Assignment No. #AC33 DATE:

Title: 2-D transformation.

Problem Statement:
Write a C++ / Java program to draw 2-D object and perform following basic transformation.
i) Scaling ii) Translation iii) Rotation.

Learning objective:
To implement the 2-D transformation with translation, scaling and rotation of 2-D object.

Theory:
1) Transformation means changing some graphics into something else by applying rules.
2) Various types of transformation are:- i) Scaling ii) translation iii) Rotation

* Homogeneous coordinates:
1) To perform a sequence of transformation such as translation followed by rotating a scaling.
2) To shorten this process, we have to use 3x3 transformation matrix instead of 2-D transformation.
3) To convert do 2x2 into 3x3 matrix we need to add extra dummy coordinates.
4) In this way, we can represent the Co-ordinate point by 3 number instead of which is homogeneous coordinate system.

DATE:

* Translation:-
- A translation moves an object to a position on screen, you can translate a point in 2D by adding translation co-ordinate (t_x, t_y) in original coordinate x, y to x', y'.

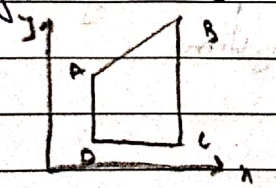
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} + \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

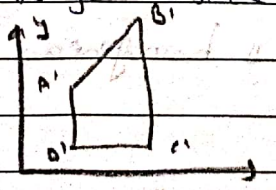
* Rotation:-
- In rotation, we rotate the object of particular angle θ from origin. From the figure we can see that point $P(x, y)$ is located at angle ϕ from horizontal x coordinate with distance r from origin.
Let us suppose you want to rotate it at angle θ. After rotating new location, you will get a new point $P'(x', y')$.

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

Rotation matrix Co-ordinate matrix

* Scaling:-
- To change the size of an object transformation is used. In scaling process, you either expand or compress the dimensions of object.
- Scaling can be achieved by multiplying the original coordinates of object with scaling factor to get desired results.





$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & S_z \end{bmatrix} * \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

DATE:

Pseudocode for multiplication of matrix :

```
for (i=0; i<n ; i++)
    for (j=0; j<n ; j++)
        transformed_M[i][j] := 0
        for (k=0; k<n ; k++)
            transformed_M[i][j] = transformed_M[i][j] +
                coordinate_M[i][k] * transformed_M[k][j];
        end for
    endfor
endfor
end Matrix multi
```

Conclusion:

We have learnt and implemented 2-D transformation with scaling, rotation and translation on 2-D object.