**SSN COLLEGE OF ENGINEERING**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**UCS1712 – GRAPHICS AND MULTIMEDIA LAB**

**EX NO: 5b – 2D Transformations – Reflection and Shearing**

**Name: Nachammai Devi Pooja S**

**RollNo:185001096**

**Date:29/08/2020**

**Aim:**

To write a C++ menu-driven program using OPENGL to perform 2D transformations – reflection and shearing for polygons.

**Algorithm:**

Step 1: Obtain no. of edges of polygon from user

Step 2: Obtain coordinates of vertices

Step 3: Plot the original polygon and line

Step 4: Obtain transformation option from user

Step 5: Reflection:

* Along X axis:

multiply -1 to the Y coordinates of the original polygon and plot

* Along Y axis:

multiply -1 to the X coordinates of the original polygon and plot

* About origin:

multiply -1 to both the X & Y coordinates of the original polygon and plot

* Along X=Y:

swap the X and Y coordinates of the original polygon and plot

Step 7: Shearing:

* Along X axis:
* Read the shearing factor for X
* Add Shearing factor to top most vertices of the polygon and plot
* Along Y axis:
* Read the shearing factor for Y
* Add Shearing factor to rightmost most vertices of the polygon and plot

**Code:**

#include <stdio.h>

#include <math.h>

#include <iostream>

#include <vector>

#include <gl/glut.h>

using namespace std;

int pntx1, pnty1, choice = 0, edges;

vector<int> pntx;

vector<int> pnty;

int transx, transy;

double scalex, scaley;

double angle, anglerad;

char reflectionaxis, shearingaxis;

int shearingx, shearingy;

double round(double d)

{return floor(d + 0.5);}

void drawPolygon()

{

glBegin(GL\_POLYGON);

glColor3f(0.4, 0, 0.2);

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

{

glVertex2i(pntx[i], pnty[i]);

}

glEnd();

}

void reflection(char reflectionaxis)

{

glBegin(GL\_POLYGON);

glColor3f(6.08, 0.67, 1.0);

if (reflectionaxis == 'x' || reflectionaxis == 'X')

{

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

{

glVertex2i(round(pntx[i]), round(pnty[i] \* -1));

}

}

else if (reflectionaxis == 'y' || reflectionaxis == 'Y')

{

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

{

glVertex2i(round(pntx[i] \* -1), round(pnty[i]));

}

}

else if (reflectionaxis == 'o' || reflectionaxis == 'O')

{

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

{

glVertex2i(round(pntx[i] \* -1), round(pnty[i] \* -1));

}

}

else if (reflectionaxis == 's' || reflectionaxis == 'S')

{

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

{

glVertex2i(round(pnty[i]+100), round(pntx[i]));

}

}

glEnd();

}

void shearing()

{

glBegin(GL\_POLYGON);

glColor3f(0.3, 0.4, 0.7);

if (shearingaxis == 'x' || shearingaxis == 'X')

{

glVertex2i(pntx[0]+200, pnty[0]);

glVertex2i(pntx[1] + shearingx+200, pnty[1]);

glVertex2i(pntx[2] + shearingx+200, pnty[2]);

glVertex2i(pntx[3]+200, pnty[3]);

}

else if (shearingaxis == 'y' || shearingaxis == 'Y')

{ glVertex2i(pntx[0]+200, pnty[0]);

glVertex2i(pntx[1]+200, pnty[1]);

glVertex2i(pntx[2]+200, pnty[2] + shearingy);

glVertex2i(pntx[3]+200, pnty[3] + shearingy);

}

glEnd();

}

void myInit(void)

{

glClearColor(1.0, 1.0, 1.0, 0.0);

glColor3f(0.0f, 0.0f, 0.0f);

glPointSize(4.0);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

gluOrtho2D(-640.0, 640.0, -480.0, 480.0);

}

void myDisplay(void)

{

while (true) {

glClear(GL\_COLOR\_BUFFER\_BIT);

glColor3f(0.0, 0.0, 0.0);

drawPolygon();

glBegin(GL\_LINES);

glVertex2d(-640, 0);

glVertex2d(640, 0);

glVertex2d(0, 480);

glVertex2d(0, -480);

glEnd();

cout << "\nSelect the required transformation:\n";

cout << "1. Reflection\n";

cout << "2. Shearing\n";

cout << "3. Exit\n";

cout << "Enter your choice : ";

cin >> choice;

if (choice == 3) {

return;

