// Name – Dahiwal Satyam Santoshkumar

// Roll Number – 207019

#include <stdio.h>

#include <math.h>

#include <GL/glut.h>

#include <iostream>

#include <math.h>

#define MAX 10

int n,op,tx,ty,sx,sy,angle,shx,shy,y,z,xr,yr;

float a[MAX][3],b[3][3],c[MAX][3];

/\* Function to plot a point \*/

void setPixel(GLint x, GLint y) {

glBegin(GL\_POINTS);

glVertex2i(x,y);

glEnd();

}

// Function of Rounding.

inline int round(const GLfloat a)

{

return int(a+0.5);

}

// DDA Line Algorithm

void lineDDA(int x0,int y0,int xEnd,int yEnd)

{

int dx = xEnd-x0;

int dy = yEnd-y0;

int steps,k;

GLfloat xIncrement,yIncrement,x=x0,y=y0;

if(abs(dx) > abs(dy))

steps = abs(dx);

else

steps = abs(dy);

xIncrement = GLfloat(dx) / GLfloat(steps);

yIncrement = GLfloat(dy) / GLfloat(steps);

setPixel(round(x),round(y));

for(k=1;k<steps;k++)

{

x+= xIncrement;

y+= yIncrement;

setPixel(round(x),round(y));

}

glFlush();

}

void axis()

{

int i,x,y;

glClear(GL\_COLOR\_BUFFER\_BIT);

glClearColor(1.0,1.0,1.0,1);

glClear(GL\_COLOR\_BUFFER\_BIT);

glColor3f (0.0, 1.0, 0.0);

lineDDA(320,30,320,450); //x axis

lineDDA(20,240,620,240); //y axis

glColor3f (0.0, 1.0, 1.0);

lineDDA(0,0,640,480);//digonal

lineDDA(0,480,640,0);//digonal

for(i=-10,x=20;i<=10;i++,x+=30) //horizontal line numbering

{

glColor3f (0.0, 1.0, 0.0);

lineDDA(x,238,x,242);

}

for(i=7,y=30;i>=-7;i--,y+=30) //vertical line numbering

{

glColor3f (0.0, 1.0, 0.0);

lineDDA(315,y,325,y);

}

}

void accept()

{

int i;

printf("Enter the coordinate : ");

for(i=0;i<n;i++)

{

printf("\nx%d,y%d : ",i+1,i+1);

scanf("%f%f",&a[i][0],&a[i][1]);

a[i][2]=1;

}

a[n][0]=a[0][0];

a[n][1]=a[0][1];

a[n][2]=1;

}

void show()

{

int i;

axis();

glColor3f (1.0, 0.0, 0.0);

// INPUT DATA

for(i=0;i<n;i++)

lineDDA(320+a[i][0],240+a[i][1],320+a[(i+1)%n][0],240+a[(i+1)%n][1]);//homogeneous form

glColor3f (0.0, 0.0, 1.0);

// OUTPUT DATA

for(i=0;i<n;i++)

lineDDA(320+c[i][0],240+c[i][1],320+c[(i+1)%n][0],240+c[(i+1)%n][1]);//homogeneous form

}

void mul()

{

int i,j,k;

for(i=0;i<n;i++)

{

for(j=0;j<3;j++)

{

c[i][j]=0;

}

}

for(i=0;i<n;i++)

{

for(j=0;j<3;j++)

{

for(k=0;k<3;k++)

{

c[i][j]=c[i][j]+a[i][k]\*b[k][j];

}

}

}

}

void trans()

{

int i,j;

for(i=0;i<3;i++)

{

for(j=0;j<3;j++)

{

if(i==j)

b[i][j]=1;

else

b[i][j]=0;

}

}

b[2][0]=tx;

b[2][1]=ty;

}

void scal()

{

b[0][0]=sx;

b[1][1]=sy;

b[2][2]=1;

b[0][1]=b[1][0]=b[1][2]=b[0][2]=b[2][1]=b[2][0]=0;

