17-MAY-2021

ASSIGNMENT 1

QUESTIONS :-

1. Identify the applications of computer graphics and compare Rastar scan, Random scan displays.

2. Develop the working principle of Bresenham's line drawing algorithm for different slopes of a line.

3. Construct any 10 openGL output primitive functions.

4. Build the working of a CRT and Electron gun.

5. Interview the following terms with respect to computer graphics:
   a. Bitmap
   b. Aspect ratio
   c. Frame buffer
   d. Attribute

6. Design the diagram with different cartesian reference frames used in process of constructing and displaying a scene.

7. Develop an OpenGL program to create a house like structure using suitable openGL

functions.

8. Apply illustrations of basic 2-D geometric transformations used in computer graphics.

9. Construct with example any 2 algorithms used to identify the interior area of a polygon.

10. Design OpenGL polygon fill area functions.

ANSWERS: -

1. APPLICATIONS OF COMPUTER GRAPHICS:

Some of the known application of computer graphics are:

i> Graphs and charts.
ii> Computer Aided Design (CAD).
iii> Virtual Reality (VR).
iv> Data visualization.
v> Education and Training.
vi> Computer Art.
vii> Image processing.
viii> Graphical User Interface (GUI).
ix> Entertainment.

GRAPHS AND CHARTS:

* An early application of computer graphics was to plot simple data on a character printer.

* Graphs and charts are practically used to summarize statistical, mathematical data for reports, research, etc.

COMPUTER AIDED DESIGN (CAD):

* Computer Aided Design (CAD) or Computer Aided Drafted Designs (CADD) are used on a large scale in architectural and engineering fields.

* CAD can be used for producing designs of circuits or architectural platforms and also to create animations related to it.

VIRTUAL REALITY (VR):

* Virtual reality is one of the latest application that helps us analyze various scenarios in real which are actually virtual.

* If gloves are provided a person can experience feeling and picking objects in the virtuality. This can be used to see the risks in construction of a design and evaluate it.

EDUCATION AND TRAINING:

* Computer graphics can be used to provide education and training to individuals in many ways, one of which is simulation tools. An individual can learn to drive a car or fly a plane by running these simulations.

RASTER SCAN DISPLAY	RANDOM SCAN DISPLAY
Electron beam is swept across the entire screen from top to bottom, one row at a time.	Electron beam is directly pointed at the place on a screen where the picture is to be drawn.
Resolution is poor as Raster scan display produces zig-zag lines leading to discrete points sets.	Resolution is smoot as Random scan display produces straight lines in line path.
Picture is stored as pixels in refresh-buffer area.	Picture is stored as line drawing instructions in a display file.
Realistic display can be produced as pixels are used to store pictures.	Realistic display cannot be produced as line drawing instructions are used to store picture.
Screen points or pixels are used to draw an image.	Mathematical functions are used to draw an image.

2. To illustrate Bresenham's approach,

i> We consider scan conversion process for lines with positive slope less than 1.0.

ii> Pixel positions along a line path are determined by sampling at unit intervals of x. Starting from end point (x_0, y_0) of a given line, we step each column and plot pixel whose scan line y value is closest to line path.

iii> Consider the equation of straight line $y = mx + c$,

where $m = \frac{dy}{dx}$

ALGORITHM:

Case 1 : if $|m| < 1.0$

i> Input the two end points and store left input end points in (x_0, y_0).

ii> Plot the first point by setting colour frame buffer position (x_0, y_0).

iii> Calculate Δx, Δy, $2\Delta y$, $2\Delta y - 2\Delta x$ and obtain starting value for decision parameter as $P_0 = 2\Delta y - \Delta x$.

iv> At each x_K along the line, from $K=0$ performing following.

a> if $P_K < 0$ the next point is (x_{K+1}, y_K) and $P_{K+1} = P_K + 2\Delta y - 2\Delta x (y_{K+1} - y_K)$ else next point is (x_{K+1}, y_{K+1}).

v> Repeat step (iv) $\Delta x - 1$ times.

Case 2: if $m >= 1.0$

i> $p_0 = 2\Delta x - \Delta y$

a> if $p_k < 0$ then next point is (x_k, y_{k+1}), $p_{k+1} = p_k + 2\Delta x$

else next point is (x_{k+1}, y_{k+1}), $p_{k+1} = p_k + 2\Delta x - 2\Delta y$.

