL() {
                                                               glVertex3f(-1.0f, -1.0f, 1.0f);
  glClearColor(0.0f, 0.0f, 0.0f, 1.0f);
                                                               glColor3f(0.0f, 0.0f, 1.0f);
  glClearDepth(1.0f);
                                                               glVertex3f(1.0f, -1.0f, 1.0f);
  glEnable(GL_DEPTH_TEST);
  glDepthFunc(GL_LEQUAL);
                                                               glColor3f(1.0f, 0.0f, 0.0f);
                                                               glVertex3f(0.0f, 1.0f, 0.0f);
  glShadeModel(GL SMOOTH):
  glHint(GL_PERSPECTIVE_CORRECTION_HINT, GL_NICEST);
                                                               glColor3f(0.0f, 0.0f, 1.0f);
                                                               glVertex3f(1.0f, -1.0f, 1.0f);
                                                               glColor3f(0.0f, 1.0f, 0.0f);
void display() {
                                                               glVertex3f(1.0f, -1.0f, -1.0f);
  glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
                                                               glColor3f(1.0f, 0.0f, 0.0f);
   glMatrixMode(GL_MODELVIEW);
                                                               glVertex3f(0.0f, 1.0f, 0.0f);
   glLoadIdentity();
                                                               glColor3f(0.0f, 1.0f, 0.0f);
  glTranslatef(1.5f, 0.0f, -7.0f);
                                                               glVertex3f(1.0f, -1.0f, -1.0f);
  glRotatef(angleCube, 1.0f, 1.0f, 1.0f);
                                                               glColor3f(0.0f, 0.0f, 1.0f);
                                                               glVertex3f(-1.0f, -1.0f, -1.0f);
   glBegin(GL_QUADS);
     glColor3f(0.0f, 1.0f, 0.0f);
                                                               {\tt glColor3f(1.0f,0.0f,0.0f);}
      glVertex3f( 1.0f, 1.0f, -1.0f);
                                                               glVertex3f( 0.0f, 1.0f, 0.0f);
      glVertex3f(-1.0f, 1.0f, -1.0f);
                                                               glColor3f(0.0f,0.0f,1.0f);
     {\tt glVertex3f(-1.0f, 1.0f, 1.0f);}
                                                               glVertex3f(-1.0f,-1.0f,-1.0f);
                                                               glColor3f(0.0f,1.0f,0.0f);
     glVertex3f( 1.0f, 1.0f, 1.0f);
                                                               glVertex3f(-1.0f,-1.0f, 1.0f);
      glColor3f(1.0f, 0.5f, 0.0f);
                                                            glEnd();
      glVertex3f( 1.0f, -1.0f, 1.0f);
                                                            glutSwapBuffers();
     glVertex3f(-1.0f, -1.0f, 1.0f);
                                                            anglePyramid += 0.2f;
     glVertex3f(-1.0f, -1.0f, -1.0f);
                                                            angleCube -= 0.15f;
     glVertex3f( 1.0f, -1.0f, -1.0f);
      glColor3f(1.0f, 0.0f, 0.0f);
                                                         void timer(int value) {
     glVertex3f( 1.0f, 1.0f, 1.0f);
glVertex3f(-1.0f, 1.0f, 1.0f);
                                                           glutPostRedisplay();
                                                            glutTimerFunc(15, timer, 0);
      glVertex3f(-1.0f, -1.0f, 1.0f);
      glVertex3f( 1.0f, -1.0f, 1.0f);
                                                         void reshape(GLsizei width, GLsizei height) {
      glColor3f(1.0f, 1.0f, 0.0f);
                                                            if (height == 0) height = 1;
