NATIONAL INSTITUTE OF TECHNOLOGY SILCHAR

Cachar, Assam

B.Tech. VIth Sem

Subject Code: CS-317

Subject Name: Graphics and Multimedia Lab

Submitted By:

Name   : Subhojit Ghimire

Sch. Id.   : 1912160

Branch   : CSE – B

1. **Implement a triangle and perform the following 2D transformation over the triangle:**
   a. **Translation**
   b. **Scaling**
   c. **Shearing**
   d. **Reflection and**
   e. **Rotation**

The program should be menu driven allowing the user to choose any of the transformation. Divide the coordinate axes; display the original object and the transformed object in different colour or different fil patterns.

➔ **CODE:**

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

GLfloat V1X = -100.0f, V1Y = -100.0f, V1Z = 0.0f;
GLfloat V2X = 100.0f, V2Y = -100.0f, V2Z = 0.0f;
GLfloat V3X = 0.0f, V3Y = 100.0f, V3Z = 0.0f;

/*
GLfloat V1X = 100.0f, V1Y = 100.0f, V1Z = 0.0f;
GLfloat V2X = 200.0f, V2Y = 100.0f, V2Z = 0.0f;
GLfloat V3X = 150.0f, V3Y = 200.0f, V3Z = 0.0f;
*/

void myInit () {
    glClear (GL_COLOR_BUFFER_BIT);
    glClearColor (0.0, 0.0, 0.0, 0.0);
    glColor3f (0.0f, 0.5f, 0.5f);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluOrtho2D (-500, 500, -500, 500);

    glBegin (GL_LINE_LOOP);
        glVertex3f (-400.0, 0.0, 0.0);
        glVertex3f (400.0, 0.0, 0.0);
    glEnd ();

    glBegin (GL_LINE_LOOP);
        glVertex3f (0.0, -400.0, 0.0);
        glVertex3f (0.0, 400.0, 0.0);
    glEnd ();

    glFlush ();
}
```

25th February, 2022

```
void drawTriangle () {
    glBegin (GL_TRIANGLES);
        glVertex3f (V1X, V1Y, V1Z);
        glVertex3f (V2X, V2Y, V2Z);
        glVertex3f (V3X, V3Y, V3Z);
    glEnd ();
    glFlush ();
}

void transformRotate (float deg, float xaxis, float yaxis, float zaxis)
{
    glColor3f (0.0f, 0.0f, 1.0f);
    glRotatef (deg, xaxis, yaxis, zaxis);
    drawTriangle ();
}

void transformTranslate (float xaxis, float yaxis, float zaxis) {
    glColor3f (0.0f, 1.0f, 0.0f);
    glTranslatef (xaxis, yaxis, zaxis);
    drawTriangle ();
}

void transformScale (float xaxis, float yaxis, float zaxis) {
    glColor3f (1.0f, 0.0f, 0.0f);
    glScalef (xaxis, yaxis, zaxis);
    drawTriangle ();
}

void transformShear (float xaxis, float yaxis, float zaxis) {
    glColor3f (0.7f, 0.5f, 0.0f);
    if (xaxis != 0.0) {
        V1X += V1Y * xaxis; V2X += V2Y * xaxis; V3X += V3Y * xaxis;
    }
    if (yaxis != 0.0) {
        V1Y += V1X * yaxis; V2Y += V2X * yaxis; V3Y += V3X * yaxis;
    }
    drawTriangle ();
}

void transformReflect (float xaxis, float yaxis, float zaxis) {
    glColor3f (0.0f, 0.5f, 0.2f);
    if (xaxis == 1.0) {
        V1Y = -V1Y; V2Y = -V2Y; V3Y = -V3Y;
    }
    if (yaxis == 1.0) {
        V1X = -V1X; V2X = -V2X; V3X = -V3X;
    }
    drawTriangle ();
```

```c
}

void myDisplay () {
    glColor3f (1.0, 1.0, 1.0);
    drawTriangle ();
    int opt;
    printf ("####################\n"
            "####### MENU #######\n"
            "####################\n"
            "1. TRANSLATION\n"
            "2. SCALING\n"
            "3. SHEARING\n"
            "4. REFLECTION\n"
            "5. ROTATION\n"
            "####################\n"
            "> ");
    scanf (" %d", &opt);
    switch (opt) {
        case 1:
            transformTranslate (200.0, 200.0, 0.0);
            break;
        case 2:
            transformScale (0.5, 0.5, 0.0);
            break;
        case 3:
            transformShear (1.2, 0.0, 0.0);
            break;
        case 4:
            transformReflect (1.0, 0.0, 0.0);
            break;
        case 5:
            transformRotate (-45.0, 0.0, 0.0, 1.0);
            break;
    }
}