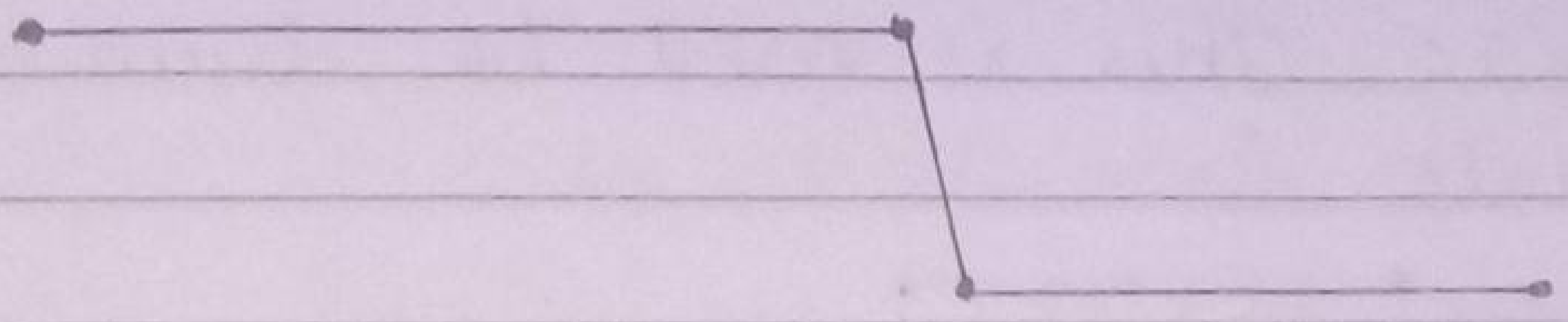
3. Few of the known OpenGL output primitive functions are:

i> GLPOINTS().
ii> GL_LINES().
iii> GL_LINES_STRIP().
iv> GL_LINE_LOOP().
v> GL_TRIANGLES().
vi> GL_TRIANGLE_STRIP().
vii> GL_QUADS().
viii> GL_QUAD_STRIP()
ix> GL_POLYGON().
x> GL_TRIANGLE_FAN().

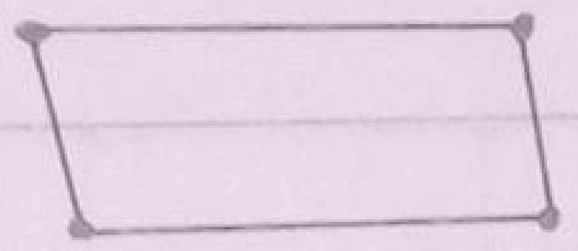
GL_POINTS(): This is used to draw mathematical points.

GL_LINES(): This is used to draw line segment.

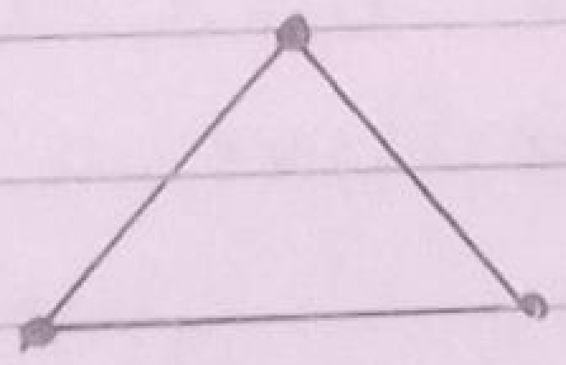
GL_LINE_STRIP(): Used to draw a sequence of line segments.



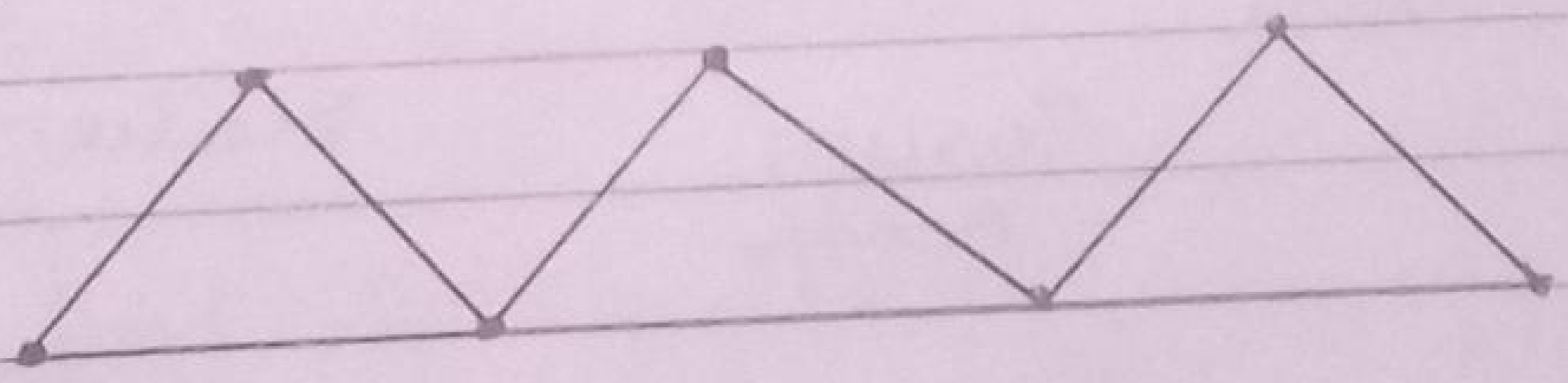
GL_LINE_LOOP(): Used to draw a sequence of line segment with closure, i.e., last point connected to first.



