UCS 1712 – GRAPHICS AND MULTIMEDIA LAB

ASSIGNMENT – 5

VISHAL N

185001198

CSEC

22.08.2021

1. 2-D TRANSFORMATIONS IN C++ USING OPENGL:

```
#include <GL/glut.h>
// #include <GL/freeglut.h>
#include <stdio.h>
#include <iostream>
#include <math.h>
#include <vector>
#include <cmath>

using namespace std;

vector<int> x_coordinates;
vector<int> y_coordinates;
int x_trans, y_trans, x_fixed, y_fixed, x_scale, y_scale, x_shear, y_shear;
double rotation_angle;
int polygon, edges, choice, x, y;

void myInit(void) {
    glClearColor(1.0, 1.0, 1.0, 0.0);
    glColor3f(0.0f, 0.0f, 0.0f);
    glPointSize(0.05);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluOrtho2D(-600.0, 600.0, -600.0, 600.0);
}

void drawAxis(void){
    glBegin(GL_LINES);
    glVertex2d(-600, 0);
    glVertex2d(600, 0);
    glEnd();
    glBegin(GL_LINES);
    glVertex2d(0, -600);
    glVertex2d(0, 600);
    glEnd();
}
```

```
double round(double d)
{
    return floor(d + 0.5);
}

void drawPolygon(void){
    if (polygon == 1){
        glBegin(GL_LINES);
    }

    else{
        glBegin(GL_POLYGON);
    }

    for (int i = 0; i < edges; i++)
    {
        glVertex2d(x_coordinates[i], y_coordinates[i]);
    }
    glEnd();
}

void drawTranslatedPolygon(int x, int y){
    if (polygon == 1)
    {
        glBegin(GL_LINES);
    }

    else{
        glBegin(GL_POLYGON);
    }

    for (int i = 0; i < edges; i++)
    {
        glVertex2d(x_coordinates[i] + x, y_coordinates[i] + y);
    }

    glEnd();
}

void drawRotatedOriginPolygon(double rotation_angle){
    if (polygon == 1)
    {
        glBegin(GL_LINES);
    }

    else{
        glBegin(GL_POLYGON);
    }
```

```
    for (int i = 0; i < edges; i++)
    {
        glVertex2d(round((x_coordinates[i] * cos(rotation_angle)) - (y_coordin
ates[i] * sin(rotation_angle))), round((x_coordinates[i] * sin(rotation_angle)
) + (y_coordinates[i] * cos(rotation_angle))));
    }

    glEnd();
}

void drawRotatedFixedPointPolygon(int x, int y, double rotation_angle){
    if (polygon == 1)
    {
        glBegin(GL_LINES);
    }

    else{
        glBegin(GL_POLYGON);
    }

    for (int i = 0; i < edges; i++)
    {
        x_coordinates[i] += x;
        y_coordinates[i] += y;
    }

    for (int i = 0; i < edges; i++)
    {
        glVertex2d(round((x_coordinates[i] * cos(rotation_angle)) - (y_coordin
ates[i] * sin(rotation_angle))), round((x_coordinates[i] * sin(rotation_angle)
) + (y_coordinates[i] * cos(rotation_angle))));
    }

    glEnd();
}

void drawScaledOriginPolygon(double x, double y){
    if (polygon == 1)
    {
        glBegin(GL_LINES);
    }

    else{
        glBegin(GL_POLYGON);
    }

    for (int i = 0; i < edges; i++)
```

```
    {
        glVertex2d(round(x_coordinates[i] * x), round(y_coordinates[i] * y));
    }

    glEnd();
}

void drawScaledFixedPointPolygon(double x, double y, int x_trans, int y_trans)
{
    if (polygon == 1)
    {
        glBegin(GL_LINES);
    }

    else
    {
        glBegin(GL_POLYGON);
    }

    // To Scale around a fixed point, first translate(-x, -
y), then scale(x, y), finally translate(x, y)
    for (int i = 0; i < edges; i++)
    {
        glVertex2d(x_coordinates[i] - x, y_coordinates[i] - y);
    }

    for (int i = 0; i < edges; i++)
    {
        glVertex2d(round(x_coordinates[i] * x), round(y_coordinates[i] * y));
    }

    for (int i = 0; i < edges; i++)
    {
        glVertex2d(x_coordinates[i] + x, y_coordinates[i] + y);
    }

    glEnd();
}

void drawReflectionXPolygon(){
    if (polygon == 1)
    {
        glBegin(GL_LINES);
    }

    else
    {
        glBegin(GL_POLYGON);
```

```
    }

    for (int i = 0; i < edges; i++)
    {
        glVertex2d(x_coordinates[i] , y_coordinates[i] * -1);
    }

    glEnd();
}

void drawReflectionYPolygon(){
    if (polygon == 1)
    {
        glBegin(GL_LINES);
    }

    else
    {
        glBegin(GL_POLYGON);
    }

    for (int i = 0; i < edges; i++)
    {
        glVertex2d(x_coordinates[i] * -1 , y_coordinates[i]);
    }

    glEnd();
}

void drawReflectionOriginPolygon(){
    if (polygon == 1)
    {
        glBegin(GL_LINES);
    }

    else
    {
        glBegin(GL_POLYGON);
    }

    for (int i = 0; i < edges; i++)
    {
        glVertex2d(x_coordinates[i] * -1 , y_coordinates[i] * -1);
    }

    glEnd();
}
```

```
void drawReflectionXequalsYPolygon(){
    if (polygon == 1)
    {
        glBegin(GL_LINES);
    }

    else
    {
        glBegin(GL_POLYGON);
    }

    for (int i = 0; i < edges; i++)
    {
        glVertex2d(y_coordinates[i], x_coordinates[i]);
    }

    glEnd();
}

void drawXShearedPolygon(int x_shear){
    if (polygon == 1)
    {
        glBegin(GL_LINES);
    }

    else
    {
        glBegin(GL_POLYGON);
    }

    for (int i = 0; i < edges; i++)
    {
        glVertex2d(x_coordinates[i] + (x_shear * y_coordinates[i]), y_coordina
tes[i]);
    }

    glEnd();
}

void drawYShearedPolygon(int y_shear){
    if (polygon == 1)
    {
        glBegin(GL_LINES);
    }

    else
    {
        glBegin(GL_POLYGON);
```

```
    }

    for (int i = 0; i < edges; i++)
    {
        glVertex2d(x_coordinates[i], y_coordinates[i] + (y_shear * x_coordinat
es[i]));
    }

    glEnd();
}

void myDisplay(void){
    glClear(GL_COLOR_BUFFER_BIT);
    glColor3f(0.0, 0.0, 0.0);
    drawAxis();
    drawPolygon();
        glClear(GL_COLOR_BUFFER_BIT);
        glColor3f(0.0, 0.0, 0.0);
        cout << "Enter your choice of transformation :\n";
        cout << "1. Translation" << endl;
        cout << "2. Rotation about Origin" << endl;
        cout << "3. Rotation wrt fixed point" << endl;
        cout << "4. Scaling wrt Origin" << endl;
        cout << "5. Scaling wrt fixed point" << endl;
        cout << "6. Reflection wrt x-axis" << endl;
        cout << "7. Reflection wrt y-axis" << endl;
        cout << "8. Reflection wrt origin" << endl;
        cout << "9. Reflection wrt line y=x" << endl;
        cout << "10. Shearing along x-direction" << endl;
        cout << "11. Shearing along y-direction" << endl;
        cout << "12. Exit" << endl;
        cout << "*****************************" << endl;
        cin >> choice;

        if (choice == 12)
        {
            glFlush();
        }

        if (choice == 1)
        {
            cout << "Enter the translation factor for X and Y: ";
            cin >> x_trans >> y_trans;
            drawAxis();
            drawPolygon();
            drawTranslatedPolygon(x_trans, y_trans);
        }
```

```
        if (choice == 2)
        {
            cout << "Enter the angle for rotation: ";
            cin >> rotation_angle;
            rotation_angle *= M_PI / 180;
            drawAxis();
            drawPolygon();
            drawRotatedOriginPolygon(rotation_angle);
        }

        if (choice == 3)
        {
            cout << "Enter the angle for rotation: ";
            cin >> rotation_angle;
            rotation_angle *= M_PI / 180;
            cout << "Enter the coordinates of the fixed point to rotate about:
 " << endl;
            cin >> x_fixed >> y_fixed;
            drawAxis();
            drawPolygon();
            drawRotatedFixedPointPolygon(x_fixed, y_fixed, rotation_angle);
        }

        if (choice == 4)
        {
            cout << "Enter the scaling factor for X and Y: ";
            cin >> x_scale >> y_scale;
            drawAxis();
            drawPolygon();
            drawScaledOriginPolygon(x_scale, y_scale);
        }

        if (choice == 5)
        {
            cout << "Enter the scaling factor for X and Y and coordinates of t
he fixed point to scale about: ";
            cin >> x_scale >> y_scale >> x_fixed >> y_fixed;
            drawAxis();
            drawPolygon();
            drawScaledFixedPointPolygon(x_scale, y_scale, x_fixed, y_fixed);
        }

        if (choice == 6)
        {
            drawAxis();
            drawPolygon();
            drawReflectionXPolygon();
        }
```

