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?**

GRAPHICS PROGRAMMING: 3D Transformation

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"

GLfloat anglePyramid = 0.0f;
GLfloat angleCube = 0.0f;

void initGL() {
   glClearColor(0.0f, 0.0f, 0.0f, 1.0f);
   glClearDepth(1.0f);
   glEnable(GL_DEPTH_TEST);
   glDepthFunc(GL_LEQUAL);
   glShadeModel(GL_SMOOTH);
   glHint(GL_PERSPECTIVE_CORRECTION_HINT, GL_NICEST);
}

void display() {
   glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
   glMatrixMode(GL_MODELVIEW);

   glLoadIdentity();
   glTranslatef(1.5f, 0.0f, -7.0f);
   glRotatef(angleCube, 1.0f, 1.0f, 1.0f);

   glBegin(GL_QUADS);
      glColor3f(0.0f, 1.0f, 0.0f);
      glVertex3f( 1.0f, 1.0f, -1.0f);
      glVertex3f(-1.0f, 1.0f, -1.0f);
      glVertex3f(-1.0f, 1.0f,  1.0f);
      glVertex3f( 1.0f, 1.0f,  1.0f);

      glColor3f(1.0f, 0.5f, 0.0f);
      glVertex3f( 1.0f, -1.0f,  1.0f);
      glVertex3f(-1.0f, -1.0f,  1.0f);
      glVertex3f(-1.0f, -1.0f, -1.0f);
      glVertex3f( 1.0f, -1.0f, -1.0f);

      glColor3f(1.0f, 0.0f, 0.0f);
      glVertex3f( 1.0f,  1.0f, 1.0f);
      glVertex3f(-1.0f,  1.0f, 1.0f);
      glVertex3f(-1.0f, -1.0f, 1.0f);
      glVertex3f( 1.0f, -1.0f, 1.0f);

      glColor3f(1.0f, 1.0f, 0.0f);
      glVertex3f( 1.0f, -1.0f, -1.0f);
      glVertex3f(-1.0f, -1.0f, -1.0f);
      glVertex3f(-1.0f,  1.0f, -1.0f);
      glVertex3f( 1.0f,  1.0f, -1.0f);

      glColor3f(0.0f, 0.0f, 1.0f);
      glVertex3f(-1.0f,  1.0f,  1.0f);
      glVertex3f(-1.0f,  1.0f, -1.0f);
      glVertex3f(-1.0f, -1.0f, -1.0f);
      glVertex3f(-1.0f, -1.0f,  1.0f);

      glColor3f(1.0f, 0.0f, 1.0f);
      glVertex3f(1.0f,  1.0f, -1.0f);
      glVertex3f(1.0f,  1.0f,  1.0f);
      glVertex3f(1.0f, -1.0f,  1.0f);
      glVertex3f(1.0f, -1.0f, -1.0f);
   glEnd();

   glLoadIdentity();
   glTranslatef(-1.5f, 0.0f, -6.0f);
   glRotatef(anglePyramid, 1.0f, 1.0f, 0.0f);

   glBegin(GL_TRIANGLES);
      glColor3f(1.0f, 0.0f, 0.0f);
      glVertex3f( 0.0f, 1.0f, 0.0f);
      glColor3f(0.0f, 1.0f, 0.0f);
      glVertex3f(-1.0f, -1.0f, 1.0f);
      glColor3f(0.0f, 0.0f, 1.0f);
      glVertex3f(1.0f, -1.0f, 1.0f);

      glColor3f(1.0f, 0.0f, 0.0f);
      glVertex3f(0.0f, 1.0f, 0.0f);
      glColor3f(0.0f, 0.0f, 1.0f);
      glVertex3f(1.0f, -1.0f, 1.0f);
      glColor3f(0.0f, 1.0f, 0.0f);
      glVertex3f(1.0f, -1.0f, -1.0f);

      glColor3f(1.0f, 0.0f, 0.0f);
      glVertex3f(0.0f, 1.0f, 0.0f);
      glColor3f(0.0f, 1.0f, 0.0f);
      glVertex3f(1.0f, -1.0f, -1.0f);
      glColor3f(0.0f, 0.0f, 1.0f);
      glVertex3f(-1.0f, -1.0f, -1.0f);

      glColor3f(1.0f,0.0f,0.0f);
      glVertex3f( 0.0f, 1.0f, 0.0f);
      glColor3f(0.0f,0.0f,1.0f);
      glVertex3f(-1.0f,-1.0f,-1.0f);
      glColor3f(0.0f,1.0f,0.0f);
      glVertex3f(-1.0f,-1.0f, 1.0f);
   glEnd();
   glutSwapBuffers();
   anglePyramid += 0.2f;
   angleCube -= 0.15f;
}