      glVertex3f( 1.0f, -1.0f, -1.0f);
                                                            GLfloat aspect = (GLfloat)width / (GLfloat)height;
                                                            glViewport(0, 0, width, height);
      glVertex3f(-1.0f, -1.0f, -1.0f);
     glVertex3f(-1.0f, 1.0f, -1.0f);
glVertex3f(1.0f, 1.0f, -1.0f);
                                                            glMatrixMode(GL_PROJECTION);
                                                            glLoadIdentity();
                                                            gluPerspective(45.0f, aspect, 0.1f, 100.0f);
      glColor3f(0.0f, 0.0f, 1.0f);
     glVertex3f(-1.0f, 1.0f, 1.0f);
glVertex3f(-1.0f, 1.0f, -1.0f);
                                                         int main(int argc, char** argv) {
     glVertex3f(-1.0f, -1.0f, -1.0f);
                                                           glutInit(&argc, argv);
                                                            glutInitDisplayMode(GLUT_DOUBLE | GLUT_DEPTH);
      glVertex3f(-1.0f, -1.0f, 1.0f);
                                                            glutInitWindowSize(640, 480);
      glColor3f(1.0f, 0.0f, 1.0f);
                                                            glutInitWindowPosition(50, 50);
      glVertex3f(1.0f, 1.0f, -1.0f);
                                                            glutCreateWindow("3d-animation");
     glVertex3f(1.0f, 1.0f, 1.0f);
                                                            glutDisplayFunc(display);
     glVertex3f(1.0f, -1.0f, 1.0f);
glVertex3f(1.0f, -1.0f, -1.0f);
                                                            glutReshapeFunc(reshape);
                                                            initGL();
   glEnd();
                                                            glutTimerFunc(0, timer, 0);
                                                            glutMainLoop();
   glLoadIdentity();
                                                            return 0;
   glTranslatef(-1.5f, 0.0f, -6.0f);
  glRotatef(anglePyramid, 1.0f, 1.0f, 0.0f);
```

ACTIVITY

- * Determine what glLoadIdentity, glTranslatef, glRotatef, glColor3f, glVertex3f, glBegin, and glEnd are for
- * Determine how to use glLoadIdentity, glTranslatef, glRotatef, glColor3f, and glVertex3f

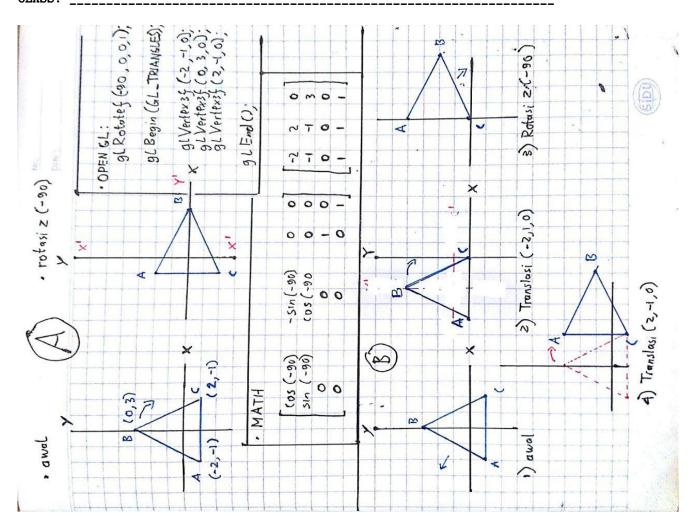
```
STUDENT REGISTRATION ID (NRP): ______
NAME: _____
CLASS:
// Compile: g++ 3d.cpp -o 3d -lglut -lGL -lGLU
#include "GL/glut.h"
                                                 glLoadIdentity();
                                                 glTranslatef(0, -4, -20);
GLfloat angle = 0;
                                                 // X Rotation
void initGL()
                                                 /* glRotatef(angle, 1, 0 , 0); */
 glClearColor(0.0f, 0.0f, 0.0f, 1.0f);
                                                 // Y Rotation
 glClearDepth(1.0f);
                                                 /* glRotatef(angle, 0, 1, 0); */
 glEnable(GL_DEPTH_TEST);
 glDepthFunc(GL_LEQUAL);
                                                 // Z Rotation
 glShadeModel(GL_SMOOTH);
                                                 /* glRotatef(angle, 0, 0, 1); */
 glHint(GL_PERSPECTIVE_CORRECTION_HINT, GL_NICEST);
                                                 // Y Rotation + Translation
                                                 glTranslatef(-3, 0, 0);
void timer(int value)
                                                 glRotatef(angle, 0, 1, 0);
 glutPostRedisplay();
                                                 glTranslatef(3, 0, 0);