}

if (choice == 1)

{

cout << "Enter reflection axis ( x / y / origin(O) / same(s)): "; cin >> reflectionaxis;

reflection(reflectionaxis);

}

else if (choice == 2)

{

cout << "Enter shearing axis ( x or y ): "; cin >> shearingaxis;

if (shearingaxis == 'x' || shearingaxis == 'X')

{

cout << "Enter the shearing factor for X: "; cin >> shearingx;

shearing();

}

else if (shearingaxis == 'y' || shearingaxis == 'Y')

{

cout << "Enter the shearing factor for Y: "; cin >> shearingy;

shearing();

}

}glFlush();

}

}

void main(int argc, char\*\* argv)

{

cout << "\n2D-Transformations\n" << endl;

cout << "Enter no of edges: "; cin >> edges;

cout << "\nEnter Polygon Coordinates : \n";

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

cout << "Vertex " << i + 1 << " : "; cin >> pntx1 >> pnty1;

pntx.push\_back(pntx1);

pnty.push\_back(pnty1);}

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);

glutInitWindowSize(640, 480);

glutInitWindowPosition(100, 150);

glutCreateWindow("2D-Transformations");

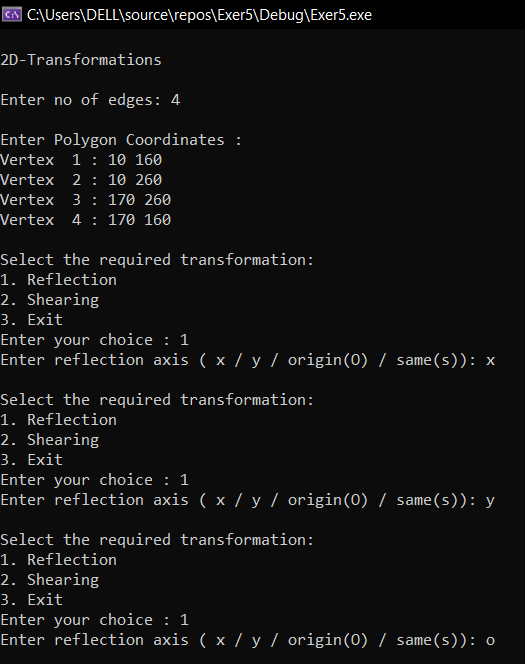
glutDisplayFunc(myDisplay);

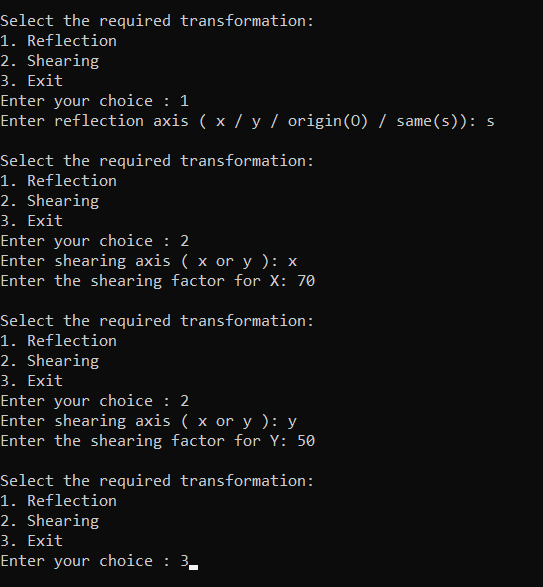
myInit();

glutMainLoop();