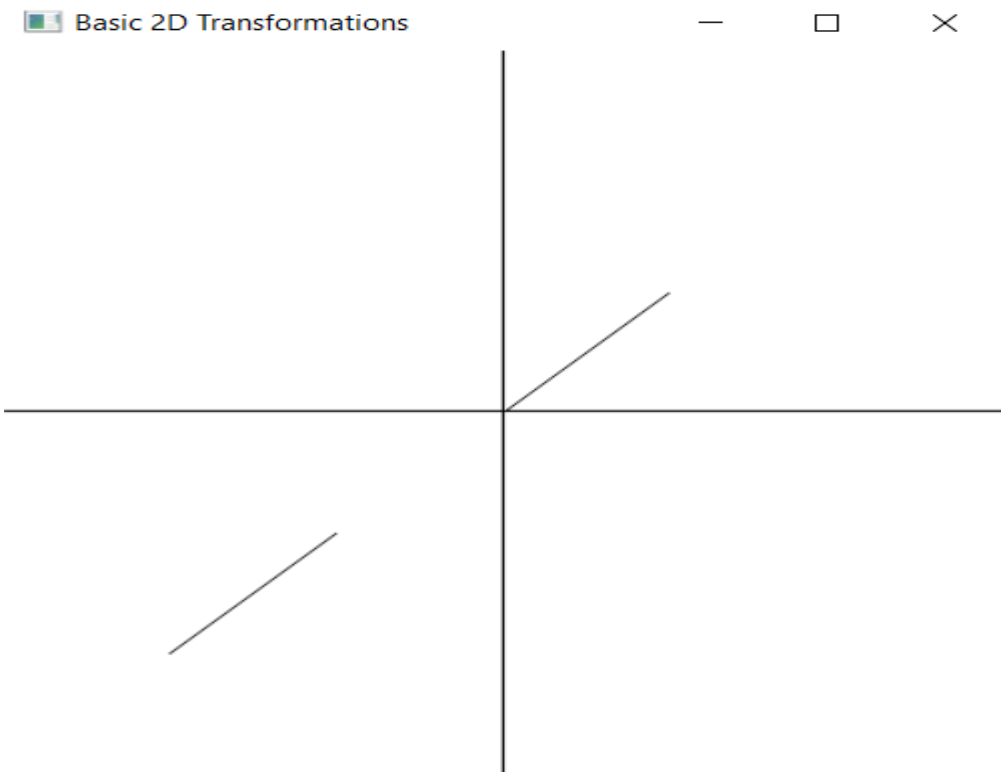
```
        if (choice == 7)
        {
            drawAxis();
            drawPolygon();
            drawReflectionYPolygon();
        }

        if (choice == 8)
        {
            drawAxis();
            drawPolygon();
            drawReflectionOriginPolygon();
        }

        if (choice == 9)
        {
            drawAxis();
            drawPolygon();
            drawReflectionXequalsYPolygon();
        }

        if (choice == 10)
        {
            drawAxis();
            cout << "Enter the x-shearing factor: " << endl;
            cin >> x_shear;
            drawPolygon();
            drawXShearedPolygon(x_shear);
        }

        if (choice == 11)
        {
            drawAxis();
            cout << "Enter the y-shearing factor: " << endl;
            cin >> y_shear;
            drawPolygon();
            drawYShearedPolygon(y_shear);
        }

    glFlush();
}

int main(int argc, char **argv)
{
    cout << "Enter the number of edges: ";
    cin >> edges;
    if (edges == 2)
```

```
    {
        polygon = 1;
    }

    else
    {
        polygon = -1;
    }

    cout << "Enter vertices: \n";
    for (int i = 0; i < edges; i++)
    {
        cout << "Enter co-ordinates for vertex " << i + 1 << " (X, Y): ";
        cin >> x >> y;
        x_coordinates.push_back(x);
        y_coordinates.push_back(y);
    }
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize(600, 600);
    glutInitWindowPosition(0, 0);
    glutCreateWindow("Basic 2D Transformations");
    myInit();
    glutDisplayFunc(myDisplay);
    glutMainLoop();
    cout << endl << endl << "Hello";
    return 1;
}
```

