**Write C++/Java program to implement translation, sheer, rotation and scaling transformations on equilateral triangle and rhombus using QT Creator.**

#ifndef TRANSFORMATION\_H

#define TRANSFORMATION\_H

#include <QWidget>

class transformation : public QWidget

{

Q\_OBJECT

public:

transformation(QWidget \*parent = 0);

void *paintEvent*(QPaintEvent \*e);

void scaling(QPainter \* qp);

private:

};

#endif // TRANSFORMATION\_H

#include "transformation.h"

#include <QPainter>

#include<iostream>

#include<math.h>

using namespace std;

transformation::transformation(QWidget \*a):QWidget(a)

{

}

void transformation::*paintEvent*(QPaintEvent \*e)

{

Q\_UNUSED(e);

QPainter qp(this);

scaling(&qp);

}

void transformation::scaling(QPainter \*qp)

{

QPen p(Qt::red,2,Qt::SolidLine);

qp->setPen(p);

qp->drawLine(620,20,620,700);

qp->drawLine(20,340,1200,340);

int a[3][3]={{0,0,1},{100,0,1},{50,70,1}};//{{20,100,1},{120,100,1},{70,50,1}};

int j=0;

for(int i=0;i<2;i++)

{

qp->drawLine(a[i][j]+620,340-a[i][j+1],a[i+1][j]+620,340-a[i+1][j+1]);

if(i==1)

qp->drawLine(a[i+1][j]+620,340-a[i+1][j+1],a[0][0]+620,340-a[0][1]);

}

qp->drawText(a[0][0]+620,a[0][1],"Original Triangle");

int sx=2;

int sy=2;

int s[3][3]={{sx,0,0},{0,sy,0},{0,0,1}};

int res[3][3]={0};

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

{

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

{

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

{

res[i][j]+=a[i][k]\*s[k][j];

}

}

}

int t3[3][3]={{1,0,0},{0,1,0},{150,150,1}};

int rest3[3][3]={0};

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

{

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

{

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

{

rest3[i][j]+=res[i][k]\*t3[k][j];

}

}

}

j=0;

for(int i=0;i<2;i++)

{

qp->drawLine(rest3[i][j]+620,340-rest3[i][j+1],rest3[i+1][j]+620,340-rest3[i+1][j+1]);

if(i==1)

qp->drawLine(rest3[i+1][j]+620,340-rest3[i+1][j+1],rest3[0][0]+620,340-rest3[0][1]);

}

qp->drawText(res[0][0]+620+150,340-170-res[0][1],"Scaled  & translated Triangle");

int yshear[3][3]={{1,2,0},{0,1,0},{0,0,1}};

int resyshear[3][3]={0};

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

{

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

{

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

{

resyshear[i][j]+=a[i][k]\*yshear[k][j];

}

}

}

int t1[3][3]={{1,0,0},{0,1,0},{10,100,1}};

int rest1[3][3]={0};

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

{

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

{

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

{

rest1[i][j]+=resyshear[i][k]\*t1[k][j];

}

}

}

j=0;

for(int i=0;i<2;i++)

{

qp->drawLine(rest1[i][j]+620,340-rest1[i][j+1],rest1[i+1][j]+620,340-rest1[i+1][j+1]);

if(i==1)

qp->drawLine(rest1[i+1][j]+620,340-rest1[i+1][j+1],rest1[0][0]+620,340-rest1[0][1]);

}

qp->drawText(resyshear[0][0]+560,210-resyshear[0][1],"y sheared &  translated Triangle");

int xshear[3][3]={{1,0,0},{2,1,0},{0,0,1}};

int resxshear[3][3]={0};

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

{

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

{

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

{

resxshear[i][j]+=a[i][k]\*xshear[k][j];

}

}

}

int t2[3][3]={{1,0,0},{0,1,0},{150,0,1}};

int rest[3][3]={0};

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

{

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

{

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

{

rest[i][j]+=resxshear[i][k]\*t2[k][j];

}

}

}

j=0;

for(int i=0;i<2;i++)

{

qp->drawLine(rest[i][j]+620,340-rest[i][j+1],rest[i+1][j]+620,340-rest[i+1][j+1]);

if(i==1)

qp->drawLine(rest[i+1][j]+620,340-rest[i+1][j+1],rest[0][0]+620,340-rest[0][1]);

}

qp->drawText(resxshear[0][0]+620+200,340-resxshear[0][1],"x sheared & translated Triangle ");

float r[3][3]={{cos(30),-sin(30),0},{sin(30),cos(30),0},{0,0,1}};

float resr[3][3]={0};

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

{

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

{

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

{

resr[i][j]+=a[i][k]\*r[k][j];

}

}

}

j=0;

for(int i=0;i<2;i++)

{

qp->drawLine(resr[i][j]+620,340-resr[i][j+1],resr[i+1][j]+620,340-resr[i+1][j+1]);

if(i==1)

qp->drawLine(resr[i+1][j]+620,340-resr[i+1][j+1],resr[0][0]+620,340-resr[0][1]);

}

qp->drawText(resr[0][0]+570,310-resr[0][1],"Rotated Triangle ");

}

#include "transformation.h"

#include <QApplication>

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

{

QApplication a(argc, argv);

transformation w;

w.show();

return a.exec();

}

**Output -**

