Q1—----------

The calculations available for computer graphics can be performed only at origin. It is a case of composite transformation which means this can be performed when more than one transformation is performed.
The following composite transformation matrix would be performed as follows.

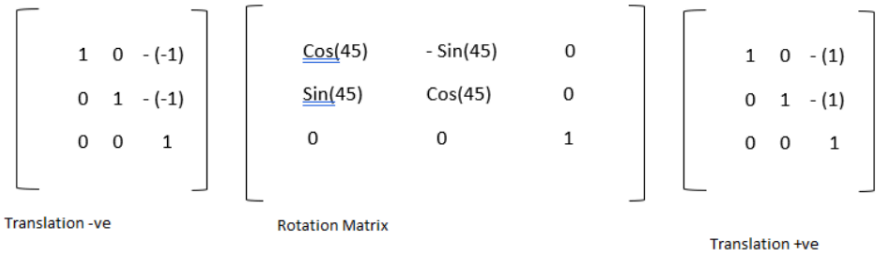
a) First bring the Point P(-1,-1) to the origin => which means translation towards origin towards origin will be negative translation. So tx and ty values would be negative.
So tx =-(-1)=1 and ty =—(-1) = 1.

b) Perform the rotation of 45 degrees.

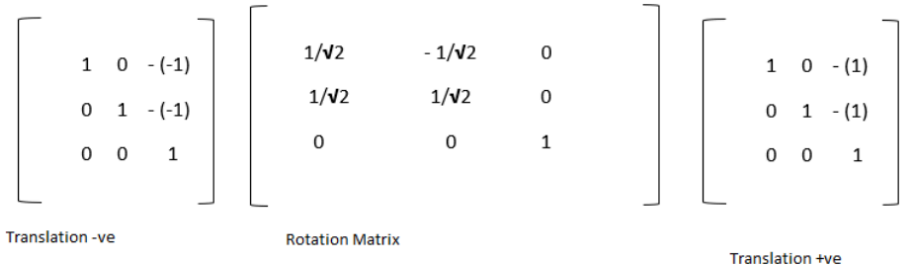
c) Send the point P(-1,-1) back => which mean translation away from origin => Away from origin would be positive translation therefore tx and ty will be positive.
So tx =-1 and ty = -1.

d) The most important point: the composite matrix should be written from right to left , So the composite rotation matrix would be :
T(x) * R(Theta) * T(-x) and NOT T(-x) * R(Theta) * T(x)

The composite rotation matrix would be = Positive translation * Rotation (45degree) * Negative translation.



Substitute of values of translation and rotation angle.



Multiply the resultant rotation matrix with the triangle matrix.

1/√2	- 1/√2	-1
1/√2	1/√2	(2/√2) - 1
0	0	1

Resultant Component Matrix

A	B	C
0	1	5
0	1	2
1	1	1

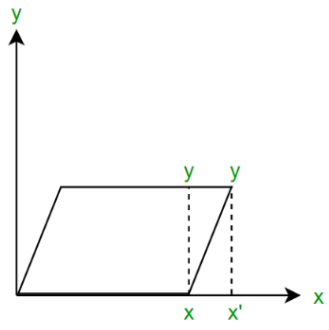
Triangle Matrix

The final resultant matrix will be as follows.

A'	B'	C'
-1	-1	(3/√2)-1
(2/√2)-1	(4/√2)-1	(9/√2)-1
1	1	1

Q2----------

Solution : Shearing deals with changing the shape and size of the 2D object along x-axis and y-axis. It is similar to sliding the layers in one direction to change the shape of the 2D object.It is an ideal technique to change the shape of an existing object in a two dimensional plane. In a two dimensional plane, the object size can be changed along X direction as well as Y direction.



$$x' = x + Sh_x * y; y' = y$$

Q3----------

a)

Double buffering provides two complete color buffers for use in drawing. One buffer is displayed while the other buffer is being drawn into. When the drawing is complete, the two buffers are swapped so that the one that was being viewed is now used for drawing.

The OpenGL program's output flickers when run.

b)

a) Enhanced Visuals and Animations: Double buffering is frequently used in graphics rendering to provide smooth and authentic animations.
b) Double buffering avoids flickering and enhances the visual experience in applications like games and multimedia by displaying one frame while buffering the next.

