Name -Isha Kishor Katariya

Class - SEIT Div - A

Roll No – 207A056

Experiment - Implement following 2D transformations on the object with respect to axis: i) Scaling ii) Rotation about arbitrary point iii) Reflection

#include <iostream>

#include <math.h> #include <time.h>

#include <GL/glut.h> #include <vector> using namespace std; int edge; vector<int> xpoint; vector<int> ypoint; int ch;

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

} void init(){ glClearColor(1.0,

1.0, 1.0, 0.0);

glMatrixMode(GL\_PROJECTION); gluOrtho2D(0,

640, 0, 480);

glClear(GL\_COLOR\_BUFFER\_BIT);

}

void translation(){ int tx, ty;

cout << "\t Enter Tx, Ty \n"; cin >> tx >> ty; // Translate the point for (int i = 0; i < edge; i++){ xpoint[i] = xpoint[i] + tx; ypoint[i] = ypoint[i] + ty;

} glBegin(GL\_POLYGON);

glColor3f(0, 0, 1); for (int i = 0; i

< edge; i++){ glVertex2i(xpoint[i], ypoint[i]);

} glEnd(); glFlush(); }void rotaion(){ int cx, cy;

cout << "\n Enter Ar point x , y "; cin >> cx >> cy; cx = cx + 320; cy = cy +

240; glColor3f(0.0, 1.0,

0.0);

glBegin(GL\_POINTS);

glVertex2i(cx, cy); glEnd(); glFlush(); double the; cout << "\n Enter thetha "; cin >> the; the = the \* 3.14 / 180;

glColor3f(0, 0, 1.0);

glBegin(GL\_POLYGON);

for (int i = 0; i < edge; i++){ glVertex2i(round(((xpoint[i] - cx) \* cos(the) -

((ypoint[i] - cy) \* sin(the))) + cx), round(((xpoint[i] - cx) \* sin(the) +

((ypoint[i] - cy) \* cos(the))) + cy));

} glEnd(); glFlush(); } void scale(){ glColor3f(1.0, 0, 0);

glBegin(GL\_POLYGON);

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

glVertex2i(xpoint[i] - 320, ypoint[i] - 240);

} glEnd(); glFlush();

cout << "\n\tIn Scaling whole screen is 1st Qudrant \n"; int sx, sy; cout << "\t Enter sx, sy \n"; cin >> sx >> sy;// scale the point for (int i = 0; i < edge; i++){ xpoint[i] = (xpoint[i] - 320) \* sx; ypoint[i] = (ypoint[i] - 240) \* sy; } glColor3f(0, 0, 1.0);

glBegin(GL\_POLYGON);

for (int i = 0; i < edge; i++){ glVertex2i(xpoint[i], ypoint[i]);

} glEnd(); glFlush(); } void reflection(){ char reflection;

cout << "Enter Reflection Axis \n"; cin >> reflection; if (reflection == 'x' || reflection == 'X'){ glColor3f(0.0, 0.0,

1.0);

glBegin(GL\_POLYGON);

for (int i = 0; i < edge; i++){ glVertex2i(xpoint[i],

(ypoint[i] \* -1) + 480);

} glEnd(); glFlush();

}

else if (reflection == 'y' || reflection == 'Y'){ glColor3f(0.0, 0.0, 1.0);

glBegin(GL\_POLYGON);

for (int i = 0; i < edge; i++){ glVertex2i((xpoint[i] \* -

1) + 640, (ypoint[i]));

} glEnd(); glFlush(); } }

void Draw(){if (ch == 2 || ch == 3 || ch == 4){ glColor3f(1.0, 0, 0);

glBegin(GL\_LINES); glVertex2i(0, 240); glVertex2i(640, 240); glEnd(); glColor3f(1.0, 0, 0);

glBegin(GL\_LINES); glVertex2i(320, 0); glVertex2i(320, 480); glEnd();

glFlush(); glColor3f(1.0, 0, 0);

glBegin(GL\_POLYGON);

for (int i = 0; i < edge; i++){ glVertex2i(xpoint[i], ypoint[i]);

} glEnd(); glFlush(); } if (ch == 1){ scale(); } else if (ch == 2){ rotaion(); } else if (ch == 3){ reflection();

} else if (ch ==

4){ translation();

} }

int main(int argc, char \*\*argv){ cout << "\n \t Enter 1) Scaling ";

cout << "\n \t Enter 2) Rotation about arbitrary point";cout << "\n \t Enter 3)

Reflection"; cout << "\n \t Enter 4) Translation \n \t"; cin >> ch; if (ch == 1 || ch

== 2 || ch == 3 || ch == 4){ cout << "Enter No of edges \n"; cin >> edge; int xpointnew, ypointnew; cout << " Enter" << edge << " point of polygon \n"; for (int i = 0; i < edge; i++){ cout << "Enter " << i << " Point "; cin >> xpointnew >> ypointnew; xpoint.push\_back(xpointnew +

320); ypoint.push\_back(ypointnew

+ 240); }

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB); glutInitWindowSize(640, 480); glutInitWindowPosition(200, 200);

glutCreateWindow("2D");

init();

glutDisplayFunc(Draw); glutMainLoop(); return

0; } else { cout << "\n \t Check Input run again"; return 0;

}

}