

4^η Εργαστηριακή Ενότητα

Άσκηση 10: Δημιουργήστε ένα αρχείο με όνομα `exer10.c` και χρησιμοποιείστε τον παρακάτω κώδικα.

α) Τι πρόκειται να εμφανίσει ο παρακάτω κώδικας;

β) Χρησιμοποιείστε τις συναρτήσεις `glPushMatrix` και `glPopMatrix` ώστε να μειώσετε το πλήθος των μετασχηματισμών.

```
#include <GL/freeglut.h>
#include <math.h>

void drawCube3d(float size) {
    float halfSize = size / 2.0f;

    GLfloat vertices[][3] = {
        // Front face
        {-halfSize, -halfSize, halfSize}, // 0
        {halfSize, -halfSize, halfSize},   // 1
        {halfSize, halfSize, halfSize},    // 2
        {-halfSize, halfSize, halfSize},   // 3

        // Back face
        {-halfSize, -halfSize, -halfSize}, // 4
        {halfSize, -halfSize, -halfSize},   // 5
        {halfSize, halfSize, -halfSize},    // 6
        {-halfSize, halfSize, -halfSize}    // 7
    };

    GLuint faces[][4] = {
        {0, 1, 2, 3}, // Front face
        {7, 6, 5, 4}, // Back face
        {0, 3, 7, 4}, // Left face
        {1, 5, 6, 2}, // Right face
        {3, 2, 6, 7}, // Top face
        {0, 4, 5, 1}  // Bottom face
    };

    // Define a single color for each face
    GLfloat colors[][3] = {
        {1.0f, 0.0f, 0.0f}, // Front face (red)
        {0.0f, 1.0f, 0.0f}, // Back face (green)
        {0.0f, 0.0f, 1.0f}, // Left face (blue)
        {1.0f, 1.0f, 0.0f}, // Right face (yellow)
        {1.0f, 0.0f, 1.0f}, // Top face (magenta)
        {0.0f, 1.0f, 1.0f}  // Bottom face (cyan)
    };

    // Draw the cube with colors
    for(int i = 0; i < 6; i++) {
        glBegin(GL_POLYGON);
        glColor3fv(colors[i]);
        for(int j = 0; j < 4; j++) {
            glVertex3fv(vertices[faces[i][j]]);
        }
        glEnd();
    }
}
```

```

}

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

    // Draw a black vertical line
    glColor3f(0.0f,0.0f,0.0f);
    glLineWidth(2); // size of the line
    glBegin(GL_LINES);
        glVertex2i(-50,0);
        glVertex2i(50,0);
    glEnd();

    // Draw a black horizontal line
    glColor3f(0.0f,0.0f,0.0f);
    glLineWidth(2); // size of the line
    glBegin(GL_LINES);
        glVertex2i(0,-50);
        glVertex2i(0,50);
    glEnd();

    // Draw a string with black Roman fonts
    glColor3f(0.0f,0.0f,0.0f);
    glRasterPos2f(-50, 50);
    glutBitmapString(GLUT_BITMAP_TIMES_ROMAN_24, (void *)"Exercise 10");

    glMatrixMode(GL_MODELVIEW);

    glTranslatef(25,25,0);
    glScalef(1.2,1.2,1.2);
    glRotatef(45,1.0,1.0,0.0);
    drawCube3d(20);
    glRotatef(-45,1.0,1.0,0.0);
    glScalef(1/1.2,1/1.2,1/1.2);
    glTranslatef(-25,-25,0);

    glFlush();
}

int main(int argc, char** argv) {
    glutInit(&argc,argv);
    glutInitWindowPosition(100,100);
    glutInitWindowSize(800,800);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB | GLUT_DEPTH);
    glutCreateWindow("A sample 3D OpenGL application");
    glMatrixMode(GL_PROJECTION);
    glOrtho(-5,105,-5,105, -100, 100);
    glTranslatef(50,50,0);
    glEnable(GL_DEPTH_TEST);
    glClearColor(1.0f,1.0f,1.0f,0);
    glutDisplayFunc(display);
    glutMainLoop();
    return 0;
}

```

Άσκηση 11: Δημιουργήστε ένα αρχείο με όνομα `exer11.c` και χρησιμοποιείτε τον παρακάτω κώδικα.