}

void rota()

{

int i,j;

float rad;

if(z==1)

angle = angle ;

else

angle = angle \* -1;

rad=(angle\*3.14)/180;

b[0][0]=b[1][1]=cos(rad);

b[2][2]=1;

b[0][1]=sin(rad);

b[1][0]=-sin(rad);

b[2][0]=b[2][1]=b[1][2]=0;

for(i=0;i<3;i++)

{

for(j=0;j<3;j++)

{

printf("%f",b[i][j]);

}

printf("\n");

}

}

void rot\_ref()

{

float rad;

int sign;

rad=(angle\*3.14)/180;

if(z==1)

sign=1;

else

sign=-1;

b[0][0]=b[1][1]=cos(rad);

b[2][2]=1;

b[0][1]=sign\*sin(rad);

b[1][0]=-sign\*sin(rad);

b[2][0]=-xr\*cos(rad)+yr\*sign\*sin(rad)+xr;

b[2][1]=-xr\*sin(rad)+yr\*(-sign)\*cos(rad)+yr;

b[0][2]=b[1][2]=0;

}

void shear()

{

int i,j;

for(i=0;i<3;i++)

{

for(j=0;j<3;j++)

{

if(i==j)

b[i][j]=1;

else

b[i][j]=0;

}

}

b[0][1]=shx;

b[1][0]=shy;

}

void reflection()

{

int i;

b[2][2]=1;

b[0][1]=b[1][0]=b[1][2]=b[0][2]=b[2][1]=b[2][0]=0;

switch(z)

{

case 1 : b[0][0]=1;

b[1][1]=-1;

break;

case 2 : b[0][0]=-1;

b[1][1]=1;

break;

case 3 : b[0][0]=-1;

b[1][1]=-1;

break;

case 4 : b[0][0]=0;

b[1][1]=0;

b[0][1]=1;

b[1][0]=1;

break;

case 5 : b[0][0]=0;

b[1][1]=0;

b[0][1]=-1;

b[1][0]=-1;

break;

}

}

void choice(void)

{

printf("\n\n\*\*\*\*\*\*\*\*\*\*\* 2D Transformations \*\*\*\*\*\*\*\*\*");

printf("\n\nEnter the no of vertices of polygon : ");

scanf("%d",&n);

accept();

printf("\n\*\*\*\*\*\*\*\*\*\*\*MENU\*\*\*\*\*\*\*\*\*\*");

printf("\n \n1) Translation \n2) Scaling \n3) Rotation \n4) Rotation of arbitary \n5) Shearing \n6) Reflection");

printf("\nEnter your choice : ");

scanf("%d",&op);

switch(op)

{

case 1: printf("\nThe polygon before translation");

printf("\nEnter the tx : ");

scanf("%d",&tx);

printf("\nEnter the ty : ");

scanf("%d",&ty);

trans();

mul();

show();

break;

case 2: printf("\nThe polygon before scaling");

printf("\nEnter the sx");

scanf("%d",&sx);

printf("\nEnter the sy");

scanf("%d",&sy);

scal();

mul();

show();

break;

case 3: printf("\nThe polygon befor rotation");

printf("\nEnter the angle : ");

scanf("%d",&angle);

printf("\nPress 1 for anticlockwise and 2 for clockwise : ");

scanf("%d",&z);

rota();

mul();

show();

break;

case 4: printf("\nThe polygon befor rotation");

printf("\nEnter the angle : ");

scanf("%d",&angle);

printf("\nPress 1 for anticlockwise and 2 for clockwise : ");

scanf("%d",&z);

printf("Enter the x and y coordinate : ");

scanf("%d%d",&xr,&yr);

rot\_ref();

mul();

show();

break;

case 5: printf("\nThe polygon before Shearing");

printf("\nEnter the Shx : ");

scanf("%d",&shx);

printf("\nEnter the Shy : ");

scanf("%d",&shy);

shear();

mul();

show();

break;

case 6: printf("\nThe polygon before Reflection");

printf("\n-----------------------------");

printf("\n\t1. Against X-axis\n\t2. Against Y-axis\n\t3. Against Origin");

printf("\n\t4. X = Y\n\t5. X = -Y\n\tEnter you Choice : ");

scanf("%d",&z);

reflection();

mul();

show();

break;

default: printf("Invalid option");