int main (int argc, char **argv) {
    glutInit (&argc, argv);
    glutInitDisplayMode (GLUT_SINGLE | GLUT_RGB);
    glutInitWindowPosition (500, 0);
    glutInitWindowSize (1000, 1000);
    glutCreateWindow ("2D TRANSFORMATION OVER THE TRIANGLE");
    glutDisplayFunc (myDisplay);
    myInit ();
    glutMainLoop ();
    return 0;
}
```

OUTPUT:

// TRANSFORMATION – TRANSLATION

Fig: Before (White) and After (Green) Translation (x: 200; y: 200)

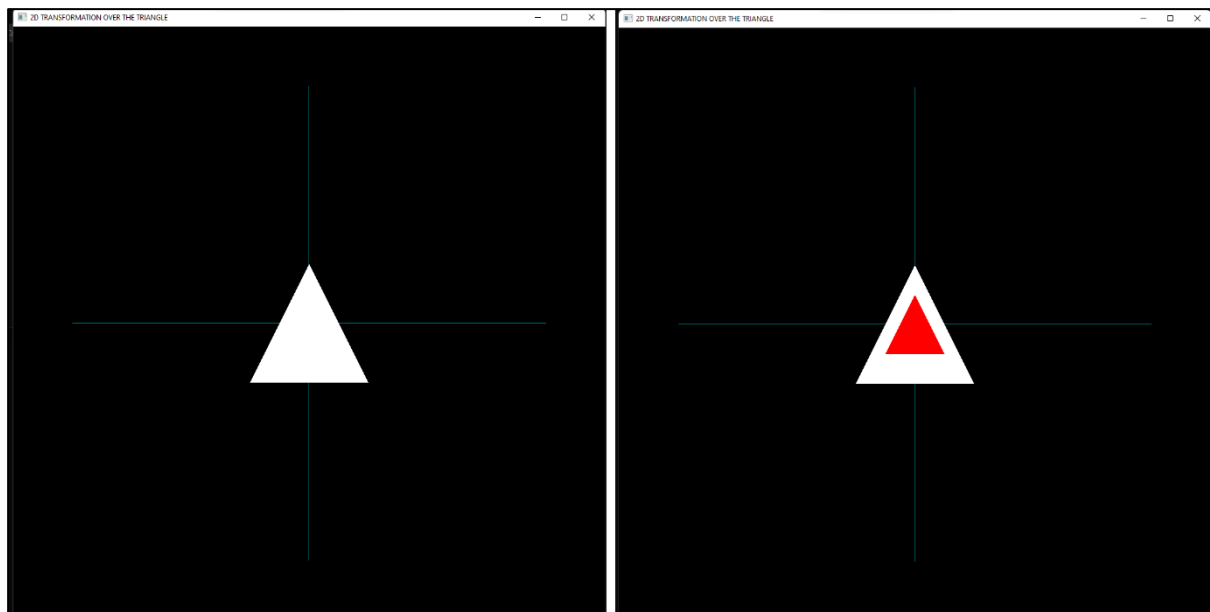## // TRANSFORMATION – SCALING

Fig: Before (White) and After (Red) Scaling (x: 0.5; y:0.5)
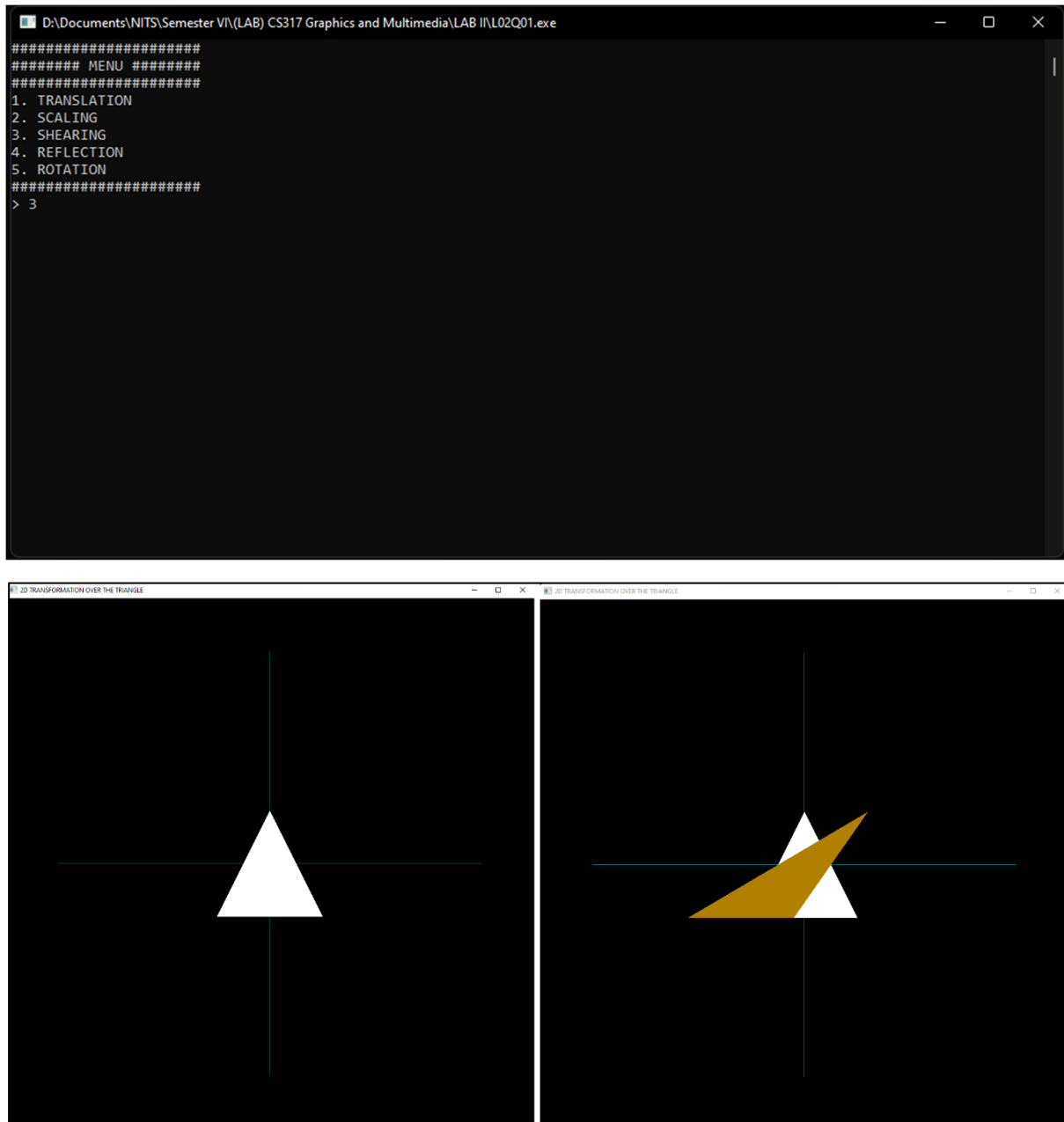
// TRANSFORMATION – SHEARING

Fig.: Before (White) and After (Golden) Shearing (x: 1.2)

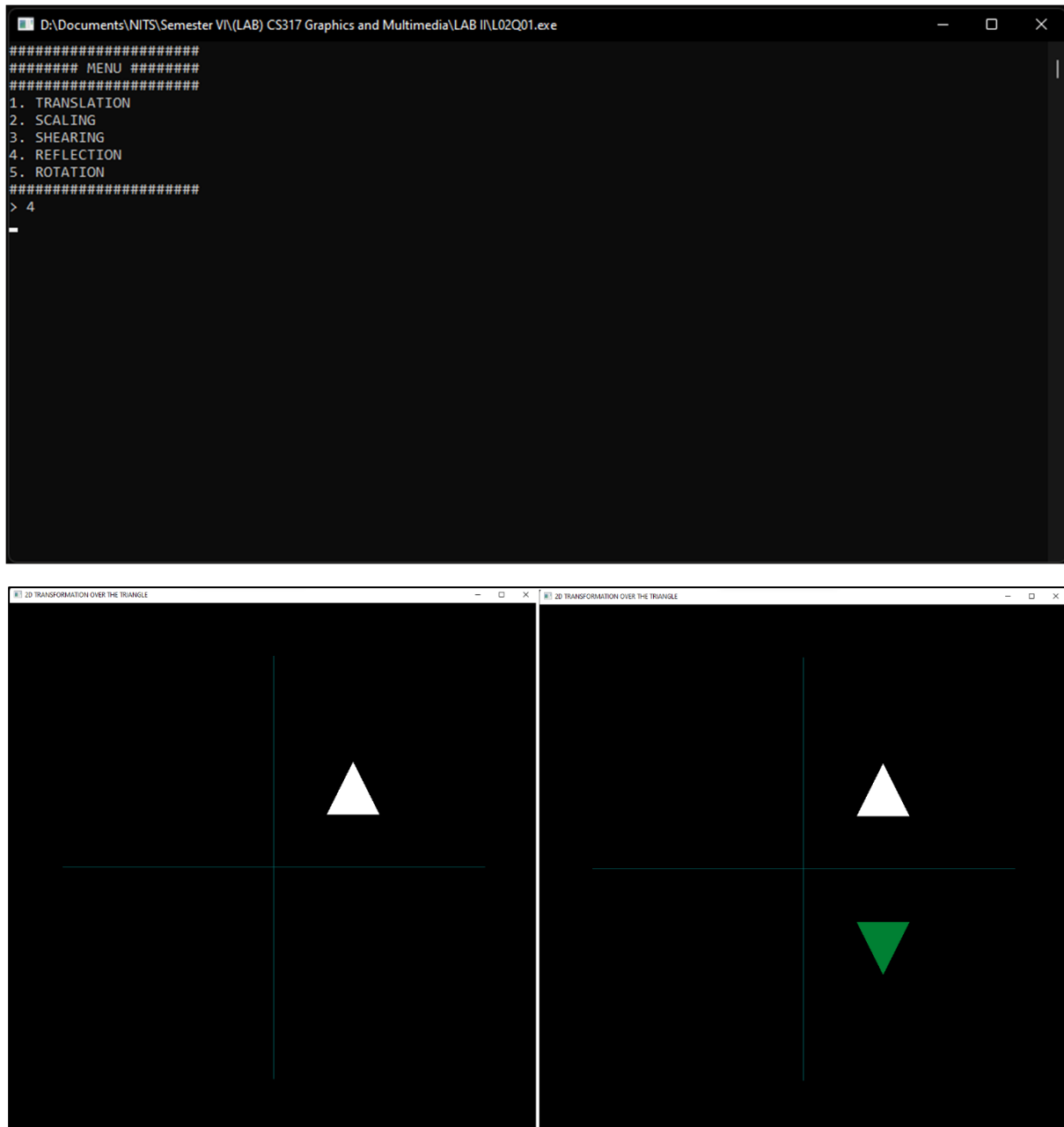## // TRANSFORMATION – REFLECTION

Fig.: Before (White) and After (Green) Reflection (x-axis)
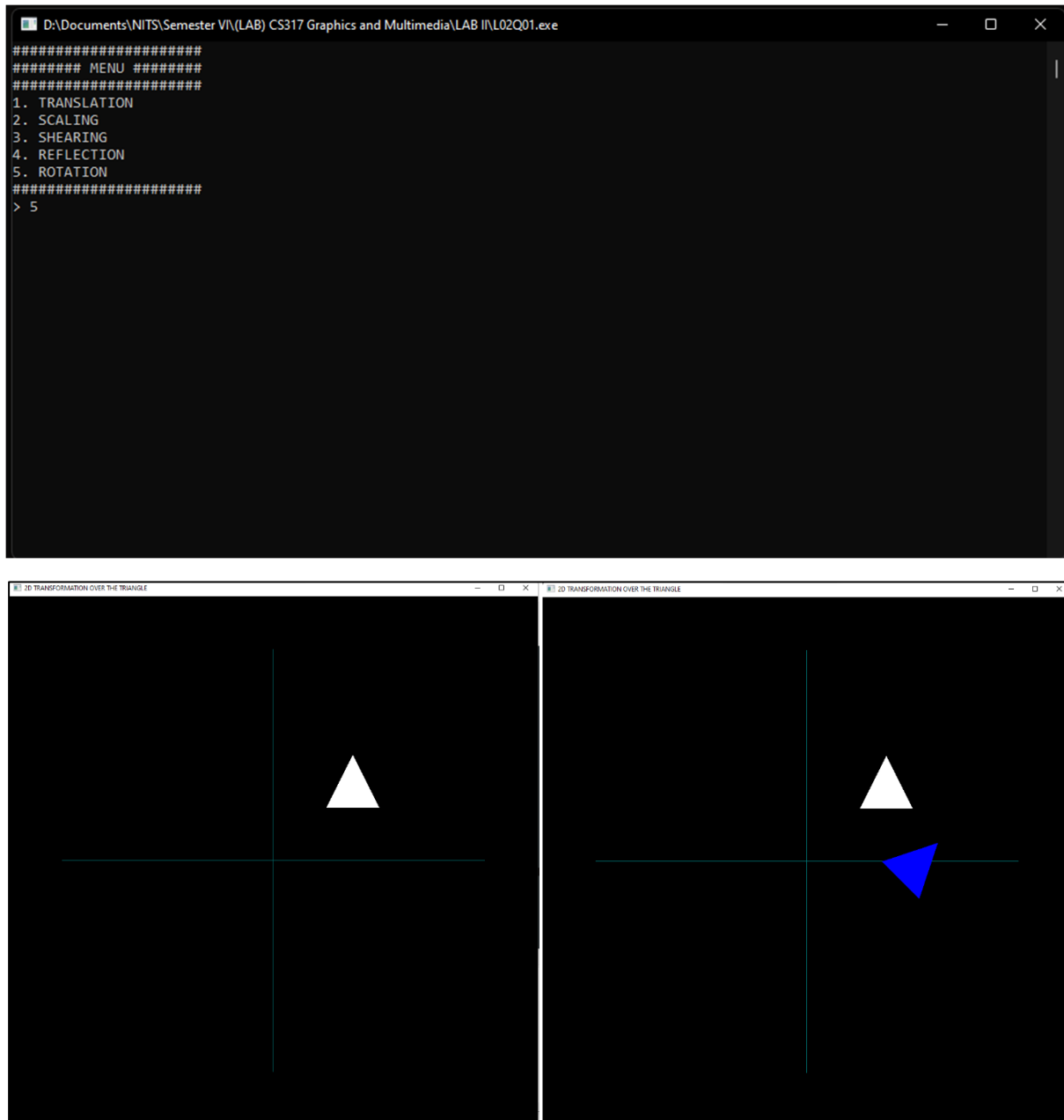
// TRANSFORMATION – ROTATION

Fig.: Before (White) and After (Blue) Rotation (45 degrees clockwise)

2. **Given a square object with coordinate points A (0, 3), B (3, 3), C (3, 0), D (0, 0) and perform**
