

```

#include<iostream>
#include<math.h>
#include<GL/glut.h>
using namespace std;
typedef float Matrix4[4][4];
Matrix4 theMatrix;
static GLfloat input[8][3] =
{
    {40,40,-50},{90,40,-50},{90,90,-50},{40,90,-50},
    {30,30,0},{80,30,0},{80,80,0},{30,80,0}
};
float output[8][3];
float tx, ty, tz;
float sx, sy, sz;
float angle;
int choice, choiceRot;
void setIdentityM(Matrix4 m)
{
    for (int i = 0; i < 4; i++)
        for (int j = 0; j < 4; j++)
            m[i][j] = (i == j);
}
void translate(int tx, int ty, int tz)
{
    for (int i = 0; i < 8; i++)
    {
        output[i][0] = input[i][0] + tx;
        output[i][1] = input[i][1] + ty;
        output[i][2] = input[i][2] + tz;
    }
}
void scale(int sx, int sy, int sz)
{
    theMatrix[0][0] = sx;
    theMatrix[1][1] = sy;
    theMatrix[2][2] = sz;
}
void RotateX(float angle)
{
    angle = angle * 3.142 / 180;
    theMatrix[1][1] = cos(angle);
    theMatrix[1][2] = -sin(angle);
    theMatrix[2][1] = sin(angle);
    theMatrix[2][2] = cos(angle);
}
void RotateY(float angle)
{
    angle = angle * 3.14 / 180;
    theMatrix[0][0] = cos(angle);
    theMatrix[0][2] = -sin(angle);

```

```

        theMatrix[2][0] = sin(angle);
        theMatrix[2][2] = cos(angle);
    }
    void RotateZ(float angle)
    {
        angle = angle * 3.14 / 180;
        theMatrix[0][0] = cos(angle);
        theMatrix[0][1] = sin(angle);
        theMatrix[1][0] = -sin(angle);
        theMatrix[1][1] = cos(angle);
    }
    void multiplyM()
    {
        for (int i = 0; i < 8; i++)
        {
            for (int j = 0; j < 3; j++)
            {
                output[i][j] = 0;
                for (int k = 0; k < 3; k++)
                {
                    output[i][j] = output[i][j] + input[i][k] *
theMatrix[k][j];
                }
            }
        }
    }
    void Axes(void)
    {
        glColor3f(0.0, 0.0, 0.0);
        glBegin(GL_LINES);
        glVertex2s(-1000, 0);
        glVertex2s(1000, 0);
        glEnd();
        glBegin(GL_LINES);
        glVertex2s(0, -1000);
        glVertex2s(0, 1000);
        glEnd();
    }
    void draw(float a[8][3])
    {
        glBegin(GL_QUADS);
        glColor3f(1.0, 1.0, 0.0);
        glVertex3fv(a[0]);
        glVertex3fv(a[1]);
        glVertex3fv(a[2]);
        glVertex3fv(a[3]);
        glColor3f(0.0, 1.0, 1.0);
        glVertex3fv(a[0]);
        glVertex3fv(a[1]);

```

```

    glVertex3fv(a[5]);
    glVertex3fv(a[4]);
    glColor3f(1.0, 0.0, 1.0);
    glVertex3fv(a[0]);
    glVertex3fv(a[4]);
    glVertex3fv(a[7]);
    glVertex3fv(a[3]);
    glColor3f(1.5, 0.4, 1.6);
    glVertex3fv(a[1]);
    glVertex3fv(a[2]);
    glVertex3fv(a[6]);
    glVertex3fv(a[5]);
    glColor3f(1.7, 1.3, 0.7);
    glVertex3fv(a[2]);
    glVertex3fv(a[3]);
    glVertex3fv(a[7]);
    glVertex3fv(a[6]);
    glColor3f(0.9, 1.3, 0.5);
    glVertex3fv(a[4]);
    glVertex3fv(a[5]);
    glVertex3fv(a[6]);
    glVertex3fv(a[7]);
    glEnd();
}

void init()
{
    glClearColor(1.0, 1.0, 1.0, 1.0);
    glOrtho(-454.0, 454.0, -250.0, 250.0, -250.0, 250.0);
    glEnable(GL_DEPTH_TEST);
}

void display()
{
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    Axes();
    glColor3f(1.0, 0.0, 0.0);
    draw(input);
    setIdentityM(theMatrix);
    switch (choice)
    {
    case 1:
        translate(tx, ty, tz);
        break;
    case 2:
        scale(sx, sy, sz);
        multiplyM();
        break;
    case 3:
        switch (choiceRot) {
        case 1:

```

```

        RotateX(angle);
        break;
    case 2: RotateY(angle);
        break;
    case 3:
        RotateZ(angle);
        break;
    default:
        break;
    }
    multiplyM();
    break;
}
draw(output);
glFlush();
}
int main(int argc, char** argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB | GLUT_DEPTH);
    glutInitWindowSize(500, 500);
    glutInitWindowPosition(0, 0);
    glutCreateWindow("3D TRANSFORMATIONS");
    init();
    cout << "Enter your choice
number:\n1.Translation\n2.Scaling\n3.Rotation\nn=>";
    cin >> choice;
    switch (choice) {
    case 1:
        cout << "\nEnter Tx,Ty &Tz: \n";
        cin >> tx >> ty >> tz;
        break;
    case 2:
        cout << "\nEnter Sx,Sy & Sz: \n";
        cin >> sx >> sy >> sz;
        break;
    case 3:
        cout << "Enter your choice for Rotation about axis:\n1.About
X-axis."
        << "(y& z)\n2.About Y-axis.(x& z)\n3.About Z-axis."
        << "(x& y)\nn =>";
        cin >> choiceRot;
        switch (choiceRot) {
        case 1:
            cout << "\nEnter Rotation angle: ";
            cin >> angle;
            break;
        case 2:
            cout << "\nEnter Rotation angle: ";
            cin >> angle;

```

```
        break;
    case 3:
        cout << "\nEnter Rotation angle: ";
        cin >> angle;
        break;
    default:
        break;
    }
    break;
default:
    break;
}
glutDisplayFunc(display);
glutMainLoop();
return 0;
}
```