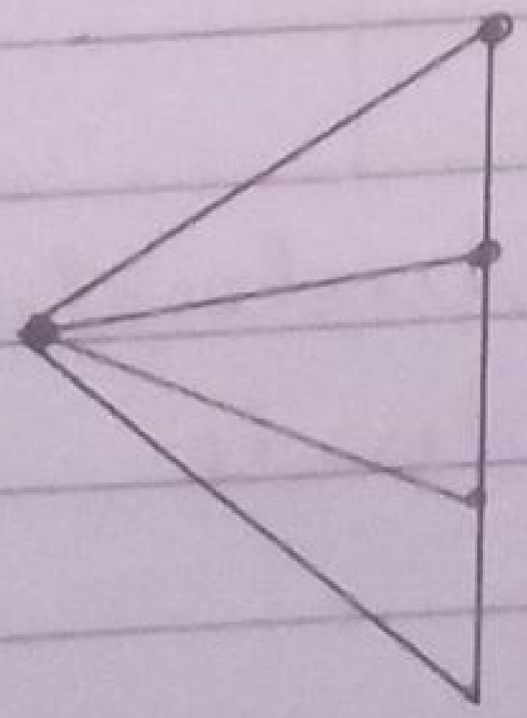
GL_TRIANGLES(): This is used to draw a triangle.



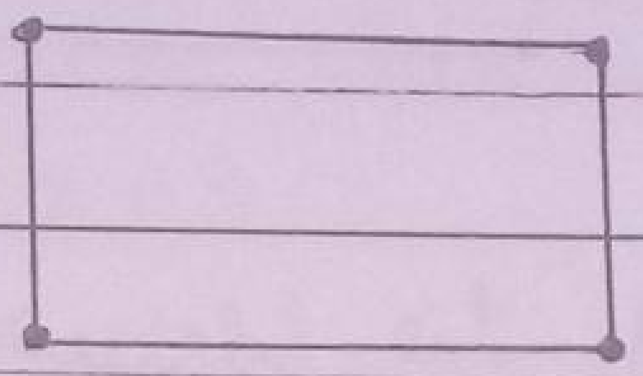
GL_TRIANGLE_STRIP(): Used to draw a sequence of triangles.



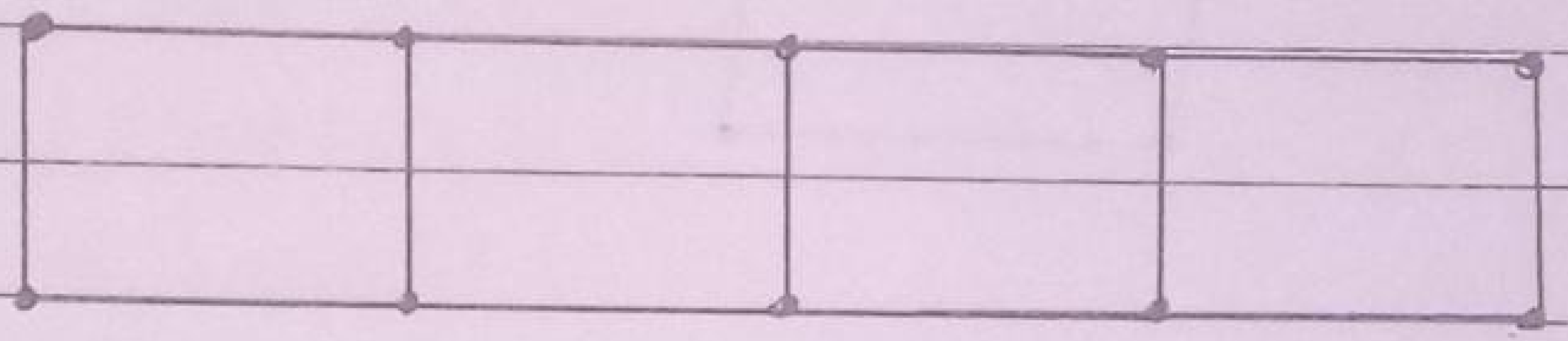
GL_TRIANGLE_FAN(): It is used to draw a fan of triangles sharing same edge and vertex.



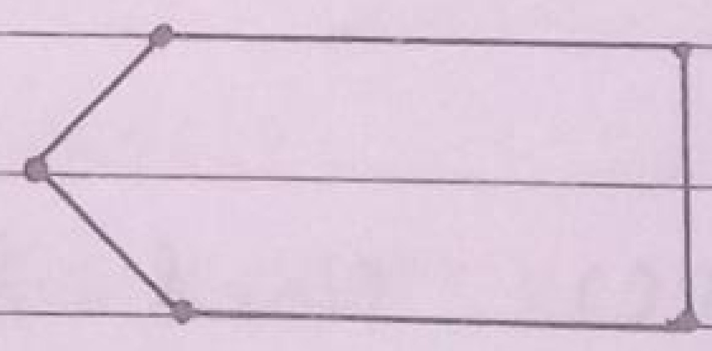
GL_QUADS(): This is used to draw convex quadrilaterals.



