V2V EDTECH LLP

Online Coaching at an Affordable Price.

OUR SERVICES:

- Diploma in All Branches, All Subjects
- Degree in All Branches, All Subjects
- BSCIT / CS
- Professional Courses

☎ **+91 93260 50669**

v2vedtech.com

V2V EdTech LLP

v2vedtech

Important Instructions to examiners:

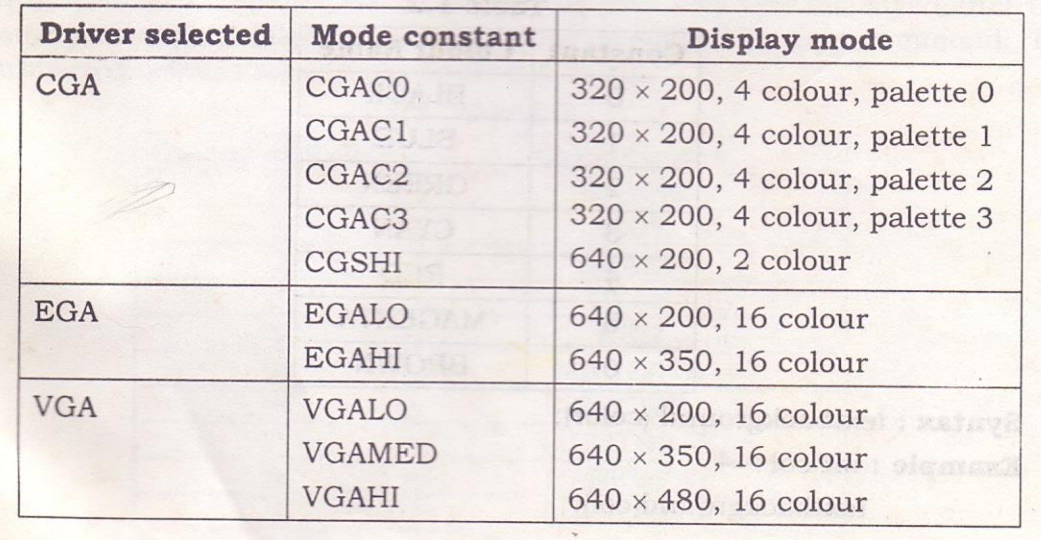
1) The answers should be examined by key words and not as word-to-word as given in the model answer scheme.
2) The model answer and the answer written by candidate may vary but the examiner may try to assess the understanding level of the candidate.
3) The language errors such as grammatical, spelling errors should not be given more Importance (Not applicable for subject English and Communication Skills.
4) While assessing figures, examiner may give credit for principal components indicated in the figure. The figures drawn by candidate and model answer may vary. The examiner may give credit for any equivalent figure drawn.
5) Credits may be given step wise for numerical problems. In some cases, the assumed constant values may vary and there may be some difference in the candidate's answers and model answer.
6) In case of some questions credit may be given by judgement on part of examiner of relevant answer based on candidate's understanding.
7) For programming language papers, credit may be given to any other program based on equivalent concept.

Q. No.	Sub Q. N.	Answer	Marking Scheme
1		**Attempt any FIVE of the following:**	**10 M**
	a	**Define:** **(i)Pixel** **(ii)Frame Buffer**	2 M
	Ans	• Pixel Pixel or Pel is defined as "the smallest addressable screen element". **OR** A pixel may be defined as the smallest size object or color spot that can be displayed and addressed on a monitor. • Frame Buffer The *frame buffer* is the video memory (RAM) that is used to hold or map the image displayed on the screen. **OR** A **framebuffer** (**frame buffer**, or sometimes **framestore**) is a portion of RAM containing a bitmap that drives a video display.	1 M each for correct definition

b **Give the characteristics of display adaptor.** 2 M

Ans The characteristics of common display adapters are given in Table. The present-day display adapter supports all the modes of the preceding display adapters