}

}

void init()

{

/\* Set the initial display mode \*/

glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);

/\* Set the initial window position and size \*/

glutInitWindowPosition(0,0);

glutInitWindowSize(640,480);

/\* Create the window with title "2D Transformations" \*/

glutCreateWindow("2D Transformations");

/\* Set clear color to white \*/

glClearColor(1.0,1.0,1.0,1.0);

/\* Set fill color to black \*/

glColor3f(0.0,0.0,0.0);

/\* glViewport(0 , 0 , 640 , 480); \*/

/\* glMatrixMode(GL\_PROJECTION); \*/

/\* glLoadIdentity(); \*/

glPointSize(1.0f);

gluOrtho2D(0 , 640 , 0 , 480);

}

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

{

glutInit(&argc, argv);

choice();

init();

glutDisplayFunc(show);

glutMainLoop();

return 0;

}

Output

satyam@ubuntu:~$ g++ seven.cpp -lglut -lGL -lGLEW -lGLU -o seven

satyam@ubuntu:~$ ./seven

\*\*\*\*\*\*\*\*\*\*\* 2D Transformations \*\*\*\*\*\*\*\*\*

Enter the no of vertices of polygon : 3

Enter the coordinate :

x1,y1 : 30

30

x2,y2 : 60

6

x3,y3 : 90

30

\*\*\*\*\*\*\*\*\*\*\*MENU\*\*\*\*\*\*\*\*\*\*

1) Translation

2) Scaling

3) Rotation

4) Rotation of arbitary

5) Shearing

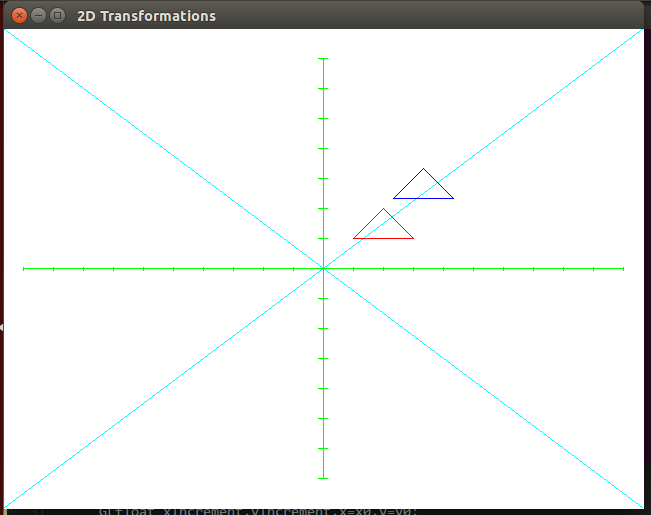
6) Reflection

Enter your choice : 1

The polygon before translation

Enter the tx : 40

Enter the ty : 40



satyam@ubuntu:~$ ./seven

\*\*\*\*\*\*\*\*\*\*\* 2D Transformations \*\*\*\*\*\*\*\*\*

Enter the no of vertices of polygon : 3

Enter the coordinate :