OUTPUTS:

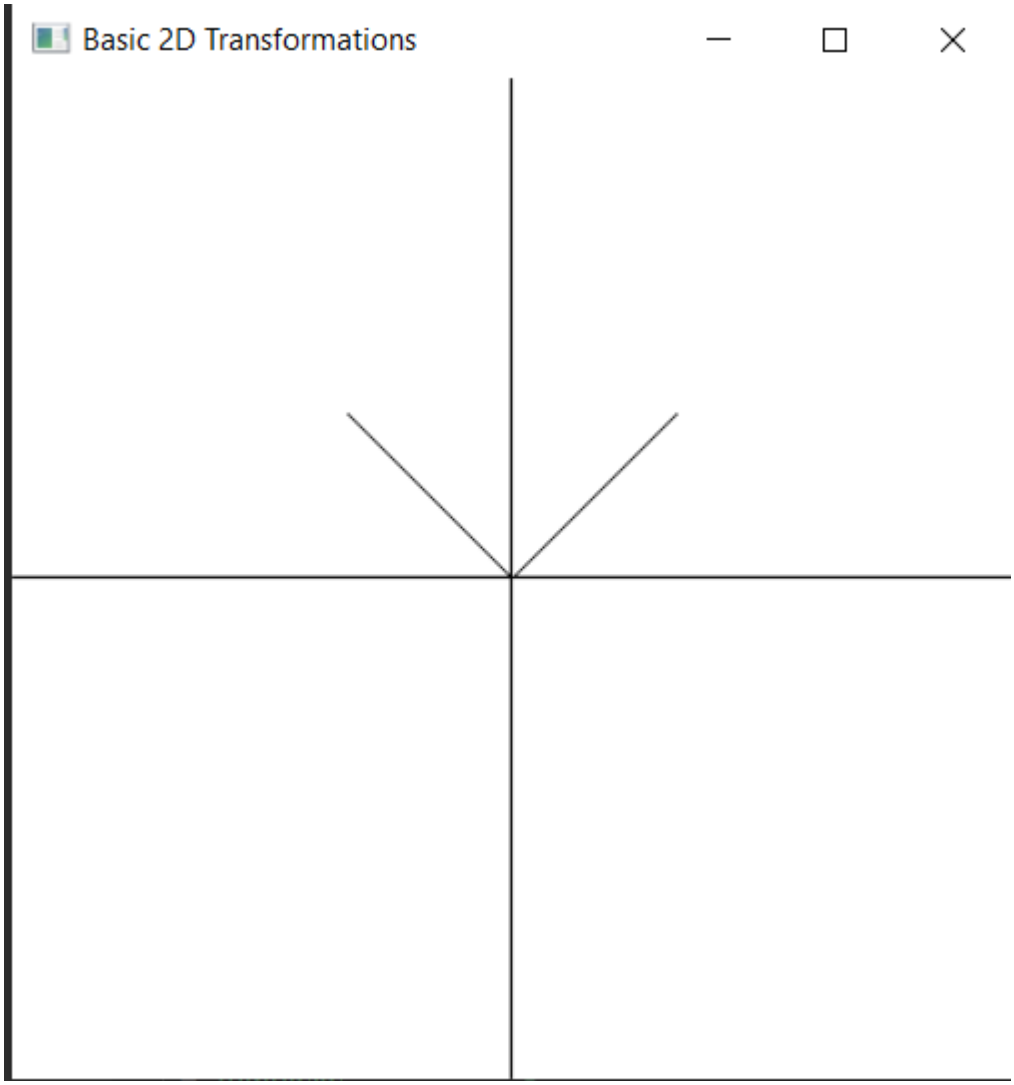
1.

```
PS C:\SSN 7th sem materials\Graphics-and-Multimedia-Lab\198-A5> g++ transformations.cpp -o transformations -lfreeglut -lopengl32 -lglew32 -lglu32
PS C:\SSN 7th sem materials\Graphics-and-Multimedia-Lab\198-A5> ./transformations
Enter the number of edges: 2
Enter vertices:
Enter co-ordinates for vertex 1 (X, Y): 0 0
Enter co-ordinates for vertex 2 (X, Y): 200 200
Enter your choice of transformation :
1. Translation
2. Rotation about Origin
3. Rotation wrt fixed point
4. Scaling wrt Origin
5. Scaling wrt fixed point
6. Reflection wrt x-axis
7. Reflection wrt y-axis
8. Reflection wrt origin
9. Reflection wrt line y=x
10. Shearing along x-direction
11. Shearing along y-direction
12. Exit
****************************
1
Enter the translation factor for X and Y: -400 -400
```