    a. **Scaling with parameter 2 towards X axis and 3 towards Y axis**
    b. **Translation with distance 1 towards X axis and 5 towards Y axis**
    c. **Reflection on the X axis**
    d. **Shear parameter 3 on X axis and 4 on Y axis**
    e. **Rotation 90 degrees anticlockwise**

➔ **CODE:**

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

GLfloat V1X = 0.0f, V1Y = 3.0f, V1Z = 0.0f;
GLfloat V2X = 3.0f, V2Y = 3.0f, V2Z = 0.0f;
GLfloat V3X = 3.0f, V3Y = 0.0f, V3Z = 0.0f;
GLfloat V4X = 0.0f, V4Y = 0.0f, V4Z = 0.0f;

void myInit () {
    glClear (GL_COLOR_BUFFER_BIT);
    glClearColor (0.0, 0.0, 0.0, 0.0);
    glColor3f (0.0f, 0.5f, 0.5f);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluOrtho2D (-100, 100, -100, 100);

    glBegin (GL_LINE_LOOP);
        glVertex3f (-400.0, 0.0, 0.0);
        glVertex3f (400.0, 0.0, 0.0);
    glEnd ();

    glBegin (GL_LINE_LOOP);
        glVertex3f (0.0, -400.0, 0.0);
        glVertex3f (0.0, 400.0, 0.0);
    glEnd ();

    glFlush ();