x1,y1 : 30

30

x2,y2 : 60

60

x3,y3 : 90

60

\*\*\*\*\*\*\*\*\*\*\*MENU\*\*\*\*\*\*\*\*\*\*

1) Translation

2) Scaling

3) Rotation

4) Rotation of arbitary

5) Shearing

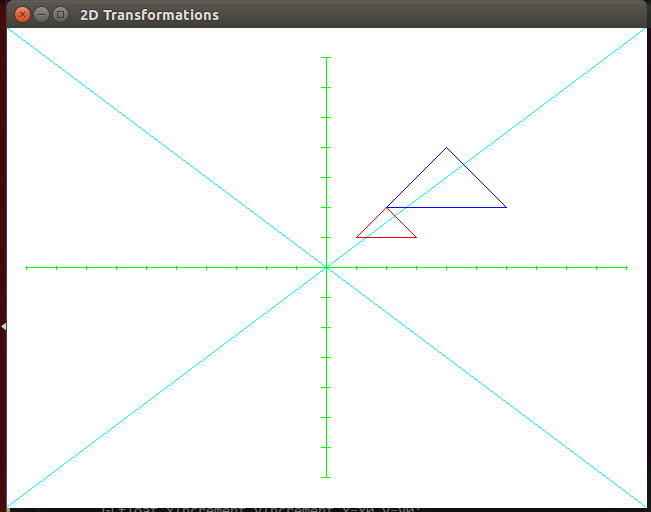
6) Reflection

Enter your choice : 2

The polygon before scaling

Enter the sx2

Enter the sy2



satyam@ubuntu:~$ ./seven

\*\*\*\*\*\*\*\*\*\*\* 2D Transformations \*\*\*\*\*\*\*\*\*

Enter the no of vertices of polygon : 3

Enter the coordinate :

x1,y1 : 30

30

x2,y2 : 60

60

x3,y3 : 90

30

\*\*\*\*\*\*\*\*\*\*\*MENU\*\*\*\*\*\*\*\*\*\*

1) Translation

2) Scaling

3) Rotation

4) Rotation of arbitary

5) Shearing

6) Reflection

Enter your choice : 3

The polygon befor rotation

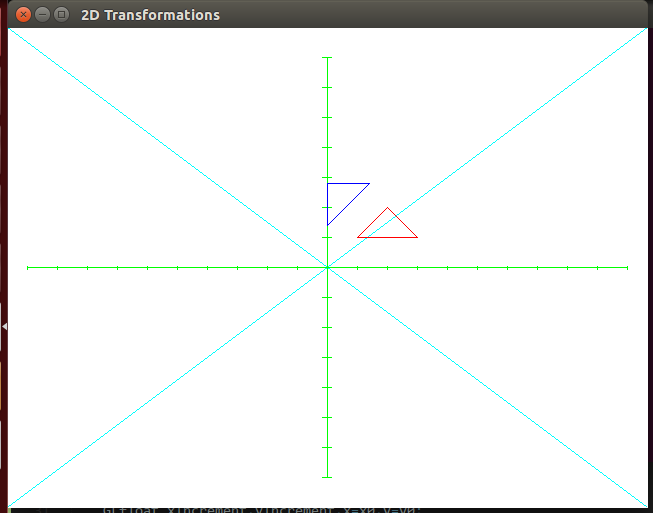
Enter the angle : 45

Press 1 for anticlockwise and 2 for clockwise : 1

0.7073880.7068250.000000

-0.7068250.7073880.000000

0.0000000.0000001.000000



satyam@ubuntu:~$ ./seven

\*\*\*\*\*\*\*\*\*\*\* 2D Transformations \*\*\*\*\*\*\*\*\*

Enter the no of vertices of polygon : 3

Enter the coordinate :