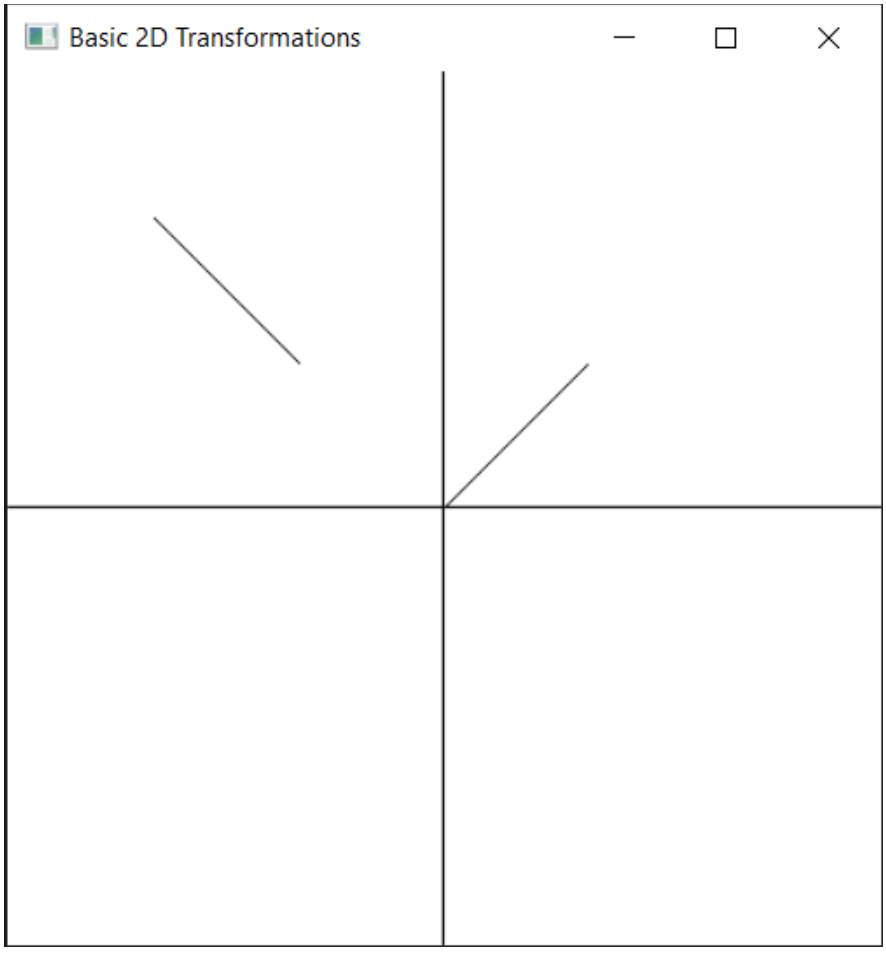
2.

```
PS C:\SSN 7th sem materials\Graphics-and-Multimedia-Lab\198-A5> ./transformations
Enter the number of edges: 2
Enter vertices:
Enter co-ordinates for vertex 1 (X, Y): 0 0
Enter co-ordinates for vertex 2 (X, Y): 200 200
Enter your choice of transformation :
1. Translation
2. Rotation about Origin
3. Rotation wrt fixed point
4. Scaling wrt Origin
5. Scaling wrt fixed point
6. Reflection wrt x-axis
7. Reflection wrt y-axis
8. Reflection wrt origin
9. Reflection wrt line y=x
10. Shearing along x-direction
11. Shearing along y-direction
12. Exit
*****************************
2
Enter the angle for rotation: 90
```


Basic 2D Transformations

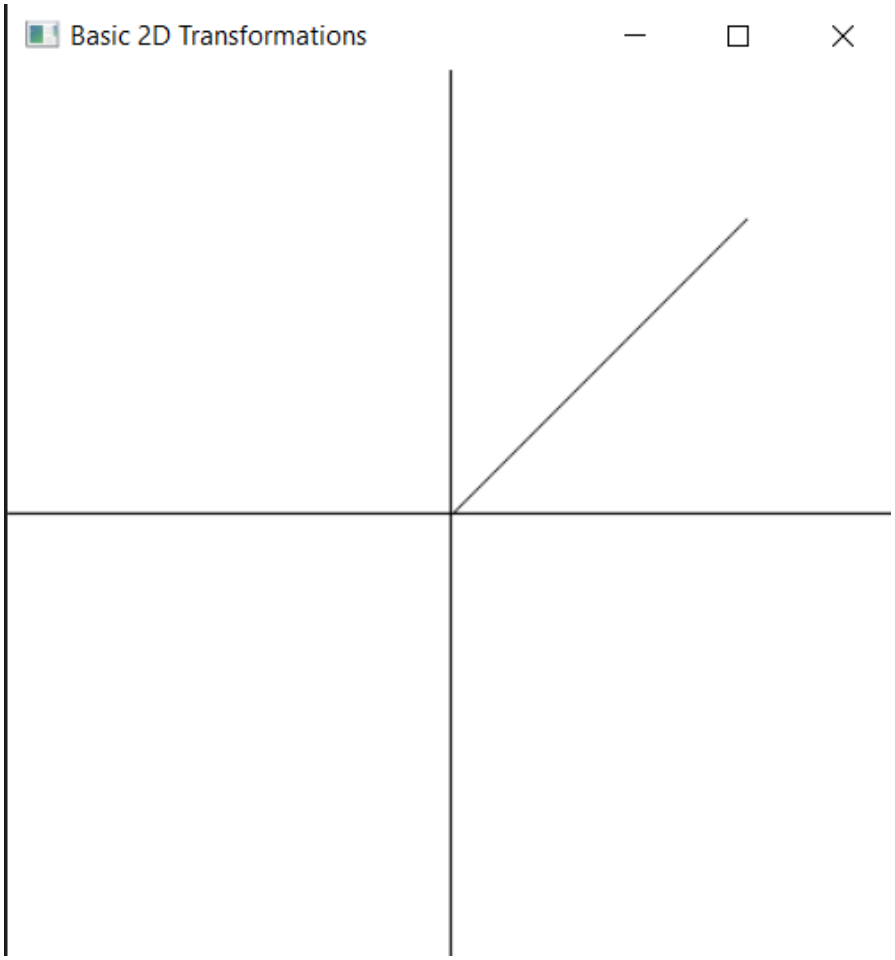
3.

```
PS C:\SSN 7th sem materials\Graphics-and-Multimedia-Lab\198-A5> ./transformations
Enter the number of edges: 2
Enter vertices:
Enter co-ordinates for vertex 1 (X, Y): 0 0
Enter co-ordinates for vertex 2 (X, Y): 200 200
Enter your choice of transformation :
1. Translation
2. Rotation about Origin
3. Rotation wrt fixed point
4. Scaling wrt Origin
5. Scaling wrt fixed point
6. Reflection wrt x-axis
7. Reflection wrt y-axis
8. Reflection wrt origin
9. Reflection wrt line y=x
10. Shearing along x-direction
11. Shearing along y-direction
12. Exit
*****************************
3
Enter the angle for rotation: 90
Enter the coordinates of the fixed point to rotate about:
200 200
```