2M for any relevant characteristics



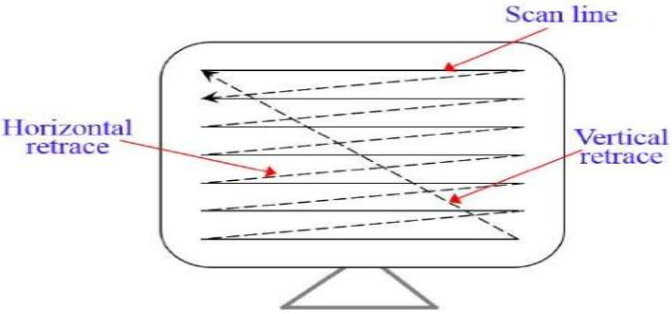
Driver selected	Mode constant	Display mode
CGA	CGAC0	320 × 200, 4 colour, palette 0
	CGAC1	320 × 200, 4 colour, palette 1
	CGAC2	320 × 200, 4 colour, palette 2
	CGAC3	320 × 200, 4 colour, palette 3
	CGSHI	640 × 200, 2 colour
EGA	EGALO	640 × 200, 16 colour
	EGAHI	640 × 350, 16 colour
VGA	VGALO	640 × 200, 16 colour
	VGAMED	640 × 350, 16 colour
	VGAHI	640 × 480, 16 colour

c **Explain Raster Scan** 2 M

Ans

- In Raster scan, the electron beam from electron gun is swept horizontally across the phosphor one row at time from top to bottom.
- The electron beam sweeps back and forth from left to right across the screen. The beam is on, while it moves from left to right. The beam is off, when it moves back from right to left. This phenomenon is called the ***horizontal retrace.***
- As soon as the beam reaches the bottom of the screen, it is turned off and is rapidly retraced back to the top to start again. This is called the ***vertical retrace***.
- Raster scan displays maintain the steady image on the screen by repeating scanning of the same image. This process is known as ***refreshing of screen***.

2 M for correct explanation



Raster Scan CRT

d **State two line drawing algorithms.** 2 M

Ans **Digital Differential Analyzer (DDA) Algorithm**

Digital Differential Analyzer algorithm generates a line from differential equations of line

1 M for each Algorithm

and hence the name DDA.

Bresenham's Algorithm

The Bresenham algorithm is another line drawing algorithm which uses integer calculations for drawing line.

e **List types of Polygon** 2 M

Ans Polygon can be of two types:-

- Convex polygon
- Concave polygon

1 M each

f **List various polygon filling algorithms** 2 M

Ans Various polygon filling algorithms are:

- Flood Fill Algorithm
- Boundary Fill Algorithm
- Scan Line Algorithm

1 M each, Any two

g **Give matrix representation for 2D scaling** 2 M

Ans Let us assume that the original co-ordinates are (X, Y), the scaling factors are (S_X, S_Y), and the produced co-ordinates are (X', Y'). This can be mathematically represented as shown below:

$$X' = X \cdot S_X \text{ and } Y' = Y \cdot S_Y$$

The scaling factor S_X, S_Y scales the object in X and Y direction respectively. The above equations can also be represented in matrix form as below:

$$\begin{bmatrix} X' \\ Y' \end{bmatrix} = \begin{bmatrix} X \\ Y \end{bmatrix} \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix}$$

2 M for proper Matrix

2 **Attempt any THREE of the following:** **12 M**

a **Differentiate between Random Scan and Raster Scan.** 4 M

Ans

Random Scan Display	Raster Scan Display
In vector scan display the beam is moved between the end points of the graphics primitives.	In raster scan display the beam is moved all over the screen one scan at a time, from top to bottom and then back to top.
Vector display flickers when the number of primitives in the buffer becomes too large.	In raster display, the refresh process is independent of the complexity of the image.

