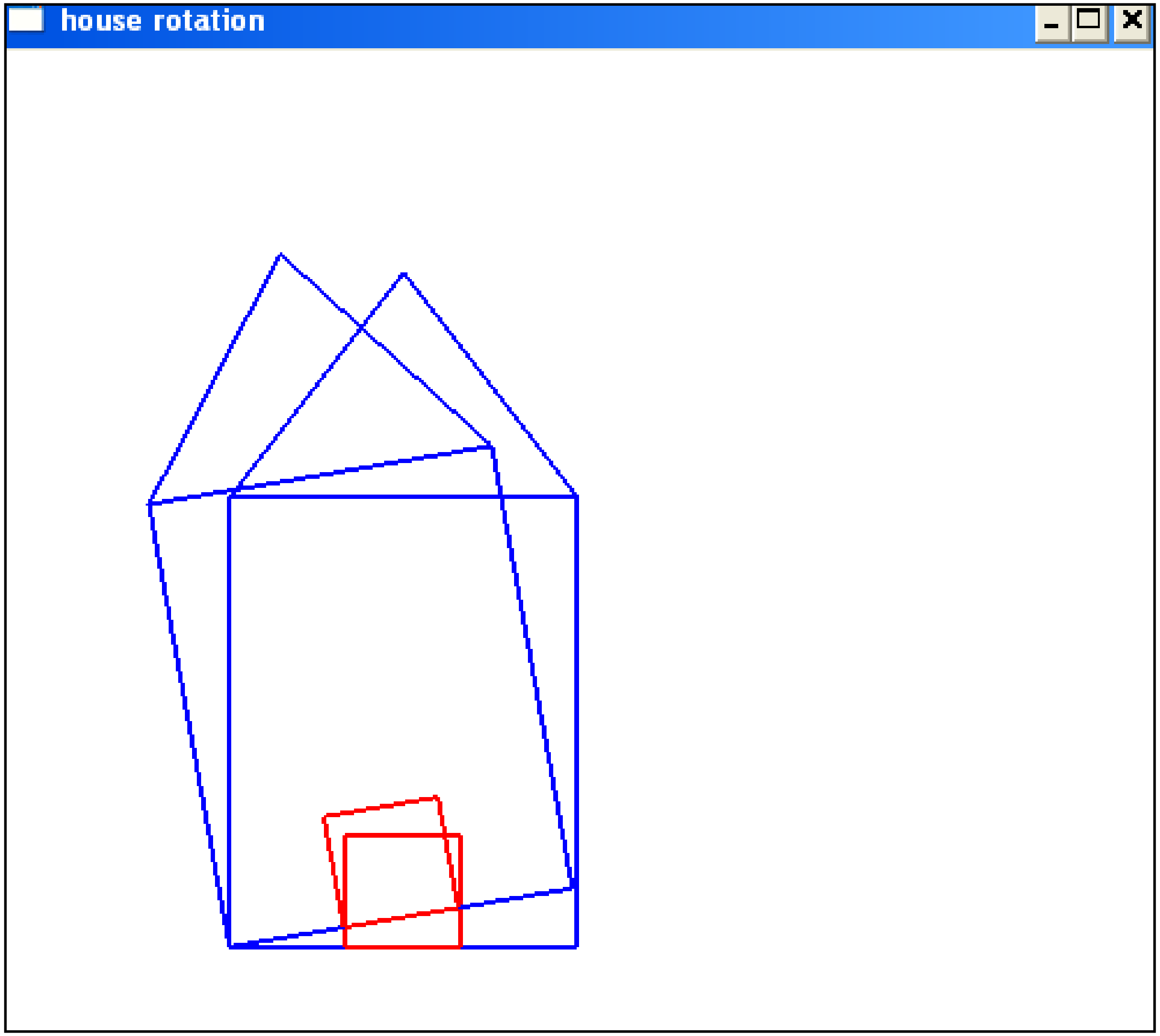
Program 4

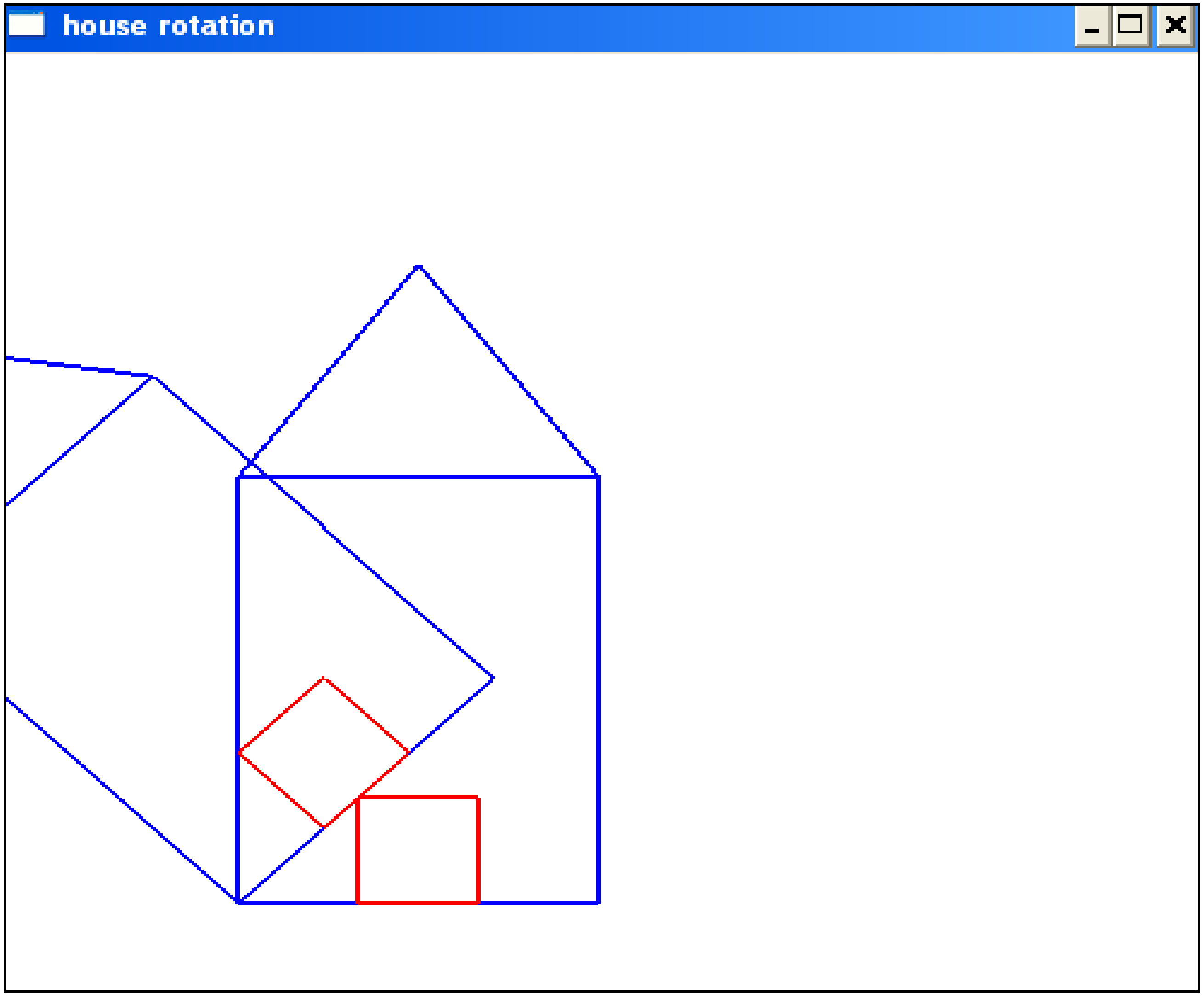
Program to create a house like figure and rotate it about a *given fixed point* using OpenGL functions.

```
Enter the rotation angle