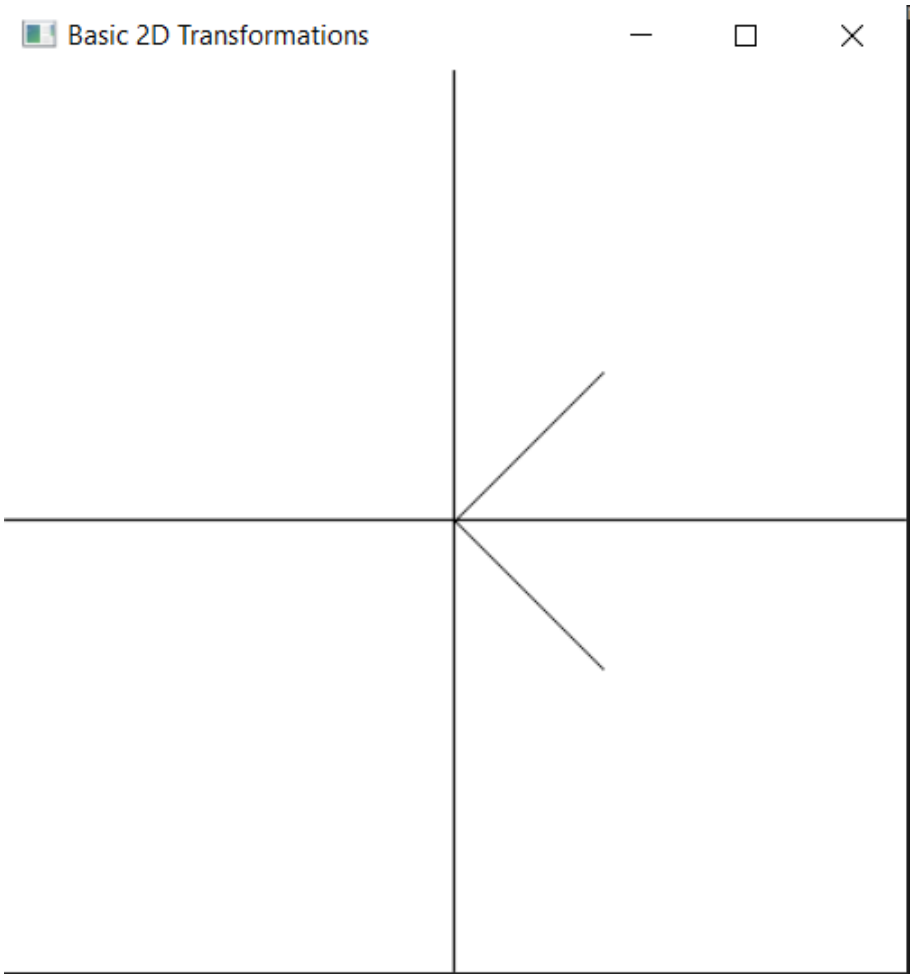
4.

```
PS C:\SSN 7th sem materials\Graphics-and-Multimedia-Lab\198-A5> ./transformations
Enter the number of edges: 2
Enter vertices:
Enter co-ordinates for vertex 1 (X, Y): 0 0
Enter co-ordinates for vertex 2 (X, Y): 200 200
Enter your choice of transformation :
1. Translation
2. Rotation about Origin
3. Rotation wrt fixed point
4. Scaling wrt Origin
5. Scaling wrt fixed point
6. Reflection wrt x-axis
7. Reflection wrt y-axis
8. Reflection wrt origin
9. Reflection wrt line y=x
10. Shearing along x-direction
11. Shearing along y-direction
12. Exit
*****************************
4
Enter the scaling factor for X and Y: 2 2
```



