// Triangles.java: This makes a chessboard of squares

// Austin Sypolt CS447

import java.awt.\*;

import java.awt.event.\*;

public class Squares extends Frame

{ public static void main(String[] args){new Squares();}

Squares()

{ super("Squares everywhere!");

addWindowListener(new WindowAdapter()

{public void windowClosing(WindowEvent e){System.exit(0);}});

setSize(640,640);

add(new newSquares());

show();

}

}

class newSquares extends Canvas

{ int maxX, maxY, minMaxXY, xCenter, yCenter;

void initgr()

{ Dimension d = getSize();

maxX = d.width - 1; maxY = d.height - 1;

minMaxXY = Math.min(maxX, maxY);

//xCenter = maxX/2; yCenter = maxY/2;

xCenter = maxX/2;

yCenter = maxY/2;

}

int iX(float x){return Math.round(x);}

int iY(float y){return maxY - Math.round(y);}

public void paint(Graphics g)

{

initgr();

float side = 0.95F \* minMaxXY, sideHalf = side,

h = sideHalf, //\* (float)Math.sqrt(3),

xA, yA, xB, yB, xC, yC,

xA1, yA1, xB1, yB1, xC1, yC1, xD, yD, xD1, yD1, p, q;

q = 0.2F;//0.05F;

p = 1 - q;

for (int k = 0; k < 8; k++){

for(int j = 0; j < 8; j++){

xA = (xCenter + sideHalf)/8 + (80\*k); yA = (yCenter - 0.5F \* h)/4 + (80\*j);

xB = (xCenter - sideHalf)/8 + (80\*k); yB = yA/4 + (80\*j);

xC = (xCenter - sideHalf)/8 + (80\*k); yC = (yCenter + 0.5F \* h)/4 + (80\*j);

xD = (xCenter + sideHalf)/8 + (80\*k); yD = yC/4 + (80\*j);

//xA = 40+(80\*k); yA = 40+(80\*j);

//xB = 40-(80\*k); yB = 40-(80\*j);

//xC = xA; yC = yA;

//xD = xB; yD = yB;

//xA = (xCenter - sideHalf)+(160\*k); yA = yCenter - 0.5F \* h + (160\*j);

//xB = xCenter + sideHalf+(160\*k); yB = yA + (160\*j);

//xC = xCenter - sideHalf+(160\*k); yC = yCenter + 0.5F \* h + (160\*j);

//xD = xCenter + sideHalf+(160\*k); yD = yC + (160\*j);

for (int i=0; i<15; i++)

{

g.setColor(Color.magenta);

g.drawLine(iX(xA), iY(yA), iX(xB), iY(yB));

g.setColor(Color.lightGray);

g.drawLine(iX(xB), iY(yB), iX(xC), iY(yC));

g.setColor(Color.green);

g.drawLine(iX(xC), iY(yC), iX(xD), iY(yD));

g.setColor(Color.red);

g.drawLine(iX(xD), iY(yD), iX(xA), iY(yA));

xA1 = p \* xA + q \* xB; yA1 = p \* yA + q \* yB;

xB1 = p \* xB + q \* xC; yB1 = p \* yB + q \* yC;

xC1 = p \* xC + q \* xD; yC1 = p \* yC + q \* yD;

xD1 = p \* xD + q \* xA; yD1 = p \* yD + q \* yA;

xA = xA1; xB = xB1; xC = xC1; xD = xD1;

yA = yA1; yB = yB1; yC = yC1; yD = yD1;

}

}

}

}

}

/\*for(int a = 0; a < 8; a++){

for(int b = 0; b < 8; b++){

//b is for the horizontal axis and a for vertical

if((a+b)%2 == 0){

g.setColor(Color.BLACK);

//g.fillRect(b \* 80,a \* 80, 80, 80);

}

g.drawLine(0, 0, 660, 0);

g.drawLine(0, 0, 0, 700);

g.drawLine(0, 80\*a, 660, 80\*a);

g.drawLine(80\*a, 0, 80\*a, 700);

//g.setColor(Color.magenta);

g.drawLine(10, 0, 80, 10);

g.drawLine(80, 10, 70, 80);

g.drawLine(70, 80, 0, 70);

g.drawLine(0, 70, 10, 0);

}

}\*/