10
```



Enter the rotation angle

45



House co-ordinates

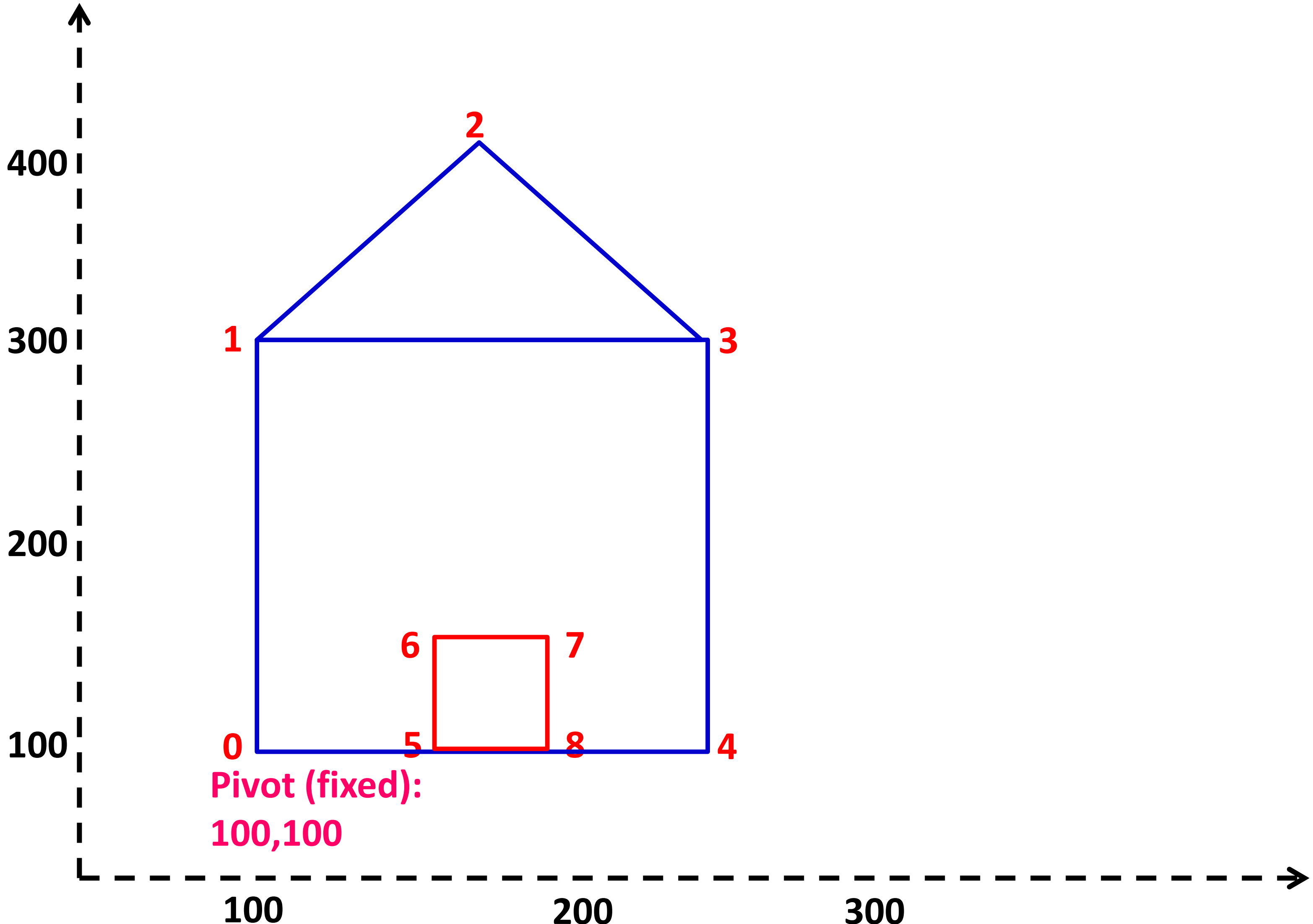
	0	1	2	3	4	5	6	7	8
0(x)	100	100	175	250	250	150	150	200	200
1(y)	100	300	400	300	100	100	150	150	100
2	1	1	1	1	1	1	1	1	1