void timer(int value) {
   glutPostRedisplay();
   glutTimerFunc(15, timer, 0);
}

void reshape(GLsizei width, GLsizei height) {
   if (height == 0) height = 1;
   GLfloat aspect = (GLfloat)width / (GLfloat)height;
   glViewport(0, 0, width, height);
   glMatrixMode(GL_PROJECTION);
   glLoadIdentity();
   gluPerspective(45.0f, aspect, 0.1f, 100.0f);
}

int main(int argc, char** argv) {
   glutInit(&argc, argv);
   glutInitDisplayMode(GLUT_DOUBLE | GLUT_DEPTH);
   glutInitWindowSize(640, 480);
   glutInitWindowPosition(50, 50);
   glutCreateWindow("3d-animation");
   glutDisplayFunc(display);
   glutReshapeFunc(reshape);
   initGL();
   glutTimerFunc(0, timer, 0);
   glutMainLoop();
   return 0;
}
```

ACTIVITY

*** Determine what glLoadIdentity, glTranslatef, glRotatef, glColor3f, glVertex3f, glBegin, and glEnd are for**

*** Determine how to use glLoadIdentity, glTranslatef, glRotatef, glColor3f, and glVertex3f**

GRAPHICS PROGRAMMING: 3D Transformation

STUDENT REGISTRATION ID (NRP): ______________________________
NAME: ____
CLASS: ____

```
// Compile: g++ 3d.cpp -o 3d -lglut -lGL -lGLU

#include "GL/glut.h"

GLfloat angle = 0;

void initGL()
{
  glClearColor(0.0f, 0.0f, 0.0f, 1.0f);
  glClearDepth(1.0f);
  glEnable(GL_DEPTH_TEST);
  glDepthFunc(GL_LEQUAL);
  glShadeModel(GL_SMOOTH);
  glHint(GL_PERSPECTIVE_CORRECTION_HINT, GL_NICEST);
}

void timer(int value)
{
  glutPostRedisplay();
  glutTimerFunc(15, timer, 0);
}

void reshape(GLsizei width, GLsizei height)
{
  if (height == 0)
    height = 1;
  GLfloat aspect = (GLfloat)width / (GLfloat)height;
  glViewport(0, 0, width, height);
  glMatrixMode(GL_PROJECTION);
  glLoadIdentity();
  gluPerspective(45.0f, aspect, 0.1f, 100.0f);
}

void display()
{
  glClear(GL_COLOR_BUFFER_BIT |
GL_DEPTH_BUFFER_BIT);
  glMatrixMode(GL_MODELVIEW);

  // draw the white big square
  glLoadIdentity();
  glTranslatef(0, -4, -20);
  glColor3f(1, 1, 1);
  glBegin(GL_QUADS);
    glVertex3f(0, 10, 0);
    glVertex3f(10, 10, 0);
    glVertex3f(10, 0, 0);
    glVertex3f(0, 0, 0);
  glEnd();
  glLoadIdentity();
  glTranslatef(0, -4, -20);

  // X Rotation
  /* glRotatef(angle, 1, 0 , 0); */

  // Y Rotation
  /* glRotatef(angle, 0, 1, 0); */

  // Z Rotation
  /* glRotatef(angle, 0, 0, 1); */

  // Y Rotation + Translation
  /*
  glTranslatef(-3, 0, 0);
  glRotatef(angle, 0, 1, 0);
  glTranslatef(3, 0, 0);
  */

  // purple square
  glColor3f(1, 0, 1);
  glBegin(GL_QUADS);
    glVertex3f(-3, 3, 0);
    glVertex3f(3, 3, 0);
    glVertex3f(3, -3, 0);
    glVertex3f(-3, -3, 0);
  glEnd();

  angle += 1;

  glutSwapBuffers();
}

int main(int argc, char **argv)
{
  glutInit(&argc, argv);
  glutInitDisplayMode(GLUT_DOUBLE | GLUT_DEPTH);
  glutInitWindowSize(640, 480);
  glutInitWindowPosition(50, 50);
  glutCreateWindow("3d-animation");
  glutDisplayFunc(display);
  glutReshapeFunc(reshape);
  initGL();
  glutTimerFunc(0, timer, 0);
  glutMainLoop();
  return 0;
}
```

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 45^0, 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**