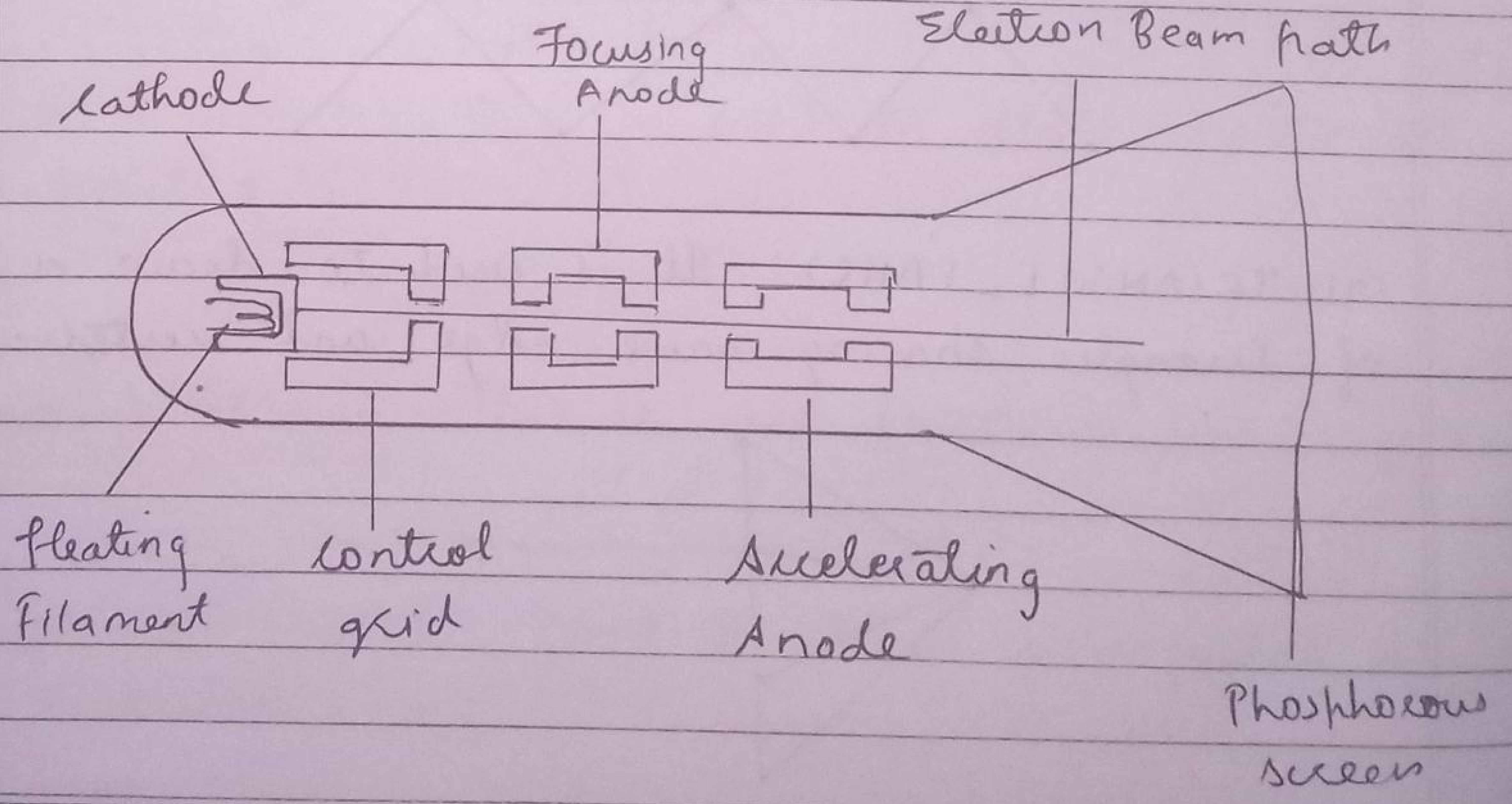
GL_QUAD_STRIP(): It is used to draw a sequence of quadrilatreals.



GL_POLYGON(): It is used to draw single filled convex n-side primitive.



4. ELECTRON GUN:



WORKING:

i> The primary components of an electron gun are heated cathode, control grid.

ii> The heat is supplied to cathode by a filament inside the cathode.

iii> The heat generated by filament causes the electrons on the surface of cathode to "boil off".