 glutTimerFunc(15, timer, 0);
                                                 */
                                                 // purple square
void reshape(GLsizei width, GLsizei height)
                                                 glColor3f(1, 0, 1);
                                                 glBegin(GL_QUADS);
                                                   glVertex3f(-3, 3, 0);
 if (height == 0)
   height = 1;
                                                   glVertex3f(3, 3, 0);
 GLfloat aspect = (GLfloat)width / (GLfloat)height;
                                                   glVertex3f(3, -3, 0);
 glViewport(0, 0, width, height);
                                                   glVertex3f(-3, -3, 0);
 glMatrixMode(GL_PROJECTION);
                                                 glEnd();
 glLoadIdentity();
 gluPerspective(45.0f, aspect, 0.1f, 100.0f);
                                                 angle += 1;
                                                 glutSwapBuffers();
void display()
 glClear(GL_COLOR_BUFFER_BIT |
                                               int main(int argc, char **argv)
GL_DEPTH_BUFFER_BIT);
 glMatrixMode(GL_MODELVIEW);
                                                 glutInit(&argc, argv);
                                                 glutInitDisplayMode(GLUT_DOUBLE | GLUT_DEPTH);
 // draw the white big square
                                                 glutInitWindowSize(640, 480);
                                                 glutInitWindowPosition(50, 50);
 glLoadIdentity():
 glTranslatef(0, -4, -20);
                                                 glutCreateWindow("3d-animation");
 glColor3f(1, 1, 1);
                                                 glutDisplayFunc(display);
 glBegin(GL_QUADS);
                                                 glutReshapeFunc(reshape);
   glVertex3f(0, 10, 0);
                                                 initGL();
   glVertex3f(10, 10, 0);
                                                 glutTimerFunc(0, timer, 0);
   glVertex3f(10, 0, 0);
                                                 glutMainLoop();
   glVertex3f(0, 0, 0);
                                                 return 0:
 glEnd();
```

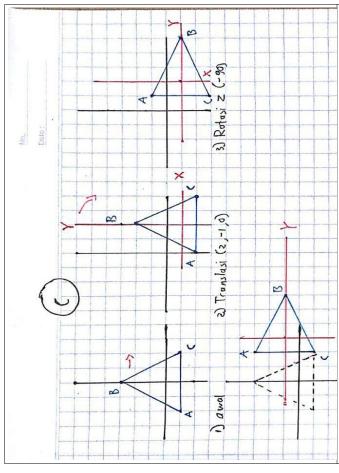
ACTIVITY

- * Determine the differences of `X Rotation`, `Y Rotation`, and `Z Rotation`
- * Find out how `Y Rotation + Translation` works.
- * Read the article in https://www.mathplanet.com/education/geometry/transformations to understand about transformation matrix.
- * Suppose the angle is 45° , determine the coordinate of the purple square (relative to bottom-left of the white square) after these operations:
 - * X Rotation
 - * Y Rotation
 - * Z Rotation
 - * Y Rotation + Translation

GRAPHICS PROGRAMMING: 3D Transformation (Composition: Math vs OpenGL)

STUDENT	REGISTRATION	ID	(NRP):	
NAME: _				
CT ACC.				





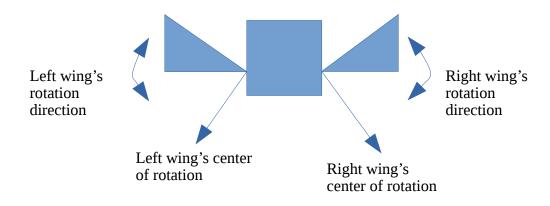
- 1. Try to understand figure A
- 2. Look at figure B. Calculate the Coordinate of ABC triangle by using matrix multiplication.
- 3. Look at figure C. Write an OpenGL program to do the rotation.

GRAPHICS PROGRAMMING: 3D Transformation (Flappy bird)

STUDENT	REGISTRATION	ID	(NRP):	 	
NAME: _				 	
CLASS:					

ACTIVITY

Create an OpenGL program to make a flappy bird like motion. The body of the bird is a square, while it's two wings are triangle



GRAPHICS PROGRAMMING: 3D Transformation (Drone)

STUDEN	T REGISTRATION	ID	(NRP):	
NAME:				
CLASS:				

ACTIVITY

Create an OpenGL program to make a drone like motion. The body of the drone is a square, while it's four copters are composed of two triangles