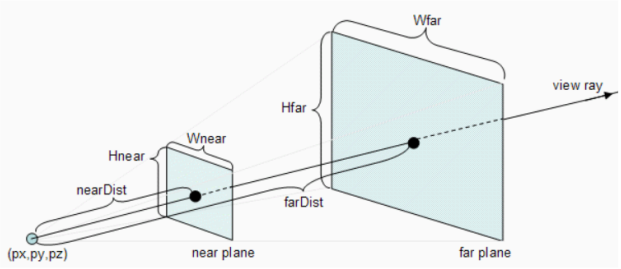
c)

In OpenGL, we can enable double buffering by calling the glutInitDisplayMode() function with the GLUT_DOUBLE flag before creating the window.

Q4—----------

a)

Input parameters of OpenGL 'gluPerspective' are : GLdouble fovy, GLdouble aspect, GLdouble zNear, GLdouble zFar.




```
Hnear = 2 * tan(fov / 2) * nearDist
```

```
Wnear = Hnear * ratio
```

```
Hfar = 2 * tan(fov / 2) * farDist
```

```
Wfar = Hfar * ratio
```

b)

Important parameters in gluLookAt are:

a) Eye Position (eyex, eyey, eyez)
b) Center Position (centerx, centery, centerz)
c) Up Vector (upx, upy, upz)

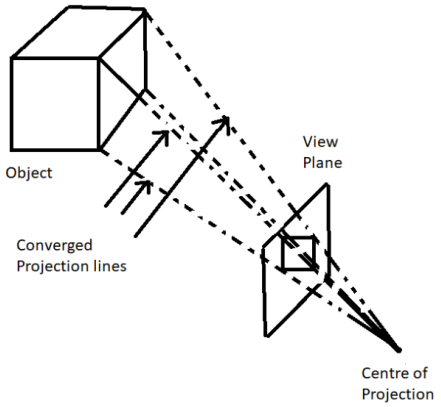
Q5—----------

a) Camera (eye) position (Ex,Ey, Ez) in world coordinate
b) Center of interest(coi)
c) Orientation like View up vector (Up_x, Up_y, Up_z)

Q6—----------

Vanishing point is a point where projection line intersects view plane. It is a point in image where a parallel line through center of projection intersects view plane. Classification of perspective projection is on basis of vanishing points.The classification is as follows :

a) One Point Perspective Projection – One point perspective projection occurs when any of principal axes intersects with projection plane or we can say when projection plane is perpendicular to principal axis.
b) Two Point Perspective Projection – Two point perspective projection occurs when projection plane intersects two of principal axis.
c) Three Point Perspective Projection – Three point perspective projection occurs when all three axis intersects with projection plane. There is no any principal axis which is parallel to projection plane.



Q7—----------

GL_LINES Use this primitive to draw unconnected line segments. OpenGL draws a line segment for each group of two vertices. If the application specifies n vertices, OpenGL renders n/2 line segments. If n is odd, OpenGL ignores the final vertex.

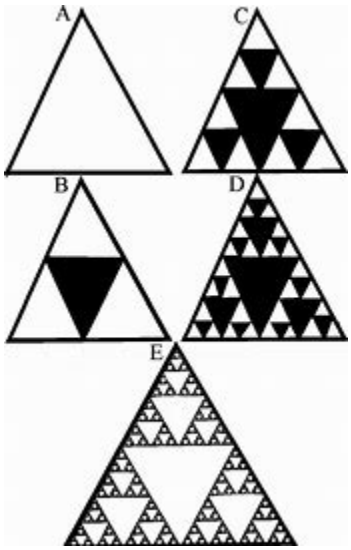
- **GL_LINE_STRIP Use this primitive to draw a sequence of connected line segments. OpenGL renders a line segment between the first and second vertices, between the second and third, between the third and fourth, and so on. If the application specifies n vertices, OpenGL renders n1 line segments.**
- **GL_LINE_LOOP Use this primitive to close a line strip. OpenGL renders this primitive like a GL_LINE_STRIP with the addition of a closing line segment between the final and first vertices.**

Q8—----------

To represent opacity.

Q9—----------

$$D = \frac{\log(N)}{\log(r)} = \frac{\log(3)}{\log(2)} = 1.585$$



Q10—----------

Aspect ratio is the ratio of the vertical points to horizontal points essential to produce equivalent length lines in both directions on the screen.

The aspect ratio will be 16:10.

Q11—----------

Transformation of object shape from one form to another is known as morphing.

Tweening is the process, which applies to animation objects defined by a sequence of points, and that change shape from frame to frame.

Q12—----------

A portion of a picture that is to be presented by a window is known as Window port.

The display method of the part selected or the design in which the selected element is viewed is called a viewport.

Q13—----------

It rotates an object in the counterclockwise direction.

Q14—----------

32 bytes