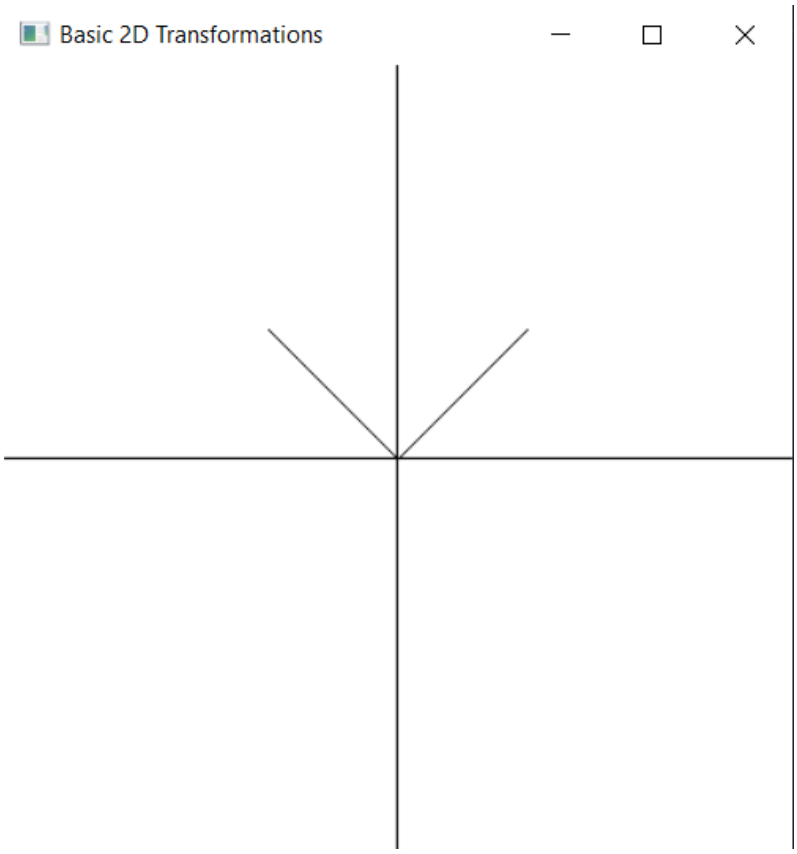
5.

```
PS C:\SSN 7th sem materials\Graphics-and-Multimedia-Lab\198-A5> ./transformations
Enter the number of edges: 2
Enter vertices:
Enter co-ordinates for vertex 1 (X, Y): 0 0
Enter co-ordinates for vertex 2 (X, Y): 200 200
Enter your choice of transformation :
1. Translation
2. Rotation about Origin
3. Rotation wrt fixed point
4. Scaling wrt Origin
5. Scaling wrt fixed point
6. Reflection wrt x-axis
7. Reflection wrt y-axis
8. Reflection wrt origin
9. Reflection wrt line y=x
10. Shearing along x-direction
11. Shearing along y-direction
12. Exit
*****************************
6
```



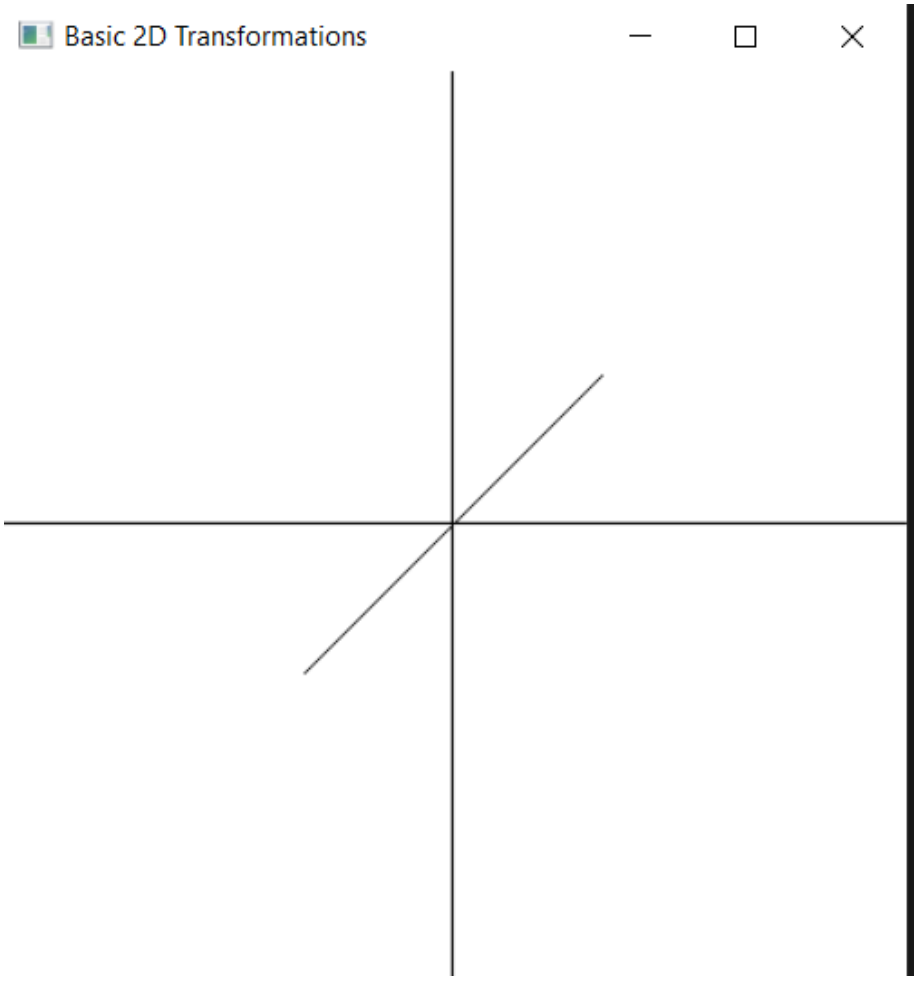
6.

```
PS C:\SSN 7th sem materials\Graphics-and-Multimedia-Lab\198-A5> ./transformations
Enter the number of edges: 2
Enter vertices:
Enter co-ordinates for vertex 1 (X, Y): 0 0
Enter co-ordinates for vertex 2 (X, Y): 200 200
Enter your choice of transformation :
1. Translation
2. Rotation about Origin
3. Rotation wrt fixed point
4. Scaling wrt Origin
5. Scaling wrt fixed point
6. Reflection wrt x-axis
7. Reflection wrt y-axis
8. Reflection wrt origin
9. Reflection wrt line y=x
10. Shearing along x-direction
11. Shearing along y-direction
12. Exit
*****************************
7
```