	0	1	2	3	4	5	6	7	8
0(x)	100	100	175	250	250	150	150	200	200
1(y)	100	300	400	300	100	100	150	150	100
2	1	1	1	1	1	1	1	1	1

LINE_LOOP : 0,1,3,4

LINE_LOOP : 1,2,3

LINE_LOOP : 5,6,7,8



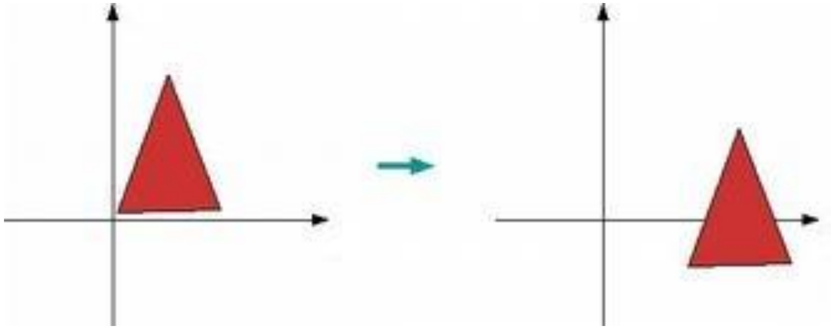
Pivot (fixed):
100,100

2D Transformations in Computer Graphics

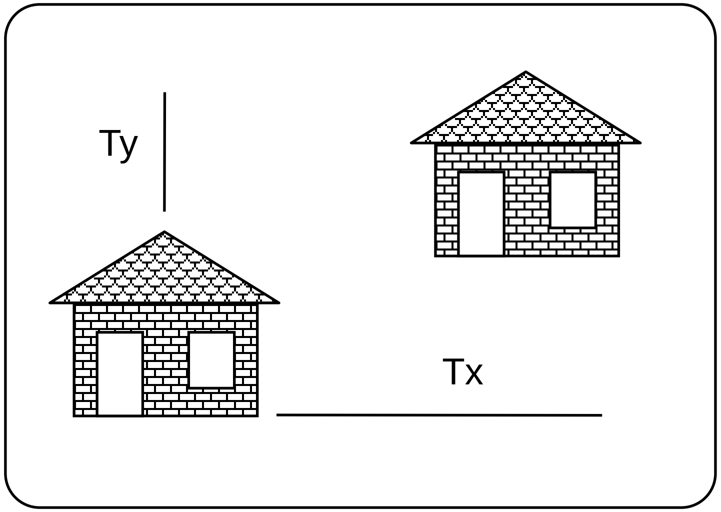
In computer graphics, the 3 transformations are

1. **Translation**
2. **Rotation**
   - About the origin
   - About the fixed (pivot) point
3. **Scaling**