}

void drawSquare () {
    glBegin (GL_POLYGON);
        glVertex3f (V1X, V1Y, V1Z);
        glVertex3f (V2X, V2Y, V2Z);
        glVertex3f (V3X, V3Y, V3Z);
        glVertex3f (V4X, V4Y, V4Z);
    glEnd ();
    glFlush ();
}
```

```
void transformRotate (float deg, float xaxis, float yaxis, float zaxis)
{
    glColor3f (0.0f, 0.0f, 1.0f);
    glRotatef (deg, xaxis, yaxis, zaxis);
    drawSquare ();
}

void transformTranslate (float xaxis, float yaxis, float zaxis) {
    glColor3f (0.0f, 1.0f, 0.0f);
    glTranslatef (xaxis, yaxis, zaxis);
    drawSquare ();
}

void transformScale (float xaxis, float yaxis, float zaxis) {
    glColor3f (1.0f, 0.0f, 0.0f);
    glScalef (xaxis, yaxis, zaxis);
    drawSquare ();
}

void transformShear (float xaxis, float yaxis, float zaxis) {
    glColor3f (0.7f, 0.5f, 0.0f);
    if (xaxis != 0.0) {
        V1X += V1Y * xaxis; V2X += V2Y * xaxis; V3X += V3Y * xaxis; V4X
+= V4Y * xaxis;
    }
    if (yaxis != 0.0) {
        V1Y += V1X * yaxis; V2Y += V2X * yaxis; V3Y += V3X * yaxis; V4Y