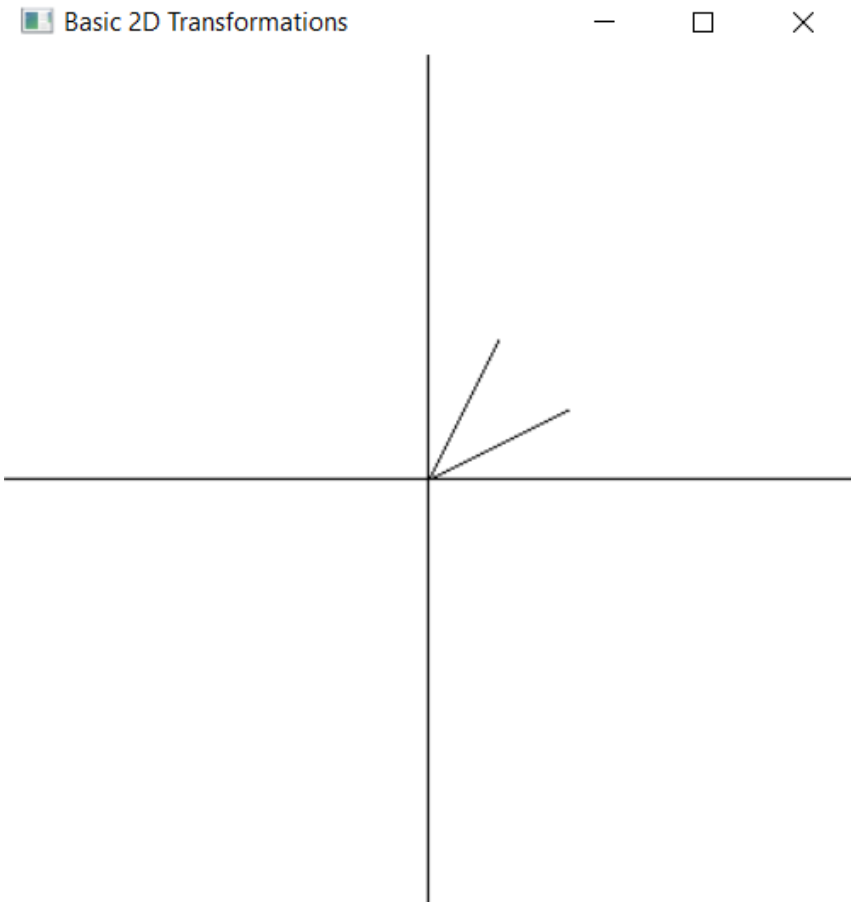
7.

```
PS C:\SSN 7th sem materials\Graphics-and-Multimedia-Lab\198-A5> ./transformations
Enter the number of edges: 2
Enter vertices:
Enter co-ordinates for vertex 1 (X, Y): 0 0
Enter co-ordinates for vertex 2 (X, Y): 200 200
Enter your choice of transformation :
1. Translation
2. Rotation about Origin
3. Rotation wrt fixed point
4. Scaling wrt Origin
5. Scaling wrt fixed point
6. Reflection wrt x-axis
7. Reflection wrt y-axis
8. Reflection wrt origin
9. Reflection wrt line y=x
10. Shearing along x-direction
11. Shearing along y-direction
12. Exit
*****************************
8
```



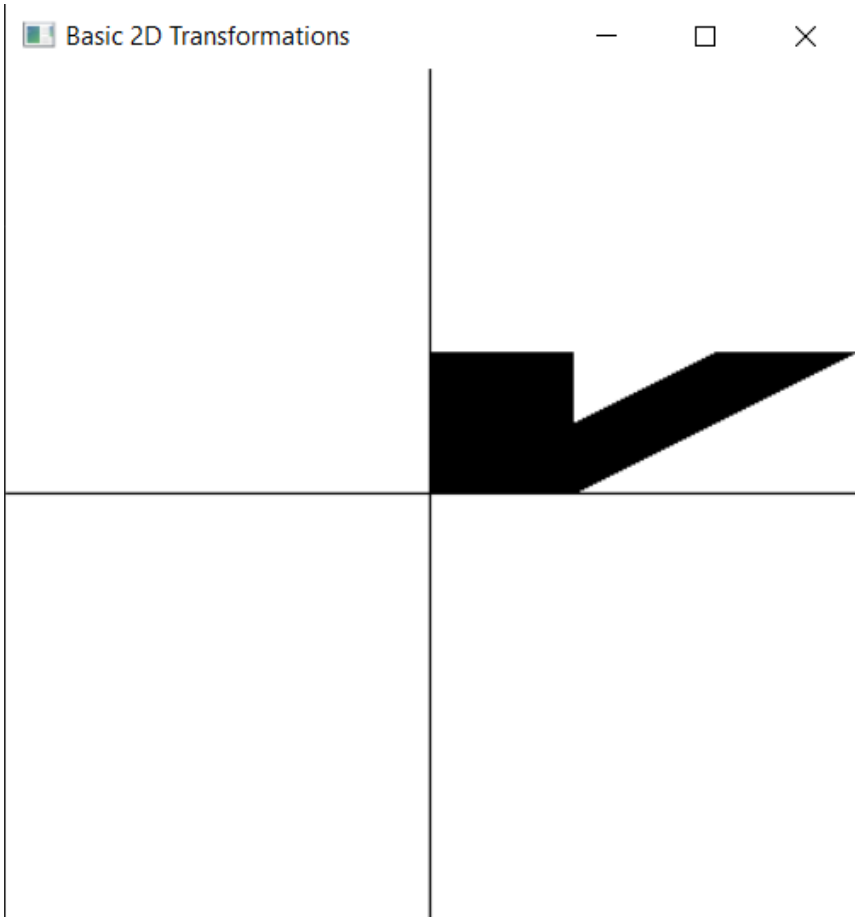
8.

```
PS C:\SSN 7th sem materials\Graphics-and-Multimedia-Lab\198-A5> ./transformations
Enter the number of edges: 2
Enter vertices:
Enter co-ordinates for vertex 1 (X, Y): 0 0
Enter co-ordinates for vertex 2 (X, Y): 200 100
Enter your choice of transformation :
1. Translation
2. Rotation about Origin
3. Rotation wrt fixed point
4. Scaling wrt Origin
5. Scaling wrt fixed point
6. Reflection wrt x-axis
7. Reflection wrt y-axis
8. Reflection wrt origin
9. Reflection wrt line y=x
10. Shearing along x-direction
11. Shearing along y-direction
12. Exit
*****************************
9
```