Any four points: 1 mark each

Scan conversion is not required.	Graphics primitives are specified in terms of their endpoints and must be scan converted into their corresponding pixels in the frame buffer.
Scan conversion hardware is not required.	Because each primitive must be scan converted real time dynamics is far more computational and requires separate scan conversion hardware.
Vector display draws continuous and smooth lines.	Raster display can display mathematically smooth lines, polygons and boundaries of curves primitives only by approximating them with pixels on the raster grid.
Mathematical functions are used to draw an image.	Screen points/pixels are used to draw an image.
It does not user interlacing.	It uses interlacing.
Editing is easy.	Editing is difficult.
Cost is more	Cost is low
Vector display only draws lines and characters.	Raster display has ability to display areas filled with solid colors or patterns.
Resolution is good because this system produces smooth lines drawings because CRT beam directly follows the line path.	Resolution is poor because raster system in contrast produces zigzag lines that are plotted as discrete point sets.
Picture definition is stored as a set of line drawing instructions in a display file.	Picture definition is stored as a set of intensity values for all screen points, called pixels in a refresh buffer area.
They are more suited to line drawing application e.g. CRO and pen plotter.	They are more suited to geometric area drawing applications e.g. monitors, TV
It uses beam-penetration method.	It uses shadow-mask method

b **Explain and write steps for DDA line drawing algorithm.** 4 M

Ans

Explanation 2M, Algorithm 2M

- This algorithm generates a line from differential equations of line and hence the name DDA.
- DDA algorithm is an incremental scan conversion method.
- A DDA is hardware or software used for linear interpolation of variables over an interval between start and end point.
- DDAs are used for rasterization of lines, triangles and polygons.
- DDA method is referred by this name because this method is very similar to the numerical differential equations. The DDA is a mechanical device that solves differential equations by numerical methods.

Algorithm:

Steps 1: Read the end points of line (x1,y1) and (x2,y2).

```
Steps 2:  $\Delta x$ = abs ($x_2 - x_1$) and
          $\Delta y$ = abs ($y_2 - y_1$)
Step 3:   if $\Delta x \geq \Delta y$ then
              length = $\Delta x$
          else
              length = $\Delta y$
          end if
Step 4: $\Delta x = (x_2 - x_1)$/length
Step 5: $\Delta y = (y_2 - y_1)$/length
Step 6: x=  $x_1$ + 0.5 * sign ($\Delta x$)
        y=  $y_1$ + 0.5 * sign ($\Delta y$)
Step 7: i = 1
            while (i ≤ length)
            {
              plot (integer (x), integer (y))
              x = x + $\Delta x$
              y = y + $\Delta y$
              i = i + 1
            }
Step 8: End
```

c **List out basic transformation techniques. Explain scaling transformation with respect to 2D.** 4 M

Ans

Basic transformations techniques are:

- Translation
- Scaling
- Rotation

Listing 1M, Explanation 3M

Scaling Transformation

- Scaling means to change the size of object. This change can either be positive or negative.
- To change the size of an object, scaling transformation is used. In the scaling process, you either expand or compress the dimensions of the object.
- Scaling can be achieved by multiplying the original co-ordinates of the object with the scaling factor to get the desired result.
- Let us assume that the original co-ordinates are (X, Y), the scaling factors are (S_X, S_Y), and the produced co-ordinates are (X', Y'). This can be mathematically represented as shown below:

$X' = X \cdot S_X$ and $Y' = Y \cdot S_Y$

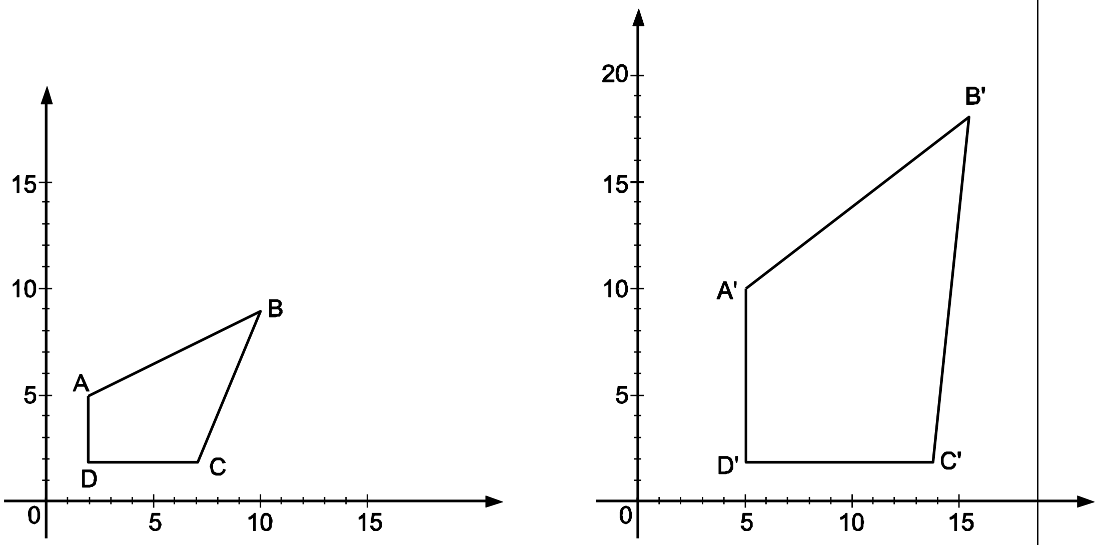
- The scaling factor S_X, S_Y scales the object in X and Y direction respectively. The above equations can also be represented in matrix form as below:

$$\begin{bmatrix}X'\\Y'\end{bmatrix} = \begin{bmatrix}X\\Y\end{bmatrix}\begin{bmatrix}S_x & 0\\0 & S_y\end{bmatrix}$$

OR

$$P'=P \cdot S$$

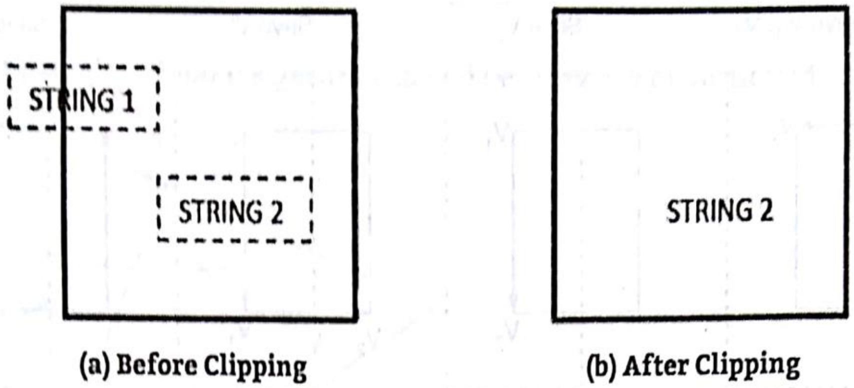
Where, S is the scaling matrix.



(a) Before Scaling **(b) After Scaling**

- If we provide values less than 1 to the scaling factor S, then we can reduce the size of the object. If we provide values greater than 1, then we can increase the size of the object.