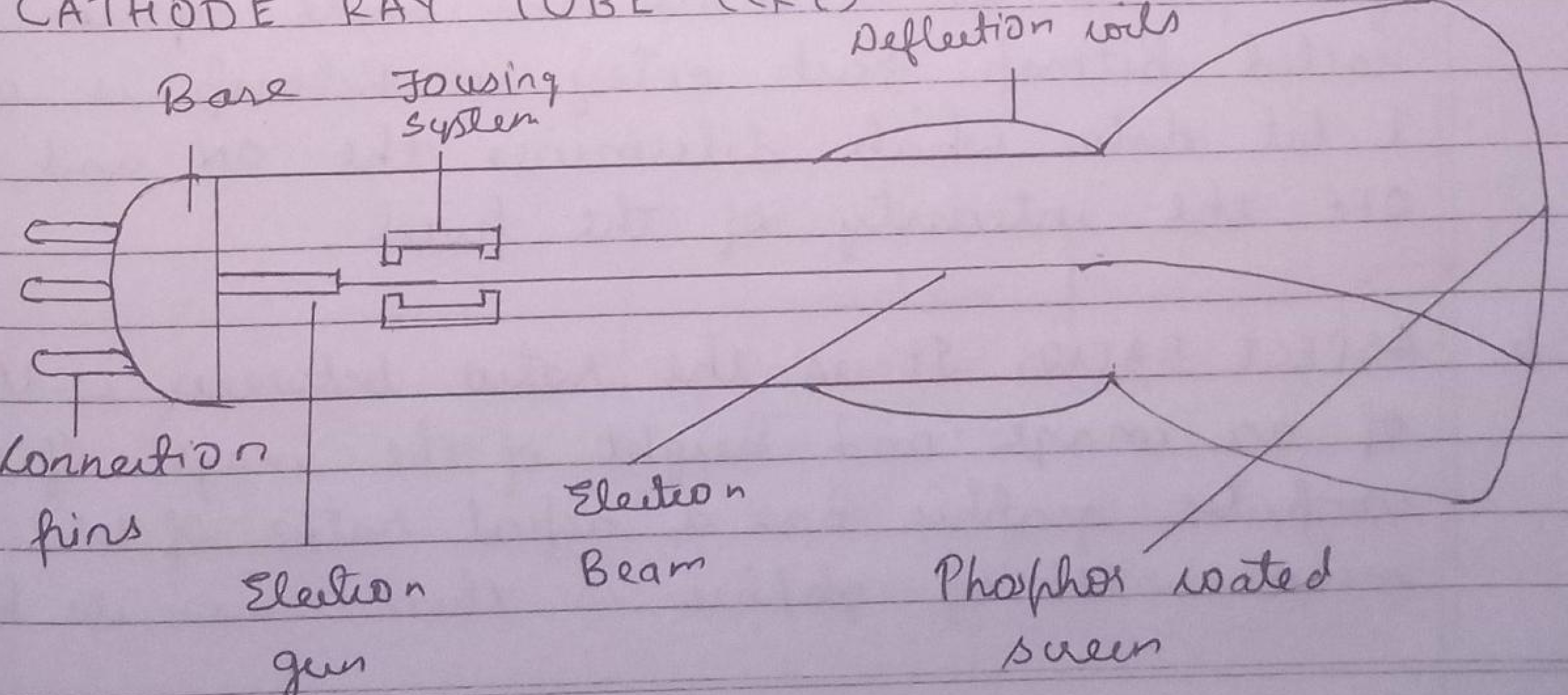
iv> These electrons travel through the beam towards phosphorous screen by a positive voltage.

v> The intensity of electron beam is controlled using the control grid.

vi> As the amount of electrons striking the phosphorous screen depends upon intensity, the brightness is controlled using control grid.

vii> The focusing anode forces electron beam to converge small cross section as it strikes phosphorous screen.

CATHODE RAY TUBE (CRT)



WORKING :

i> When a beam of electrons is emitted by electron gun, passed through focusing and deflection systems, the beam is directed to a specified position on the phosphorous screen.

ii> Now the phosphor produces a small spot of light at each position, contacted by electron beam and light emitted by phosphor fades very rapidly.

iii> One way to maintain the screen picture is to store picture as a charge distribution within CRT.

iv> The most common method is to redraw picture repeatedly by quickly directing electron beam back over same screen points.

v> The frequency at which pictures are redrawn on screen ~~are~~ is called refresh rate.

5.

a. BITMAP: The frame buffer storing the values of ~~pia~~ pixels on a black and white system is called Bitmap. Each entry in Bitmap is a 1-bit data which determines the ON and OFF the intensity of the pixel.

b. ASPECT RATIO: It is the ratio between width of an image and height of the image. If a computer graphic has a aspect ratio of 3:1, means width of graphic is three times its height.

21 - MAY - 2021

c. FRAME BUFFER: The memory in which picture definition is stored is called frame buffer. The frame buffer states intensity values for all the three screen points.

d. ATTRIBUTE: They are properties of output primitive. We can change size, position or orientation of an object.

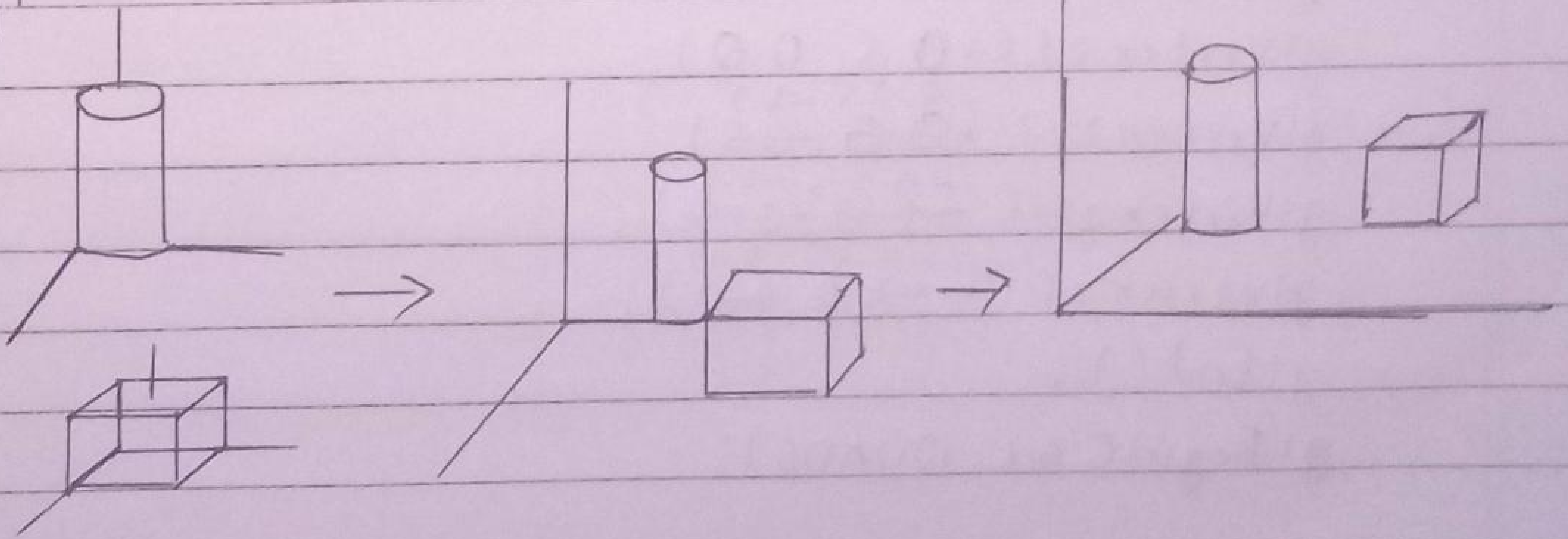
6. i> First we ~~design~~, define shapes of individual objects, such as trees, furnitures. These reference frames are called modelling co-ordinates or local co-ordinates.