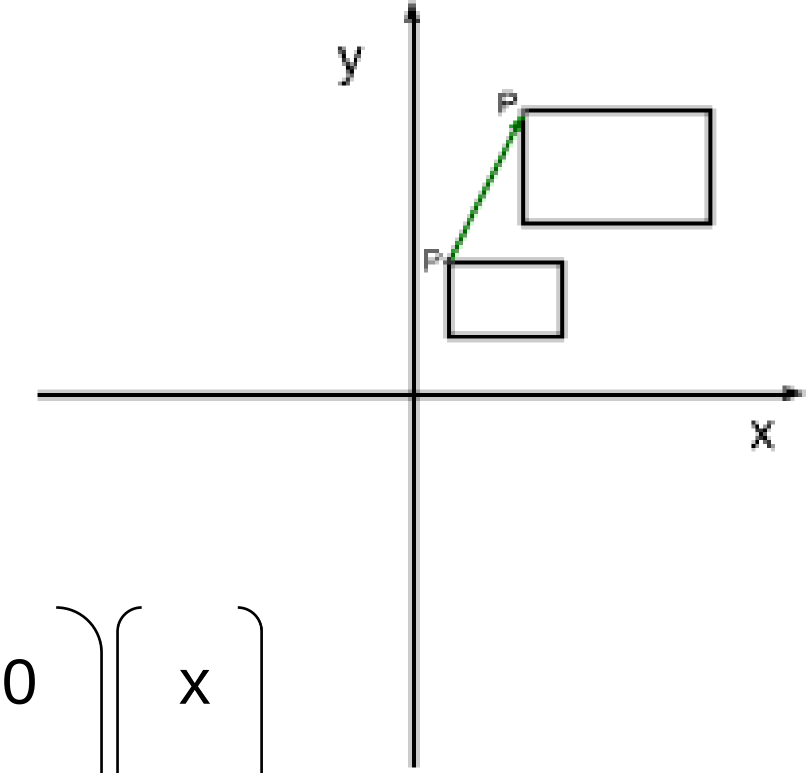
1. Translation



$$x' = x + T_x$$
$$y' = y + T_y$$



2. Scaling

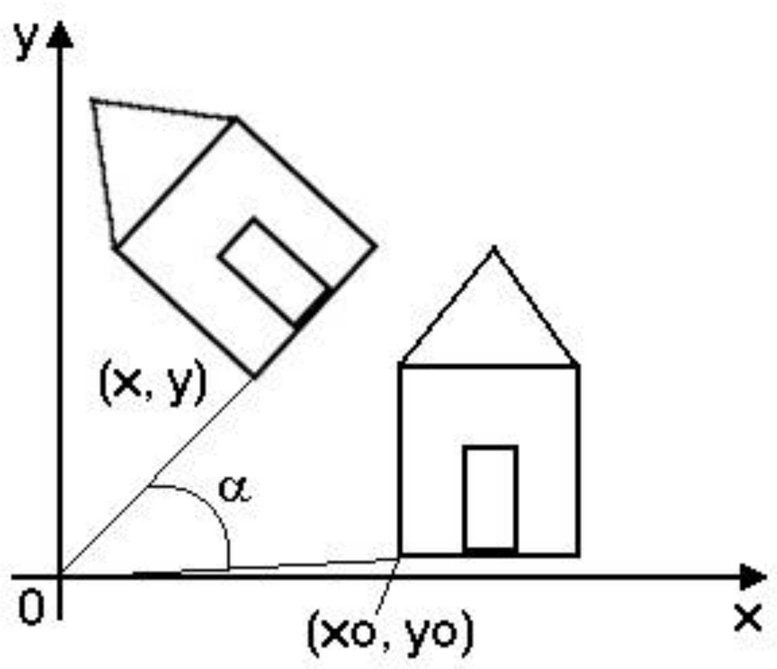


$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} S_x & 0 \\ 0 & S_y \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