x1,y1 : 30

30

x2,y2 : 90

90

x3,y3 : 140

30

\*\*\*\*\*\*\*\*\*\*\*MENU\*\*\*\*\*\*\*\*\*\*

1) Translation

2) Scaling

3) Rotation

4) Rotation of arbitary

5) Shearing

6) Reflection

Enter your choice : 4

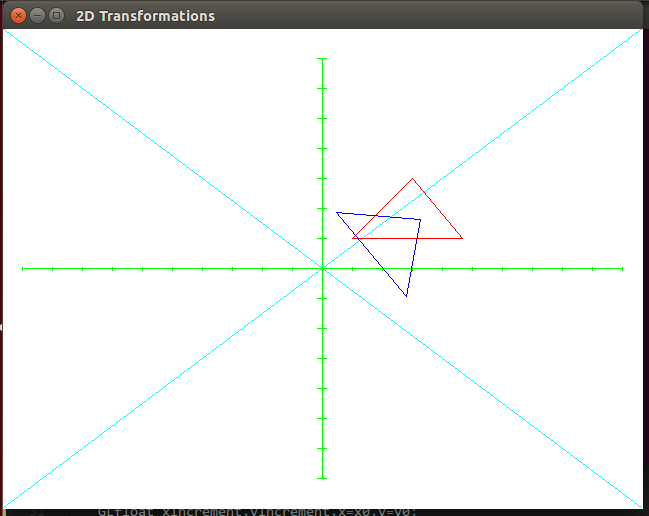
The polygon befor rotation

Enter the angle : 50

Press 1 for anticlockwise and 2 for clockwise : 2

Enter the x and y coordinate : 50

60



satyam@ubuntu:~$ ./seven

\*\*\*\*\*\*\*\*\*\*\* 2D Transformations \*\*\*\*\*\*\*\*

Enter the no of vertices of polygon : 30

Enter the coordinate :

x1,y1 : ^C

satyam@ubuntu:~$ ./seven

\*\*\*\*\*\*\*\*\*\*\* 2D Transformations \*\*\*\*\*\*\*\*\*

Enter the no of vertices of polygon : 3

Enter the coordinate :

x1,y1 : 30

30

x2,y2 : 60

60

x3,y3 : 90

30

\*\*\*\*\*\*\*\*\*\*\*MENU\*\*\*\*\*\*\*\*\*\*

1) Translation

2) Scaling

3) Rotation

4) Rotation of arbitary

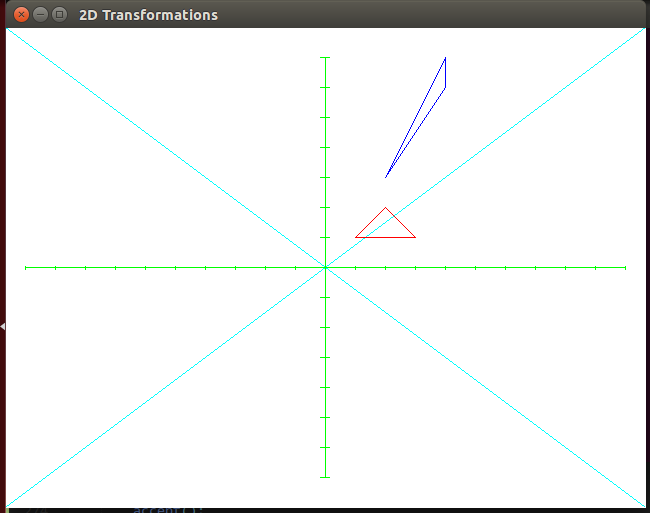
5) Shearing

6) Reflection

Enter your choice : 5

The polygon before Shearing

Enter the Shx : 2

Enter the Shy : 1

satyam@ubuntu:~$ ./seven

\*\*\*\*\*\*\*\*\*\*\* 2D Transformations \*\*\*\*\*\*\*\*\*

Enter the no of vertices of polygon : 3

Enter the coordinate :

x1,y1 : 30

30

x2,y2 : 90

90

x3,y3 : 140

20

\*\*\*\*\*\*\*\*\*\*\*MENU\*\*\*\*\*\*\*\*\*\*

1) Translation

2) Scaling

3) Rotation

4) Rotation of arbitary

5) Shearing

6) Reflection

Enter your choice : 6

The polygon before Reflection

-----------------------------

1. Against X-axis

2. Against Y-axis

3. Against Origin

4. X = Y

5. X = -Y

Enter you Choice : 3