+= V4X * yaxis;
    }
    drawSquare ();
}

void transformReflect (float xaxis, float yaxis, float zaxis) {
    glColor3f (0.0f, 0.5f, 0.2f);
    if (xaxis == 1.0) {
        V1Y = -V1Y; V2Y = -V2Y; V3Y = -V3Y; V4Y = -V4Y;
    }
    if (yaxis == 1.0) {
        V1X = -V1X; V2X = -V2X; V3X = -V3X; V4X = -V4X;
    }
    drawSquare ();
}

void myDisplay () {
    glColor3f (1.0, 1.0, 1.0);
    drawSquare ();
    system ("pause");
```

```
        transformScale (2.0, 3.0, 0.0);
        // transformTranslate (1.0, 5.0, 0.0);
        // transformReflect (1.0, 0.0, 0.0);
        // transformShear (3.0, 4.0, 0.0);
        // transformRotate (90.0, 0.0, 0.0, 1.0);
    }

    int main (int argc, char **argv) {
        glutInit (&argc, argv);
        glutInitDisplayMode (GLUT_SINGLE | GLUT_RGB);
        glutInitWindowPosition (500, 0);
        glutInitWindowSize (1000, 1000);
        glutCreateWindow ("2D TRANSFORMATION OVER THE SQUARE");
        glutDisplayFunc (myDisplay);
        myInit ();
        glutMainLoop ();
        return 0;
    }
```