}

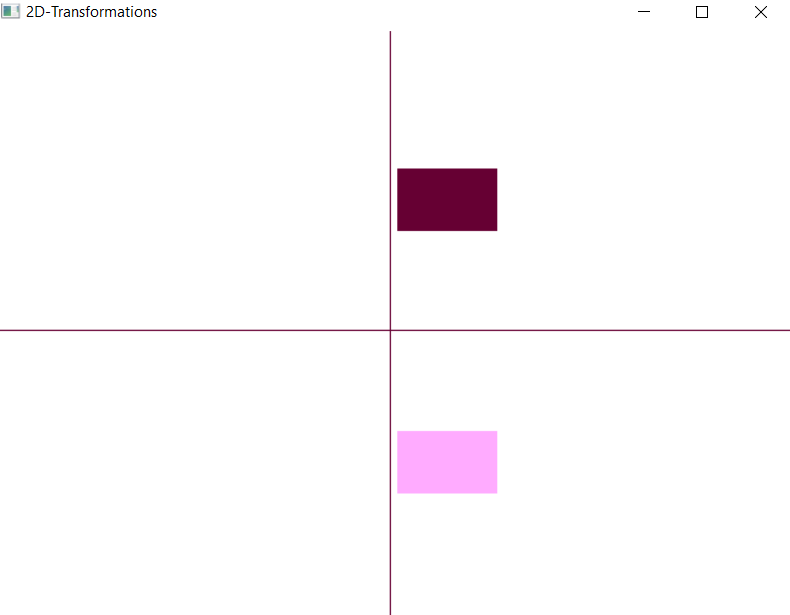
**OUTPUT:**

****

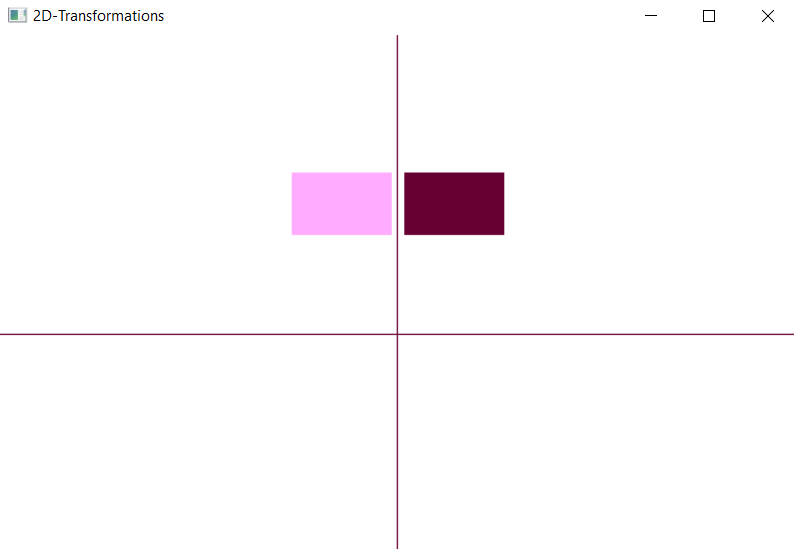
****

**1)Reflection:**

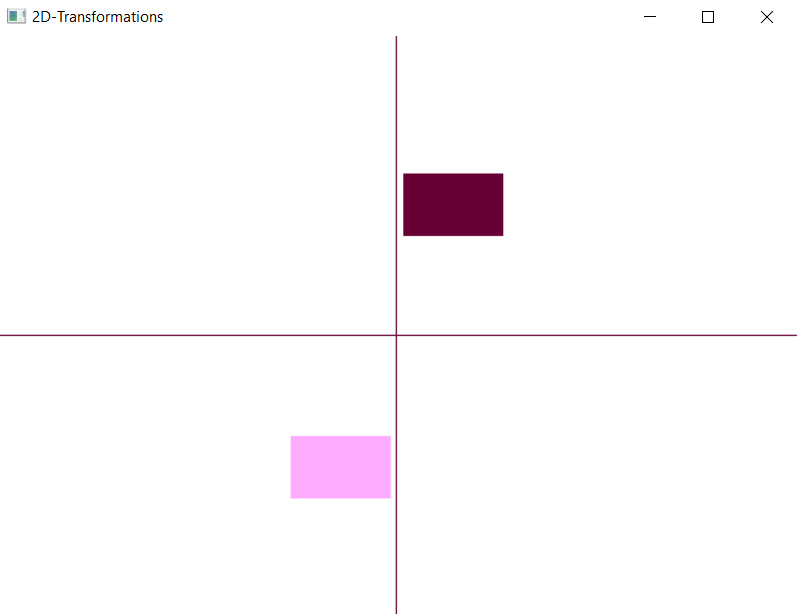
**->Along X:**

****

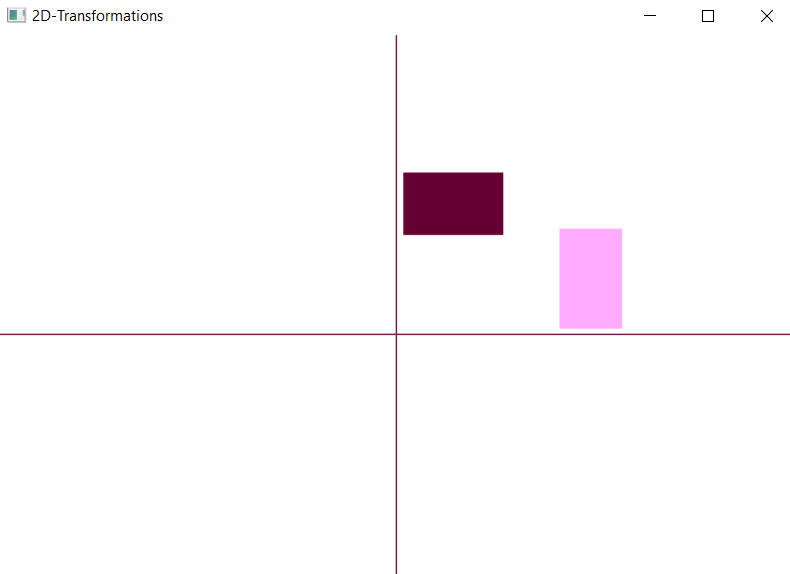
**->Along Y:**

****

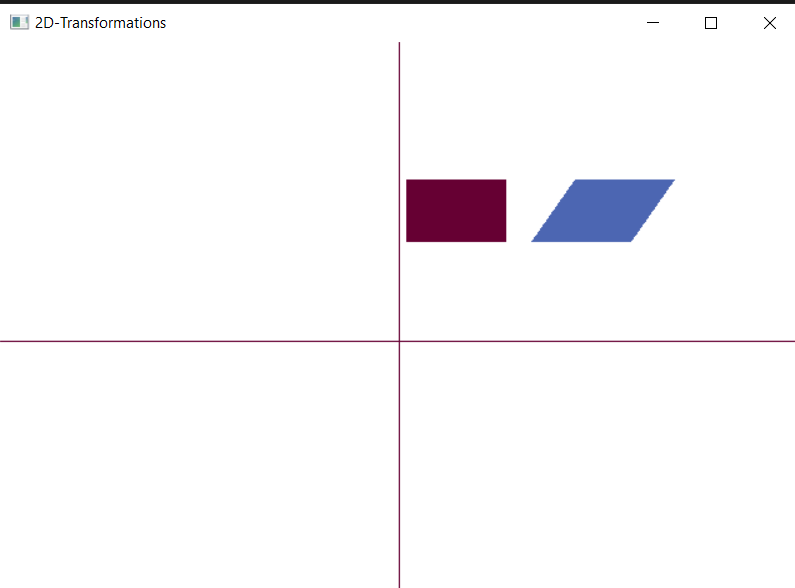