3. Rotation

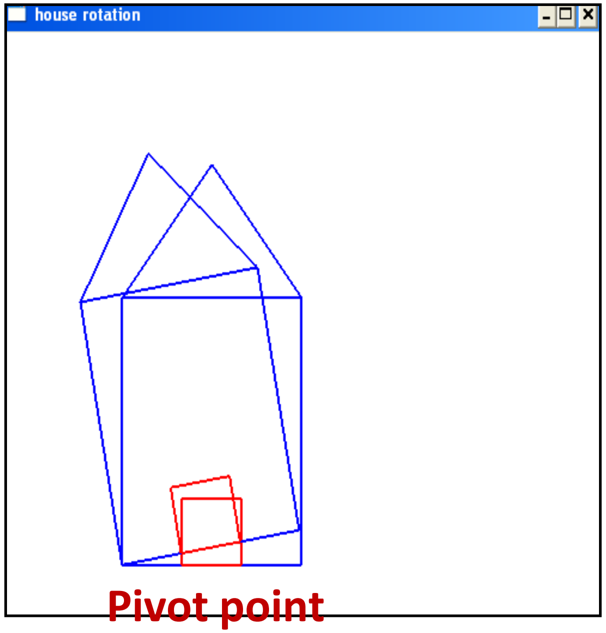
Rotation

about the origin

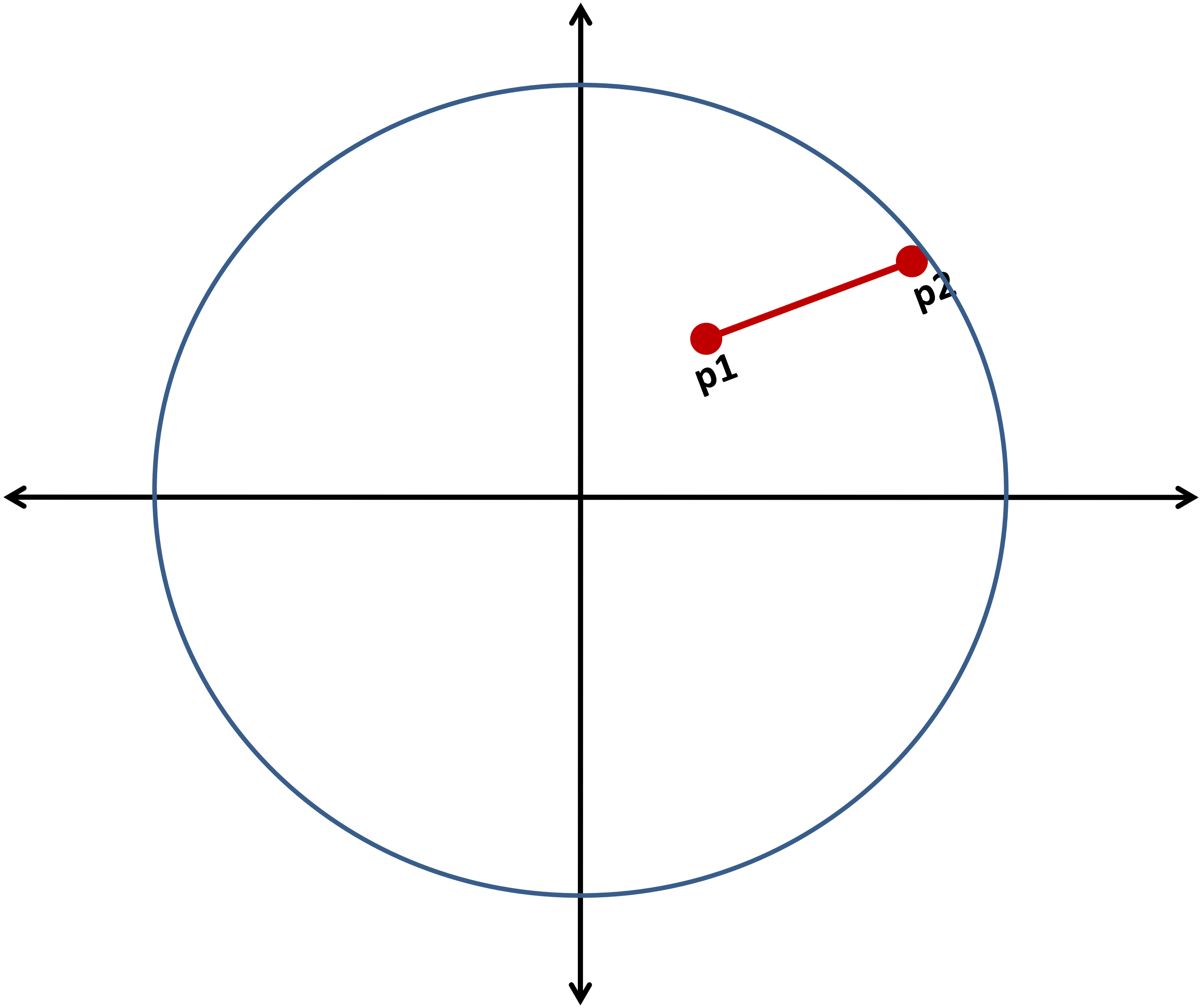


Rotation

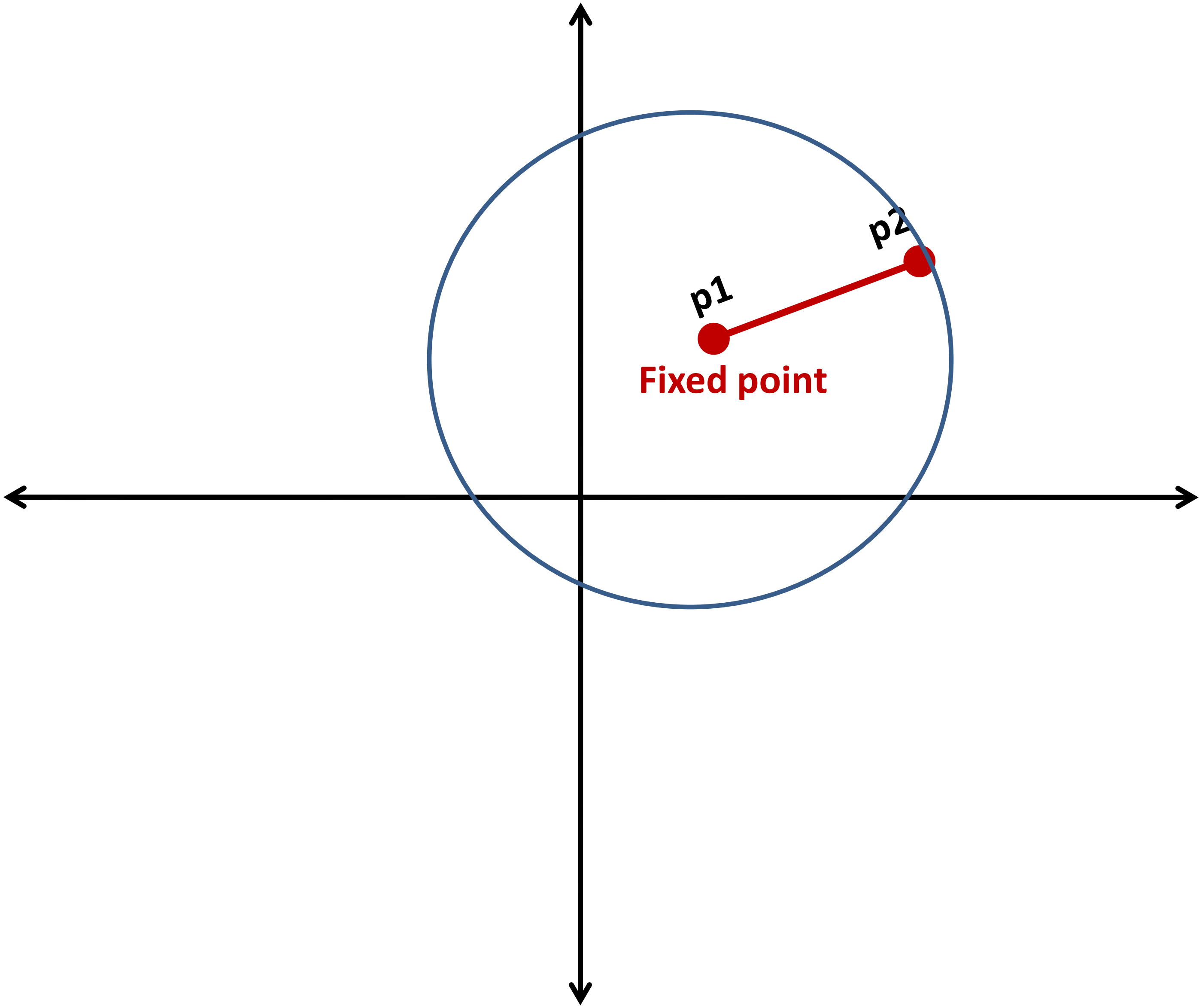
about the fixed (pivot) point



Rotation about the *origin*



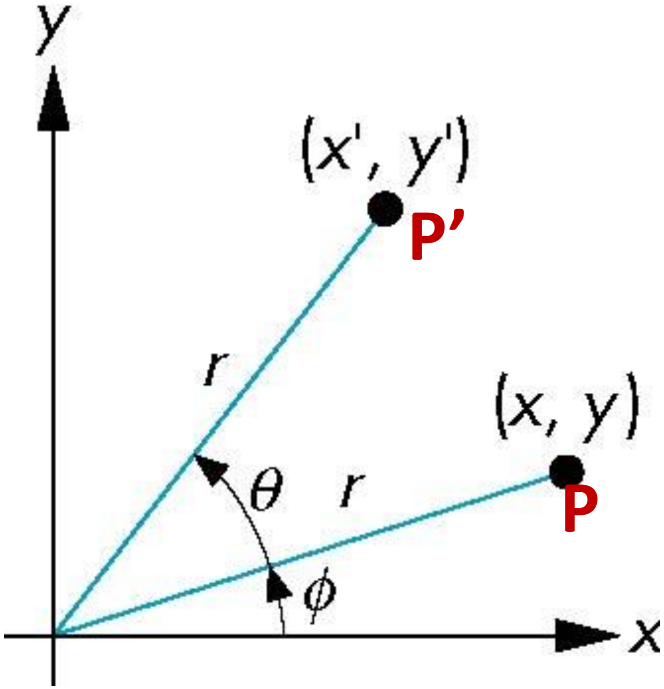
Rotation about the ***fixed point***



Rotation about the origin

Consider rotation about the origin by θ degrees:

The radius r stays the same, angle increases by θ



Original point P(x,y)

$x = r \cos \phi$

$y = r \sin \phi$

Rotated point P'(x',y')

$x' = r \cos (\phi + \theta)$

$y' = r \sin (\phi + \theta)$

Rotation about the origin

Original point p(x,y)