OUTPUT:

// TRANSFORMATION – SCALING
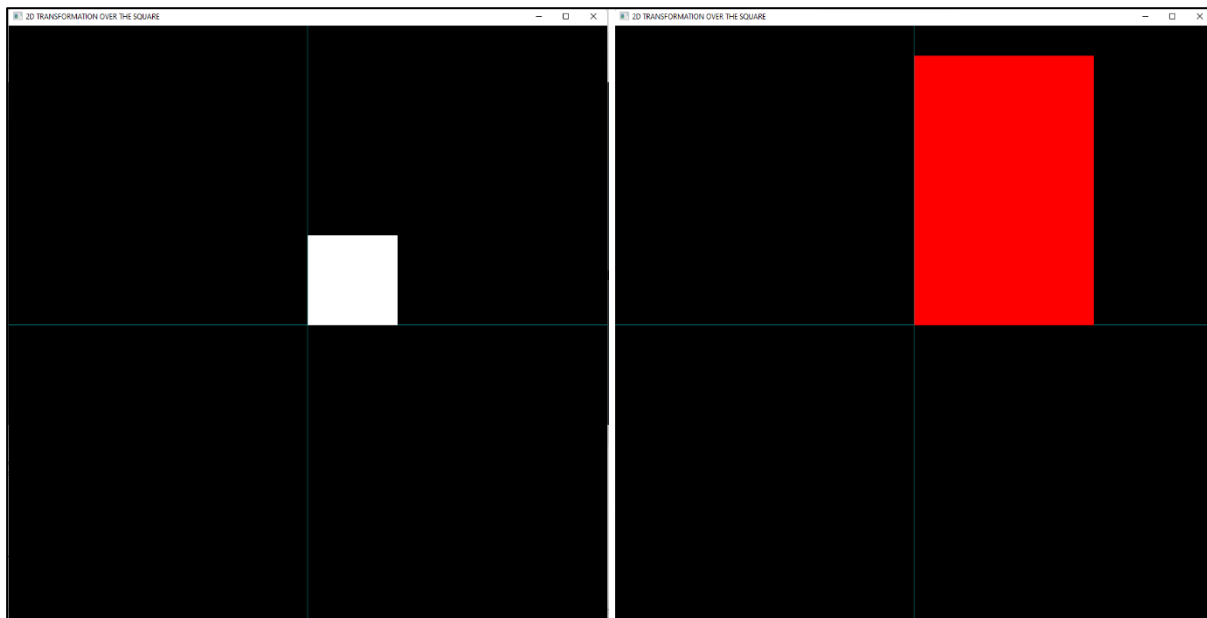
Fig.: Before (White) and After (Red) Scaling (x: 2.0; y: 3.0)