α) Τι πρόκειται να εμφανίσει ο παρακάτω κώδικας;

β) Βελτιώστε τον κώδικα ώστε με το πάτημα των πλήκτρων $\uparrow \downarrow$ ο χρήστης να μπορεί να μπορεί περιστρέψει το εικονιζόμενο σχήμα ως προς τον άξονα των X.

```
#include <GL/freeglut.h>
#include <math.h>

float rotate_y = 0;

void drawCube3d(float size) {
    float halfSize = size / 2.0f;

    GLfloat vertices[][3] = {
        // Front face
        {-halfSize, -halfSize, halfSize}, // 0
        {halfSize, -halfSize, halfSize}, // 1
        {halfSize, halfSize, halfSize}, // 2
        {-halfSize, halfSize, halfSize}, // 3

        // Back face
        {-halfSize, -halfSize, -halfSize}, // 4
        {halfSize, -halfSize, -halfSize}, // 5
        {halfSize, halfSize, -halfSize}, // 6
        {-halfSize, halfSize, -halfSize} // 7
    };

    GLuint faces[][4] = {
        {0, 1, 2, 3}, // Front face
        {7, 6, 5, 4}, // Back face
        {0, 3, 7, 4}, // Left face
        {1, 5, 6, 2}, // Right face
        {3, 2, 6, 7}, // Top face
        {0, 4, 5, 1} // Bottom face
    };

    // Define a single color for each face
    GLfloat colors[][3] = {
        {1.0f, 0.0f, 0.0f}, // Front face (red)
        {0.0f, 1.0f, 0.0f}, // Back face (green)
        {0.0f, 0.0f, 1.0f}, // Left face (blue)
        {1.0f, 1.0f, 0.0f}, // Right face (yellow)
        {1.0f, 0.0f, 1.0f}, // Top face (magenta)
        {0.0f, 1.0f, 1.0f} // Bottom face (cyan)
    };

    // Draw the cube with colors
    for(int i = 0; i < 6; i++) {
        glBegin(GL_POLYGON);
        glColor3fv(colors[i]);
        for(int j = 0; j < 4; j++) {
            glVertex3fv(vertices[faces[i][j]]);
        }
        glEnd();
    }
}

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

    // Draw a black vertical line
    glColor3f(0.0f, 0.0f, 0.0f);
    glLineWidth(2); // size of the line
    glBegin(GL_LINES);
        glVertex2i(-50, 0);
        glVertex2i(50, 0);
    glEnd();
}
```

```

// Draw a black horizontal line
glColor3f(0.0f,0.0f,0.0f);
glLineWidth(2); // size of the line
glBegin(GL_LINES);
    glVertex2i(0,-50);
    glVertex2i(0,50);
glEnd();

// Draw a string with black Roman fonts
glColor3f(0.0f,0.0f,0.0f);
glRasterPos2f(-50, 50);
glutBitmapString(GLUT_BITMAP_TIMES_ROMAN_24, (void *)"Exercise 11");

glMatrixMode(GL_MODELVIEW);

glPushMatrix();
glTranslatef(25,25,0);
glScalef(1.2,1.2,1.2);
glRotatef(45,1.0,1.0,0.0);
glRotatef(rotate_y,0.0,1.0,0.0);

drawCube3d(20);

glPopMatrix();

glFlush();
}

void specialInput(int key, int x, int y) {
    switch(key) {
        case GLUT_KEY_LEFT:
            rotate_y -= 0.5;
            break;
        case GLUT_KEY_RIGHT:
            rotate_y += 0.5;
            break;
    }

    glutPostRedisplay();
}

int main(int argc, char** argv) {
    glutInit(&argc,argv);
    glutInitWindowPosition(100,100);
    glutInitWindowSize(800,800);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB | GLUT_DEPTH);
    glutCreateWindow("A sample 3D OpenGL application");
    glMatrixMode(GL_PROJECTION);
    glOrtho(-5,105,-5,105, -100, 100);
    glTranslatef(50,50,0);
    glEnable(GL_DEPTH_TEST);
    glClearColor(1.0f,1.0f,1.0f,0);
    glutSpecialFunc(specialInput);
    glutDisplayFunc(display);
    glutMainLoop();
    return 0;
}

```

Άσκηση 12: Βελτιώστε τον κώδικα της Άσκησης 11 ώστε:

- α) Πατώντας το πλήκτρο ESC, ο χρήστης να μπορεί να τερματίσει την εκτέλεση του προγράμματος.
- β) Πατώντας τα πλήκτρα '+' και '-', ο χρήστης να μπορεί να αυξάνει και να μειώνει αντίστοιχα την ταχύτητα περιστροφής ως προς του άξονες X και Y.

Γενικότερη σημείωση για τις εργαστηριακές ασκήσεις

Σε μερικά συστήματα στο περιβάλλον WSL χρειάζεται να εκτελείτε την ακόλουθη εντολή σε κάθε νέο τερματικό παράθυρο που ανοίγετε. Σε αντίθετη περίπτωση μπορεί να εμφανιστεί Segmentation Fault χωρίς να υπάρχει κάποιο σφάλμα στην εφαρμογή σας.

```
export LIBGL_ALWAYS_SOFTWARE=1
```