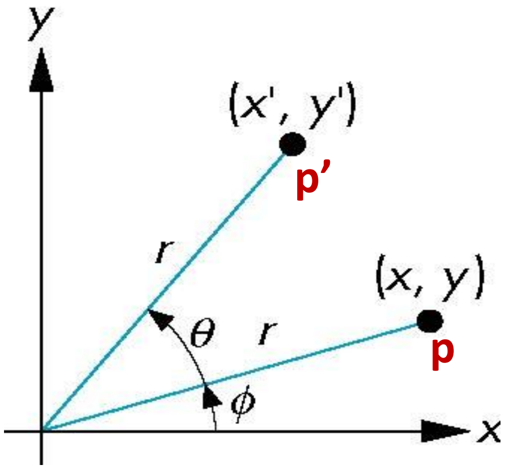
$x = r \cos \phi$

$y = r \sin \phi$

Rotated point p'(x',y')

$x' = r \cos (\phi + \theta)$

$y ‘= r \sin (\phi + \theta)$



WKT

sin(A+B) = sinA cosB + cosA sinB

cos(A+B) = cosA cosB – sinA sinB

Substituting for x' and y'

$x' = r \cos (\phi + \theta)$

$x' = x \cos\theta - y \sin\theta$

$y ‘= r \sin (\phi + \theta)$

$y' = x \sin\theta + y \cos\theta$

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

Rotation about the origin

$x' = x \cos\theta - y \sin\theta$
$y' = x \sin\theta + y \cos\theta$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Homogeneous co-ordinate System