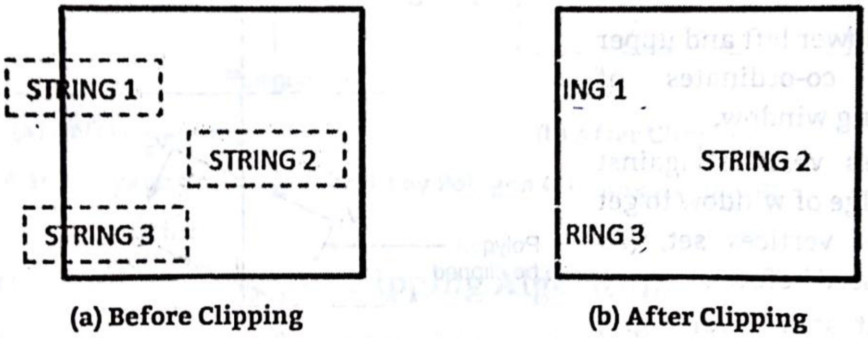
	d	**Explain differ types of Text clipping in brief.**	4 M
	Ans	Many techniques are used to provide text clipping in a computer graphics. It depends on the methods used to generate characters and the requirements of a particular application. There are three methods for text clipping which are listed below − 1) All or none string clipping 2) All or none character clipping 3) Text clipping The following figure shows all or none string clipping −	Explanation of 3 methods with diagrams 4 marks



(a) Before Clipping (b) After Clipping

In all or none string clipping method, either we keep the entire string or we reject entire string based on the clipping window. As shown in the above figure, Hello2 is entirely inside the clipping window so we keep it and Hello1 being only partially inside the window, we reject.

The following figure shows all or none character clipping –



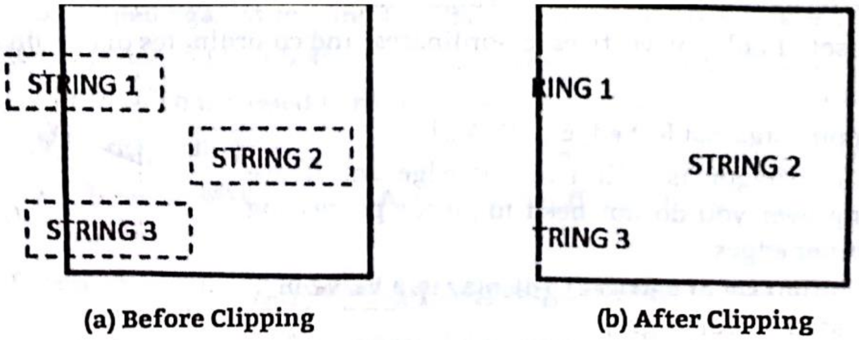
(a) Before Clipping (b) After Clipping

This clipping method is based on characters rather than entire string. In this method if the string is entirely inside the clipping window, then we keep it. If it is partially outside the window, then −

You reject only the portion of the string being outside

If the character is on the boundary of the clipping window, then we discard that entire character and keep the rest string.

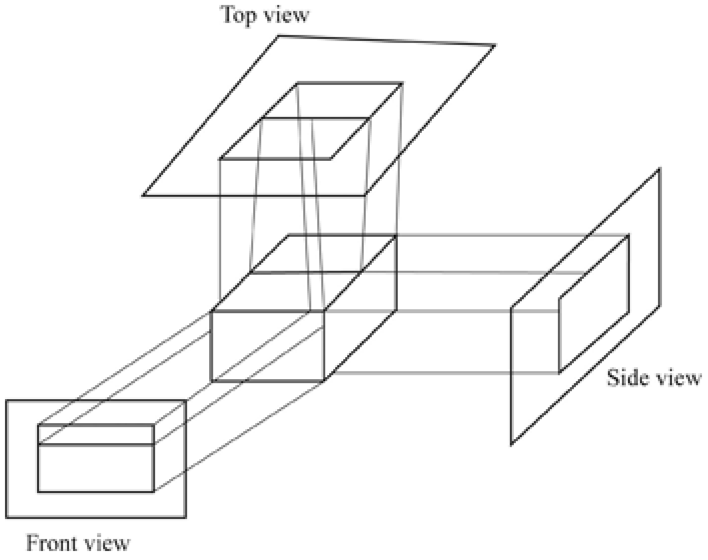
The following figure shows text clipping –