**->About Origin:**

****

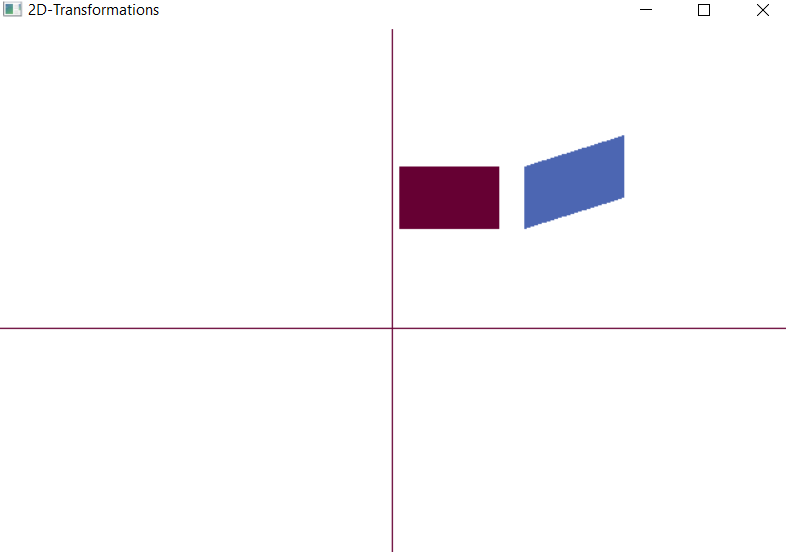
**->Along X=Y:**

****

**2)Shearing:**

**->Along X:**

**->Along Y:**

****

**Result:**

A C++ menu-driven program using OPENGL to perform 2D transformations – reflection and shearing for polygon was written and implemented successfully.