ii> We place the objects into appropriate locations within a reference frame.

iii> The scene is then stored in normalized co-ordinates which range from 0 to 1 or -1 to 1.

iv> The co-ordinate system for displaying device are called device co-ordinates or screen co-ordinates.

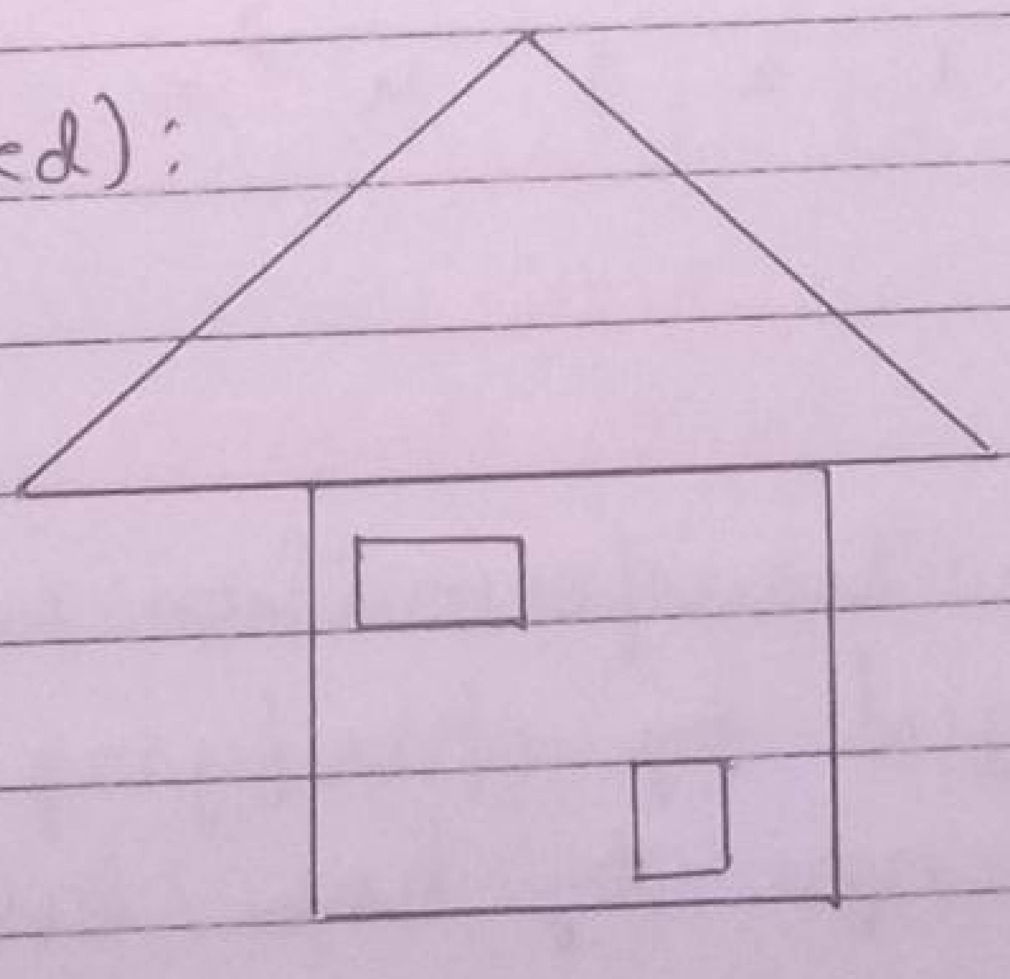
Eg:



7. /* OPENGL PROGRAM TO CREATE HOUSE LIKE STRUCTURE USING OPEN-GL FUNCTIONS */

```
#include <GL/glut.h>
#include <stdio.h>
void display () {
  glClear(GL_COLOR_BUFFER_BIT);
   glColor3f (1.0, 0.0,0.0);
   glPointSize (2.0);
   glBegin (GL_QUADS);
   glVertex2f (2.0, 1.0);
   glVertex2f (2.0,-1.5);
   glVertex2f (-2.0, -1.5);
   glVertex 2f (-2.0, 7.0);
    glEnd();
   glBegin (GL_TRIANGLES);
    glColor3f( 0.0, 1.0, 0.0);
    glVertex2i ( -3, 1);
    glVertex2i (3,1);
    glVertex2i (0,3);
    glEnd();
    glBegin( GL_QUADS );
    glColor3f(0.0, 1.0, 1.0);
    glVertex2f(+0.5, 0.0);
    glVertex2f( ~~+1.0~~ 0.5, ~~0.5~~ -1.5);
    glVertex2f( ~~-1.0~~ -0.5, ~~-0.5~~ -1.5);
    glVertex2f ( ~~-1.0~~ -0.5, ~~0.0~~ 0.0);
    glEnd ();
    glBegin( GL_QUADS);
    glColor3f(0.0, 1.0, 0.9);
    glVertex2f (-1.5, 0.5);
    glVertex2f(-1.0, 0.5);
    glVertex2f( -1.0, 0.0);
    glVertex2f  (-1.5, 0.0);
    glEnd();
    glFlush();
}
void myinit(){
   glClearColor(0.0, 0.0, 0.0, 1.0);
   glOrtho2D(-5.0, 5.0, -5.0, 5.0);
}
int main(int argc, char ** argv){
    glutInit (&argc, argv);
    glutInitDisplaymode(GLUT_RGR|GLUT_SINGLE);
    glutInitWindowSize(500, 500);
    glutInitWindowPosition (0,0);
    glutCreateWindow("SIMPLE HOUSE");
    myinit();
    glutDisplayfunc(display);
    glutMainloop();
}
```

OUTPUT(Expected):



Scanned by TapScanner

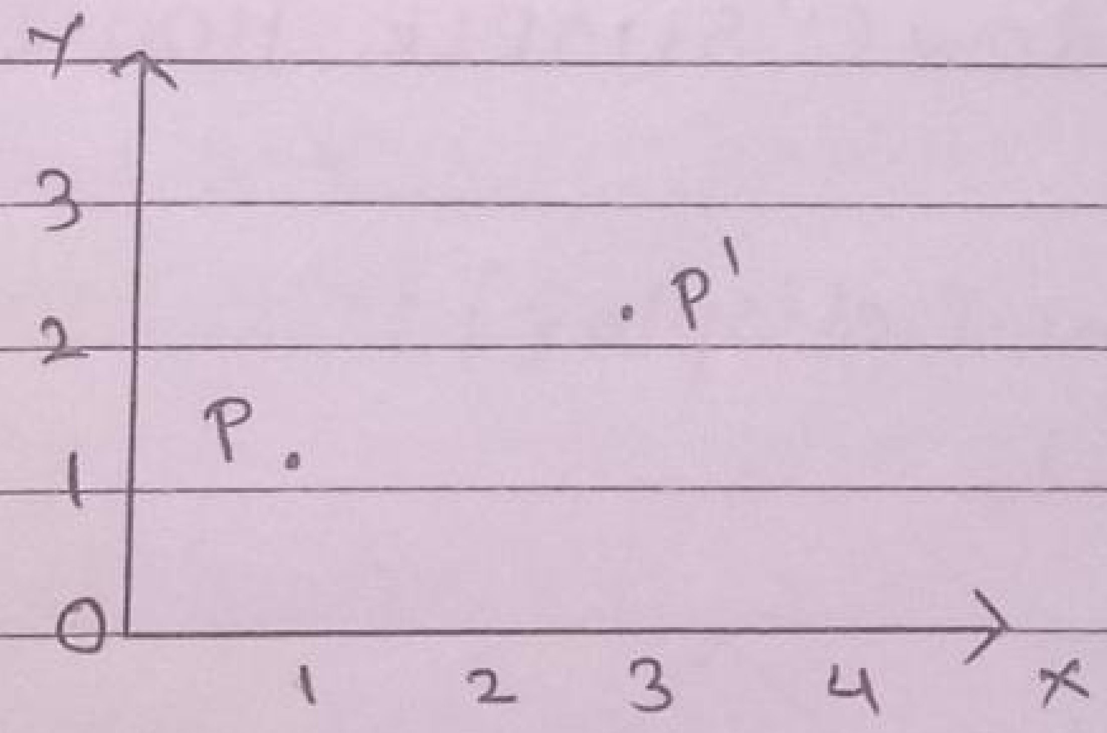
8. The basic 2D transformation's used in computer graphics are:
   i> 2D Translations
   ii> 2D Rotation
   iii> 2D Scaling

2D TRANSLATION:

a> In this transformation we are changing the position of a particular object by adding an offset value to its co-ordinates.

b> These values (offset) used to move object are called translation distances.

Eg: consider a point P(x, y) has to be moved from P(x, y) to P'(x', y') then using translation $x' = tx + x$, $y' = ty + y$ will shift the point P to P' where tx, ty are translation distances.