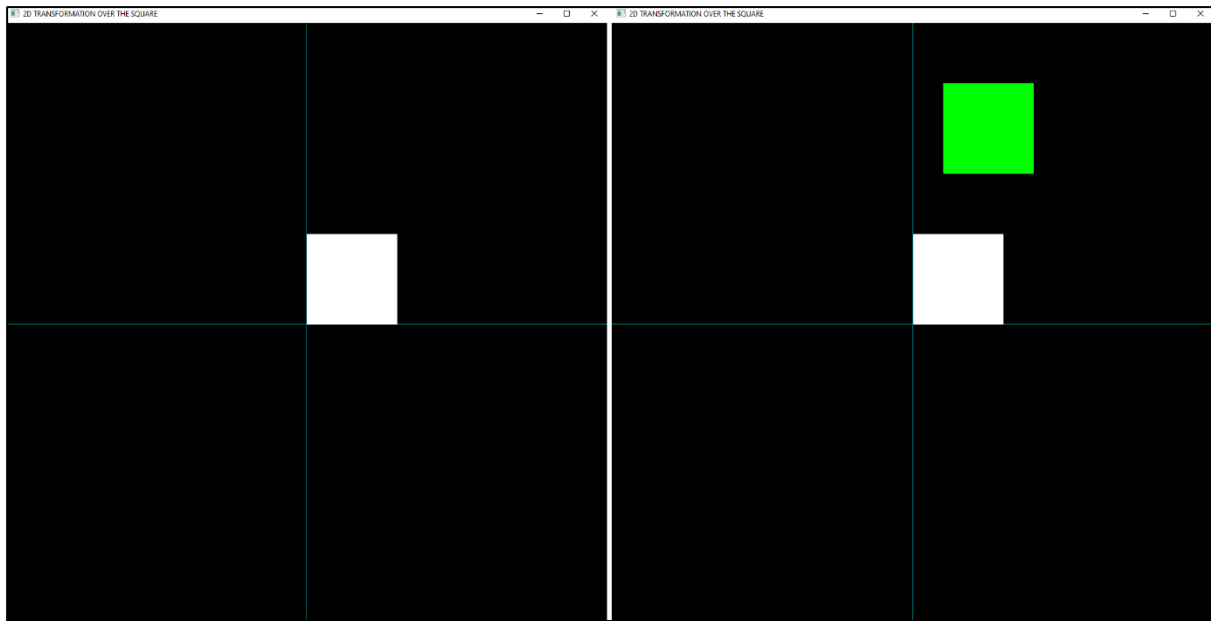**// TRANSFORMATION – TRANSLATION**

Fig.: Before (White) and After (Green) Translation (x: 1.0; y: 5.0)
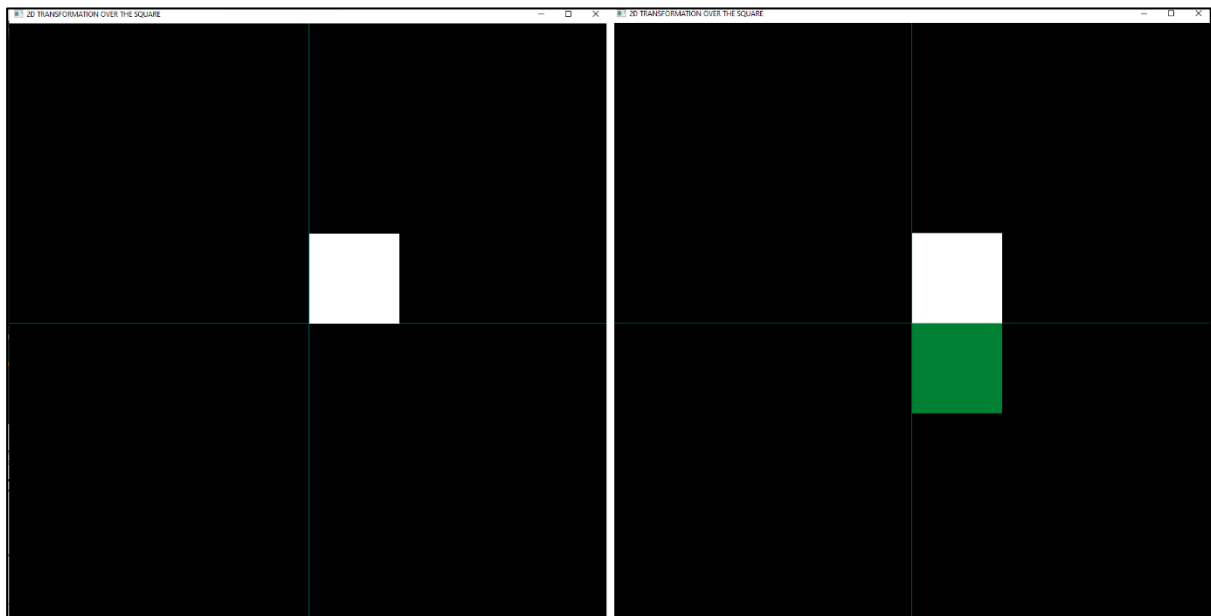
// TRANSFORMATION – REFLECTION

Fig.: Before (White) and After (Green) Reflection (x-axis)