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

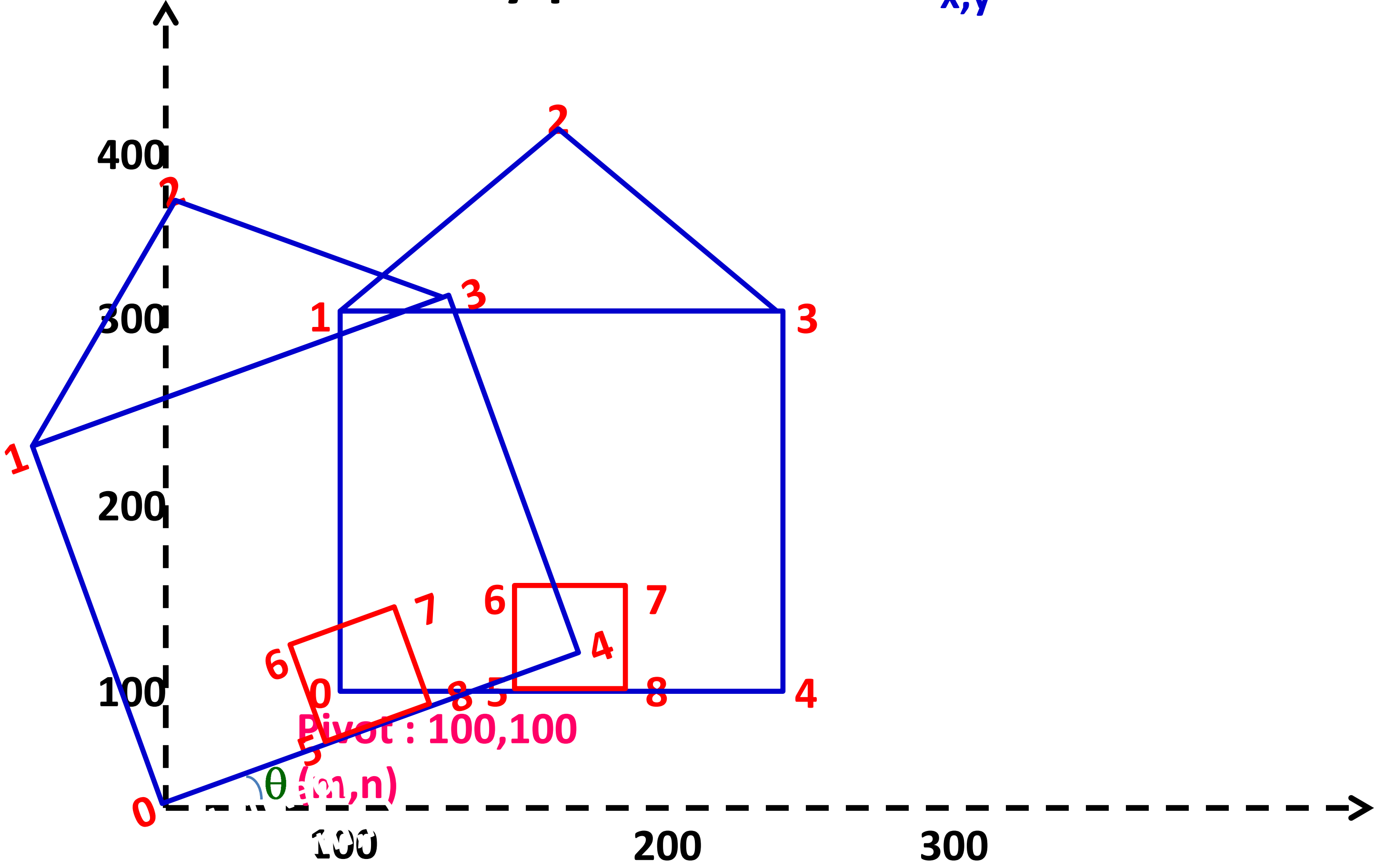
2D Transformations

Transformation	Equation	Homogeneous Equation
Translation	$x' = x + dx$ $y' = y + dy$	$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & dx \\ 0 & 1 & dy \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$
Rotation	$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$	$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$
Scaling	$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} S_x & 0 \\ 0 & S_y \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$	$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$

Rotation about an arbitrary point

Rotation about an arbitrary point (m,n)

1. Translate to origin -----> $T_{-x, -y}$
2. Rotate through θ -----> R_{θ}
3. Translate back to the arbitrary point ------> $T_{x,y}$



Rotation about an arbitrary point (m,n)

1. **Translate to origin -----> $T_{-x,\ -y}$**
2. **Rotate through θ -----> R_{θ}**
3. **Translate back to the arbitrary point ------> $T_{x,\ y}$**

Result C =

$$\underbrace{\begin{pmatrix} 1 & 0 & m \\ 0 & 1 & n \\ 0 & 0 & 1 \end{pmatrix}}_{T_{x,y}} \underbrace{\begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}}_{R_{\theta}} \underbrace{\begin{pmatrix} 1 & 0 & -m \\ 0 & 1 & -n \\ 0 & 0 & 1 \end{pmatrix}}_{T_{-x,-y}}$$

Rotation about an arbitrary point (m,n)

Result C =

$$\underset{T_{x,y}}{\begin{pmatrix} 1 & 0 & m \\ 0 & 1 & n \\ 0 & 0 & 1 \end{pmatrix}} \underset{R_\theta}{\begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}} \underset{T_{-x,-y}}{\begin{pmatrix} 1 & 0 & -m \\ 0 & 1 & -n \\ 0 & 0 & 1 \end{pmatrix}}$$

$$\underset{T_{x,y} \text{ X } R_\theta}{\begin{pmatrix} \cos\theta & -\sin\theta & m \\ \sin\theta & \cos\theta & n \\ 0 & 0 & 1 \end{pmatrix}} \underset{T_{-x,-y}}{\begin{pmatrix} 1 & 0 & -m \\ 0 & 1 & -n \\ 0 & 0 & 1 \end{pmatrix}}$$

$$\underset{T_{x,y} \text{ X } R_\theta \text{ X } T_{-x,-y}}{\begin{pmatrix} \cos\theta & -\sin\theta & -x\cos\theta + y\sin\theta + x \\ \sin\theta & \cos\theta & -x\sin\theta - y\cos\theta + y \\ 0 & 0 & 1 \end{pmatrix}}$$

