```c
#include <GL/glew.h>
#include <GL/glut.h>
#include <stdio.h>
#include <stdlib.h>

static char* vsSource = "#version 120 \n\
in vec4 aPosition; \n\
in vec4 aColor; \n\
out vec4 vColor; \n\
uniform float udist; \n\
void main(void) { \n\
  gl_Position.x = aPosition.x + udist; \n\
  gl_Position.yzw = aPosition.yzw; \n\
  vColor = aColor; \n\
}";

static char* fsSource = "#version 120 \n\
in vec4 vColor; \n\
void main(void) { \n\
  gl_FragColor = vColor; \n\
}";

GLuint vs = 0;
GLuint fs = 0;
GLuint prog = 0;

char buf[1024];
float dist = 0.0f;

GLfloat vertices[] = {
    -0.2, -0.2, -0.2, 1.0, // 0
    -0.2, -0.2, +0.2, 1.0, // 1
    -0.2, +0.2, -0.2, 1.0, // 2
    -0.2, +0.2, +0.2, 1.0, // 3
    +0.2, -0.2, -0.2, 1.0, // 4
    +0.2, -0.2, +0.2, 1.0, // 5
    +0.2, +0.2, -0.2, 1.0, // 6
    +0.2, +0.2, +0.2, 1.0, // 7
};


GLfloat colors[] = {
    1.0, 0.0, 0.0,
    0.0, 1.0, 0.0,
    0.0, 0.0, 1.0,
    1.0, 0.0, 0.0,
    0.0, 1.0, 0.0,
    0.0, 0.0, 1.0,
    1.0, 0.0, 0.0,
    0.0, 1.0, 0.0
};

GLushort indices[] = { // 36 points, 12 triangles
    0, 4, 6,
    6, 2, 0,
    4, 5, 7,
```

```
57      7, 6, 4,
58      1, 3, 7,
59      7, 5, 1,
60      0, 2, 3,
61      3, 1, 0,
62      2, 6, 7,
63      7, 3, 2,
64      0, 1, 5,
65      5, 4, 0,
66  };
67
68  void myinit(void) {
69      GLuint status;
70
71      printf("***** Your student number and name *****\n");
72      vs = glCreateShader(GL_VERTEX_SHADER);
73      glShaderSource(vs, 1, &vsSource, NULL);
74      glCompileShader(vs);
75      glGetShaderiv(vs, GL_COMPILE_STATUS, &status);
76      printf("vs compile status = %s\n", (status == GL_TRUE) ? "true" :
          "false");
77      glGetShaderInfoLog(vs, sizeof(buf), NULL, buf);
78      printf("vs log = [%s]\n", buf);
79
80      fs = glCreateShader(GL_FRAGMENT_SHADER);
81      glShaderSource(fs, 1, &fsSource, NULL);
82      glCompileShader(fs);
83      glGetShaderiv(fs, GL_COMPILE_STATUS, &status);
84      printf("fs compile status = %s\n", (status == GL_TRUE) ? "true" :
          "false");
85      glGetShaderInfoLog(fs, sizeof(buf), NULL, buf);
86      printf("fs log = [%s]\n", buf);
87
88      prog = glCreateProgram();
89      glAttachShader(prog, vs);
90      glAttachShader(prog, fs);
91      glLinkProgram(prog);
92      glGetProgramiv(prog, GL_LINK_STATUS, &status);
93      printf("program link status = %s\n", (status == GL_TRUE) ? "true" :
          "false");
94      glGetProgramInfoLog(prog, sizeof(buf), NULL, buf);
95      printf("link log = [%s]\n", buf);
96      glValidateProgram(prog);
97      glGetProgramiv(prog, GL_VALIDATE_STATUS, &status);
98      printf("program validate status = %s\n", (status == GL_TRUE) ? "true" :
          "false");
99      glGetProgramInfoLog(prog, sizeof(buf), NULL, buf);
100     printf("validate log = [%s]\n", buf);
101     glUseProgram(prog);
102
103     GLuint loc;
104     GLuint vbo[1];
105     // using vertex buffer object
106     glGenBuffers(1, vbo);
107     glBindBuffer(GL_ARRAY_BUFFER, vbo[0]);
108     glBufferData(GL_ARRAY_BUFFER, 2 * 8 * 4 * sizeof(GLfloat), NULL,
```

```
        GL_STATIC_DRAW);
    glBufferSubData(GL_ARRAY_BUFFER, 0, 8 * 4 * sizeof(GLfloat), vertices);
    glBufferSubData(GL_ARRAY_BUFFER, 8 * 4 * sizeof(GLfloat), 8 * 4 * sizeof
      (GLfloat),
        colors);

    loc = glGetAttribLocation(prog, "aPosition");
    glEnableVertexAttribArray(loc);
    glVertexAttribPointer(loc, 4, GL_FLOAT, GL_FALSE, 0, (GLvoid *)0);

    loc = glGetAttribLocation(prog, "aColor");
    glEnableVertexAttribArray(loc);
    glVertexAttribPointer(loc, 4, GL_FLOAT, GL_FALSE, 0, (GLvoid *)(3 * 4 *
      sizeof(GLfloat)));

}

void mykeyboard(unsigned char key, int x, int y) {
    switch (key) {
    case 27: // ESCAPE
        exit(0);
        break;
    }
}



void myidle(void) {
    dist += 0.0001f;
    if (dist > 1.5)
        dist = 0.0f;

    // redisplay
    glutPostRedisplay();
}


void mydisplay(void) {
    GLuint loc;

    glClearColor(0.7f, 0.7f, 0.7f, 1.0f); // gray
    glClear(GL_COLOR_BUFFER_BIT);

    loc = glGetUniformLocation(prog, "udist");
    glUniform1f(loc, dist);

    glDrawElements(GL_TRIANGLES, 12 * 3, GL_UNSIGNED_SHORT, indices);

    glFlush();
    glutSwapBuffers();
}

int main(int argc, char* argv[]) {
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB);
    glutInitWindowSize(500, 500);
```

```
162     glutInitWindowPosition(0, 0);
163     glutCreateWindow("*** Your Student Number and Name ***");
164     glutDisplayFunc(mydisplay);
165     glutIdleFunc(myidle);
166     glutKeyboardFunc(mykeyboard);
167     glewInit();
168     myinit();
169     glutMainLoop();
170     return 0;
171 }
172
```