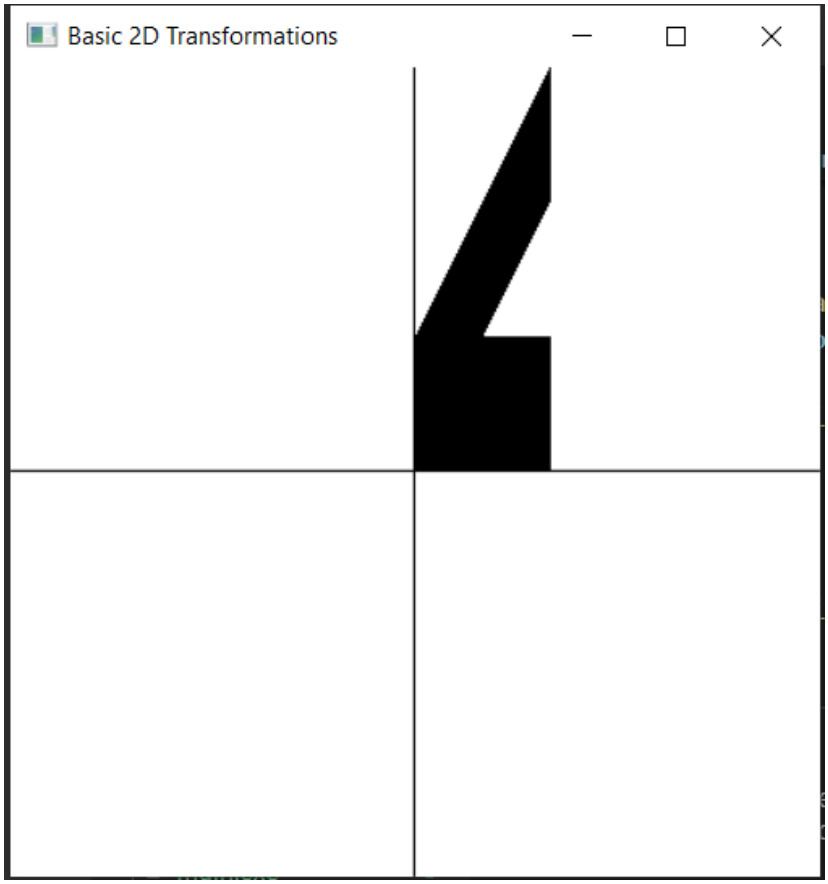
9.

```
PS C:\SSN 7th sem materials\Graphics-and-Multimedia-Lab\198-A5> ./transformations
Enter the number of edges: 4
Enter vertices:
Enter co-ordinates for vertex 1 (X, Y): 0 0
Enter co-ordinates for vertex 2 (X, Y): 200 0
Enter co-ordinates for vertex 3 (X, Y): 200 200
Enter co-ordinates for vertex 4 (X, Y): 0 200
Enter your choice of transformation :
1. Translation
2. Rotation about Origin
3. Rotation wrt fixed point
4. Scaling wrt Origin
5. Scaling wrt fixed point
6. Reflection wrt x-axis
7. Reflection wrt y-axis
8. Reflection wrt origin
9. Reflection wrt line y=x
10. Shearing along x-direction
11. Shearing along y-direction
12. Exit
*****************************
10
Enter the x-shearing factor:
2
```



10.

```
PS C:\SSN 7th sem materials\Graphics-and-Multimedia-Lab\198-A5> ./transformations
Enter the number of edges: 4
Enter vertices:
Enter co-ordinates for vertex 1 (X, Y): 0 0
Enter co-ordinates for vertex 2 (X, Y): 200 0
Enter co-ordinates for vertex 3 (X, Y): 200 200
Enter co-ordinates for vertex 4 (X, Y): 0 200
Enter your choice of transformation :
1. Translation
2. Rotation about Origin
3. Rotation wrt fixed point
4. Scaling wrt Origin
5. Scaling wrt fixed point
6. Reflection wrt x-axis
7. Reflection wrt y-axis
8. Reflection wrt origin
9. Reflection wrt line y=x
10. Shearing along x-direction
11. Shearing along y-direction
12. Exit
*****************************
11
Enter the y-shearing factor:
2
```



11.

```
PS C:\SSN 7th sem materials\Graphics-and-Multimedia-Lab\198-A5> ./transformations
Enter the number of edges: 4
Enter vertices:
Enter co-ordinates for vertex 1 (X, Y): 0 0
Enter co-ordinates for vertex 2 (X, Y): 200 0
Enter co-ordinates for vertex 3 (X, Y): 200 200
Enter co-ordinates for vertex 4 (X, Y): 0 200
Enter your choice of transformation :
1. Translation
2. Rotation about Origin
3. Rotation wrt fixed point
4. Scaling wrt Origin
5. Scaling wrt fixed point
6. Reflection wrt x-axis
7. Reflection wrt y-axis
8. Reflection wrt origin
9. Reflection wrt line y=x
10. Shearing along x-direction
11. Shearing along y-direction
12. Exit
*****************************
1
Enter the translation factor for X and Y: 300 300
```