Rotation Matrix

$$R(\theta) = \begin{pmatrix} \cos\theta & -\sin\theta & -x\cos\theta + y\sin\theta + x \\ \sin\theta & \cos\theta & -x\sin\theta - y\cos\theta + y \\ 0 & 0 & 1 \end{pmatrix}$$

```
#include <stdio.h>
#include <math.h>
#include <GL/glut.h>

GLfloat house[3][9]={
{100.0,100.0,175.0,250.0,250.0,150.0,150.0,200.0,200.0},
{100.0,300.0,400.0,300.0,100.0,100.0,150.0,150.0,100.0},
{1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0}
 };
GLfloat rot_mat[3][3]={   {0}, {0},  {0}  };
GLfloat result[3][9]={  {0}, {0}, {0}  };
GLfloat x=100.0; // Pivot point
GLfloat y=100.0; // Pivot point
GLfloat theta;
```

```
/* Rotation MATRIX and Object Matrix => Resultant
Transformed House */
void multiply()
{
        int i,j,k;
        for(i=0;i<3;i++)
        for(j=0;j<9;j++)
        {
                result[i][j]=0;
                for(k=0;k<3;k++)
                result[i][j]=result[i][j]+rot_mat[i][k]*house[k][j];
        }
}
```

```
// Build the rotation matrix
void rotate()
{
        GLfloat m,n;
        m=x-(x*cos(theta))+(y*sin(theta)); // m=-xcosθ + ysinθ + x
        n=y-(x*sin(theta))-(y*cos(theta));  // n -xsinθ - y cosθ + y
        rot_mat[0][0]=cos(theta);
        rot_mat[0][1]=-sin(theta);
        rot_mat[0][2]=m;
        rot_mat[1][0]=sin(theta);
        rot_mat[1][1]=cos(theta);
        rot_mat[1][2]=n;
        rot_mat[2][0]=0;
        rot_mat[2][1]=0;
        rot_mat[2][2]=1;
        multiply();
}
```

```
void drawhouse()
{
        glColor3f(0.0, 0.0, 1.0);
        glBegin(GL_LINE_LOOP);
                glVertex2f(house[0][0],house[1][0]);
                glVertex2f(house[0][1],house[1][1]);
                glVertex2f(house[0][3],house[1][3]);
                glVertex2f(house[0][4],house[1][4]);
        glEnd();

        glColor3f(1.0,0.0,0.0);
        glBegin(GL_LINE_LOOP);
                glVertex2f(house[0][5],house[1][5]);
                glVertex2f(house[0][6],house[1][6]);
                glVertex2f(house[0][7],house[1][7]);
                glVertex2f(house[0][8],house[1][8]);
        glEnd();
```

```
    glColor3f(0.0, 0.0, 1.0);
    glBegin(GL_LINE_LOOP);
            glVertex2f(house[0][1],house[1][1]);
            glVertex2f(house[0][2],house[1][2]);
            glVertex2f(house[0][3],house[1][3]);
    glEnd();
}
```

```
void drawrotatedhouse()
{
        glColor3f(0.0, 0.0, 1.0);
        glBegin(GL_LINE_LOOP);
                glVertex2f(result[0][0],result[1][0]);
                glVertex2f(result[0][1],result[1][1]);
                glVertex2f(result[0][3],result[1][3]);
                glVertex2f(result[0][4],result[1][4]);
        glEnd();

        glColor3f(1.0,0.0,0.0);
        glBegin(GL_LINE_LOOP);
                glVertex2f(result[0][5],result[1][5]);
                glVertex2f(result[0][6],result[1][6]);
                glVertex2f(result[0][7],result[1][7]);
                glVertex2f(result[0][8],result[1][8]);
        glEnd();
        glColor3f(0.0, 0.0, 1.0);
        glBegin(GL_LINE_LOOP);
                glVertex2f(result[0][1],result[1][1]);
                glVertex2f(result[0][2],result[1][2]);
                glVertex2f(result[0][3],result[1][3]);
        glEnd();
}
```

```
void display()
{
    glClear(GL_COLOR_BUFFER_BIT);
    drawhouse();
    rotate();
    drawrotatedhouse();
    glFlush();
}

void myinit()
{
        glClearColor(1.0,1.0,1.0,1.0);
        glColor3f(1.0,0.0,0.0);
        glMatrixMode(GL_PROJECTION);
        glLoadIdentity();
        gluOrtho2D(0.0,499.0,0.0,499.0);
}
```

```
void main(int argc, char** argv)
{

        printf("Enter the rotation angle\n");
        scanf("%f", &theta);
        theta=(3.14/180)*theta;
        glutInit(&argc,argv);
        glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
        glutInitWindowSize(500,500);
        glutCreateWindow("house rotation");
        glutDisplayFunc(display);
        myinit();
        glutMainLoop();
}
```