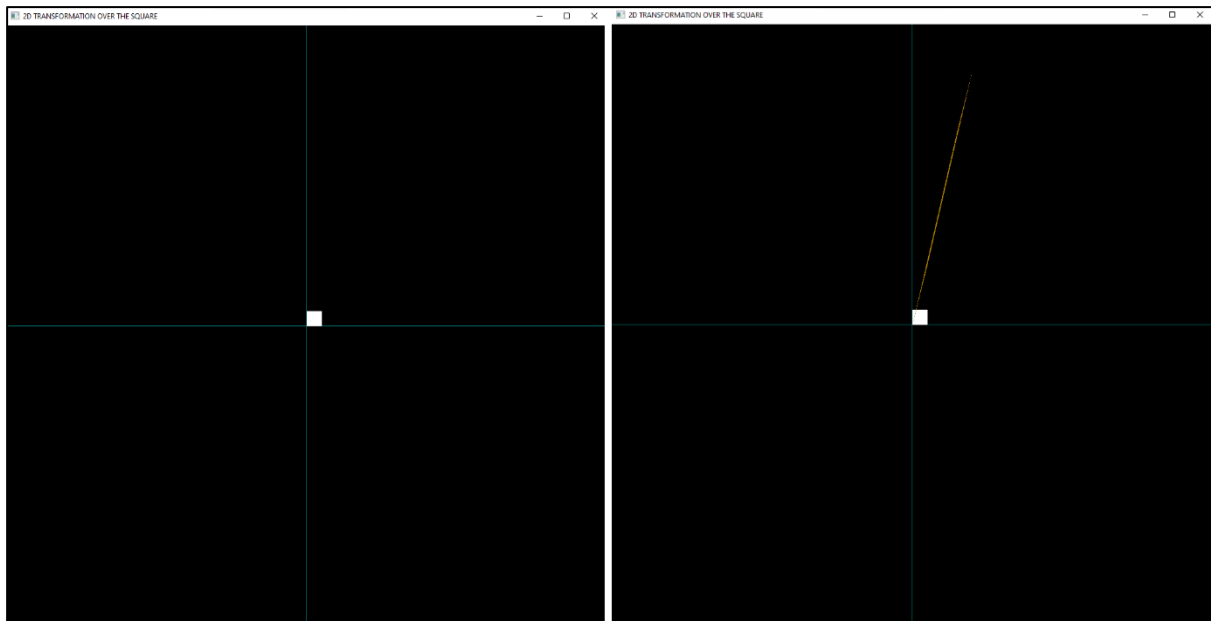## // TRANSFORMATION – SHEARING

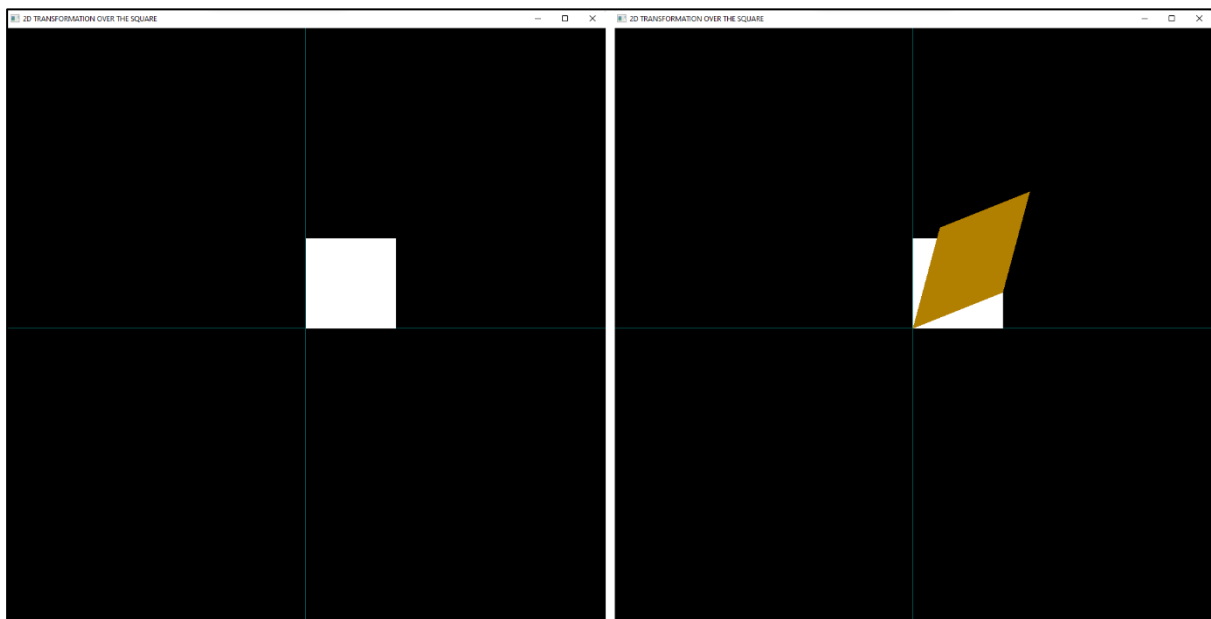Fig.: Before (White) and After (Golden Streak) Shearing (x: 3.0; y: 4.0)

Fig.: Before (White) and After (Golden) Shearing (x: 0.3; y: 0.4)
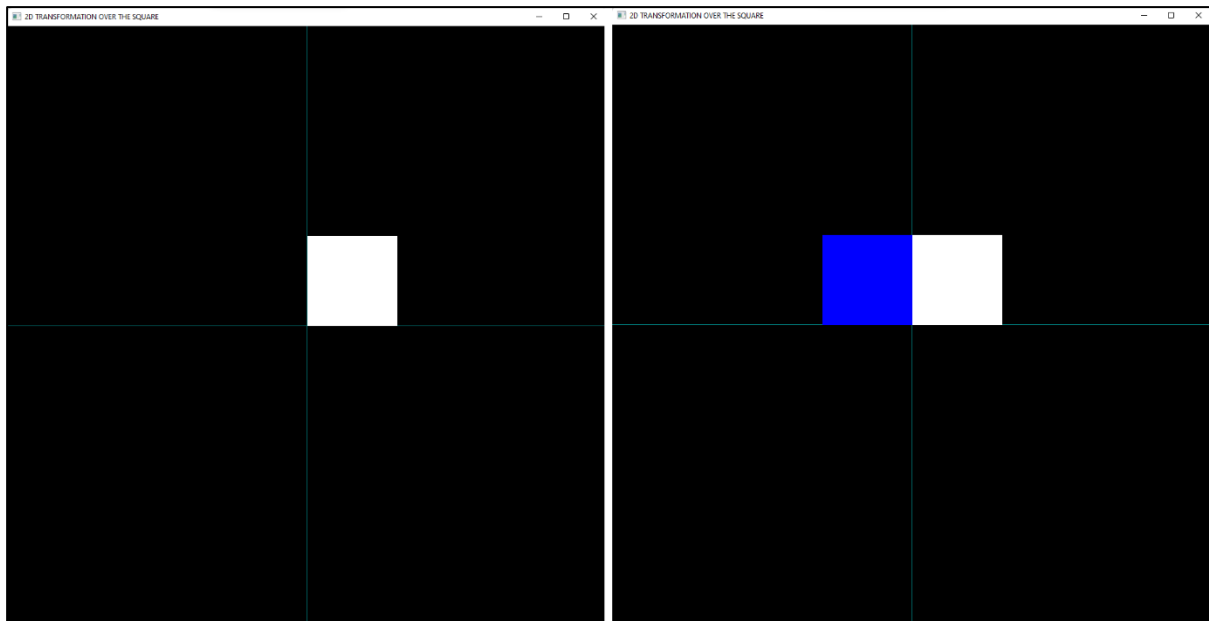
// TRANSFORMATION – ROTATION

Fig.: Before (White) and After (Blue) Rotation (90 degrees anti-clockwise)