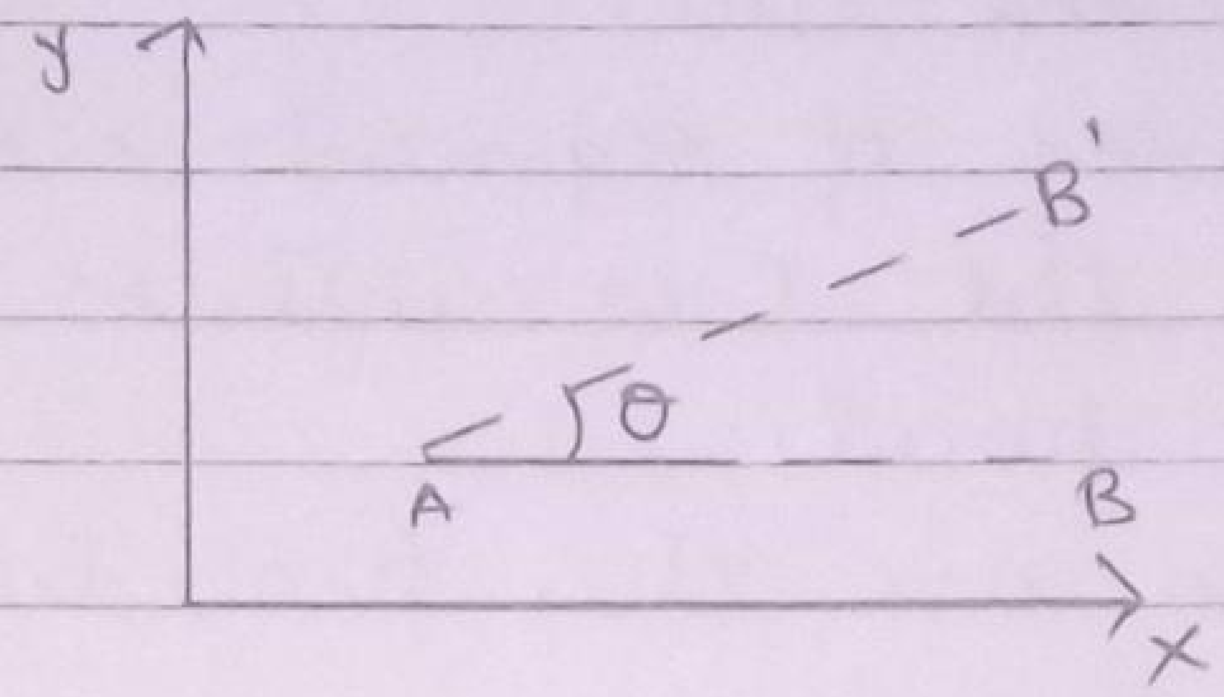


P(1,1)
P'(3,2)
⇒ tx = 2, ty = 1

2D ROTATION:

a> In this transformation we are rotating a particular object by specifying the rotation axis and the angle by how much it should be rotated.

Eg:
A line seiment AB to be rotated by angle θ to reach position AB'.



2D SCALING :

a> In this method we are changing the size of a particular object by multiplying the previous values with a factor.

b> If (x,y) are the co-ordinates the new size can be obtained by (S×x, S×y) where S is the scaling factor.

c> If we multiply co-ordinates by value less than 1 (i.e, S<1) then size decreases.

d> If we multiply co-ordinates by value greater than 1 (i.e, S>1) then size increases.

9. The algorithms used to identify interior area of a polygon are
i> Index Outside tests.
ii> Non-zero winding rule.

INDEX OUTSIDE TESTS

a> Draw a line from position P to a

distant point outside the co-ordinate extends of the closed polyline.

b> Then count the number of line segments crossing along the line.

c> If number of segments crossed by this line is odd, the P is interior point else P is exterior point.

d> We can use this procedure to fill the interior area.

e> This method is also called as odd-parity or even-parity rule.

NON-ZERO WINDING RULE:

a> In this method the number of times the boundary of an objects "winds" around a particular point in the counter-clockwise direction is counted.

b> This is called winding number.

c> The line we choose must not pass through any end point co-ordinates.

d> As we move along the line from position P to distant point, we count number of object line segments that cross reference line in each direction.

e> We add 1 to the winding number every time we intersect a segment.

f> If the winding number is non-zero, P is interior point or else it is exterior point.

21-MAY-2021

10. i> A ~~glvertex~~ glvertex function is used to input the co-ordinates for a single polygon vertex or a complete polygon is described with a list of vertex placed between a glBegin - glEnd pair.

ii> Thre are six different symbolic constants that we can use as argument in the glBegin function to desribe the polygon.

iii> In some implementation of openGL, following routine can be more different and efficient than generating a full retangle using glVertex.

iv> Suffix codes for glRect speify the co-ordinate datatype and whether co-ordinate are to be expressed as array elements.

v> These codes are

a> i → integers

b> s → short

c> f → float

d> d → double

e> v → vector

Eg:

```
glRecti(200,100,500,250);
int vertex1() = {200,100};
int vertex2() = {500,250};
glRectiv(Vertex1, Vertex2);
```

——X——X——X——