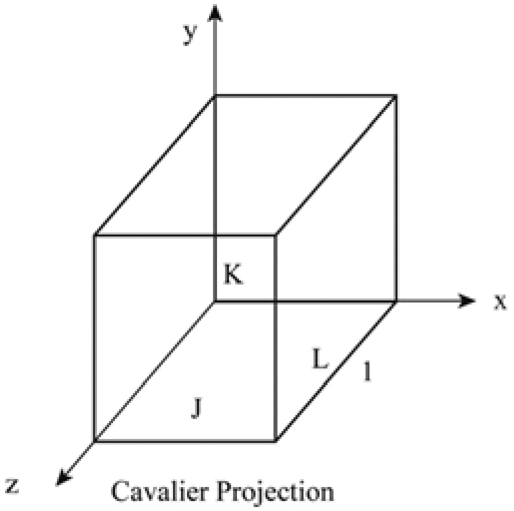
(a) Before Clipping (b) After Clipping

This clipping method is based on characters rather than the entire string. In this method if the string is entirely inside the clipping window, then we keep it. If it is partially outside the window, then you reject only the portion of string being

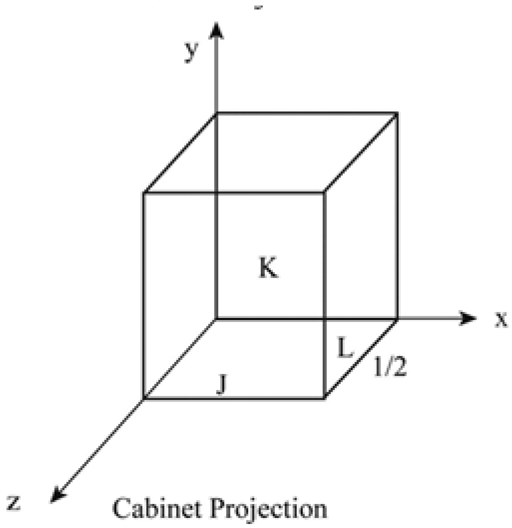
		outside. If the character is on the boundary of the clipping window, then we discard only that portion of character that is outside of the clipping window.	
3		**Attempt any THREE of the following:**	**12 M**
	a	**Explain stroke method and Bitmap method with example.**	4M
	Ans	**1)STROKE METHOD** • Stroke method is based on natural method of text written by human being. In this method graph is drawing in the form of line by line. • Line drawing algorithm DDA follows this method for line drawing. • This method uses small line segments to generate a character. The small series of line segments are drawn like a stroke of pen to form a character. • We can build our own stroke method character generator by calls to the line drawing algorithm. Here it is necessary to decide which line segments are needed for each character and then drawing these segments using line drawing algorithm. **2)BITMAP METHOD** • Bitmap method is a called dot-matrix method as the name suggests this method use array of bits for generating a character. These dots are the points for array whose size is fixed. • In bitmatrix method when the dots is stored in the form of array the value 1 in array represent the characters i.e. where the dots appear we represent that position with numerical value 1 and the value where dots are not present is represented by 0 in array. • It is also called dot matrix because in this method characters are represented by an array of dots in the matrix form. It is a two dimensional array having columns and rows. • A 5x7 array is commonly used to represent characters. However 7x9 and 9x13 arrays are also used. Higher resolution devices such as inkjet printer or laser printer may use character arrays that are over 100x100.	Stroke Method 2 Marks; Bitmap Method 2 Marks
	b	**Explain types of Parallel Projection with example.**	4M
	Ans	• Orthographic projection – the projection direction is a normal one to the plane and it is categorized as ○ Top projection ○ Front projection ○ Side projection	Orthographic projection 2 marks; Oblique projection 2 Marks



- Oblique projection – the projection direction is not a normal one to the plane; it gives a better view and it is categorized as
  - ○ Cavalier projection
  - ○ Cabinet projection



Cavalier Projection



Cabinet Projection

	c	**Write down Cohen-Sutherland Line clipping algorithm.**	4M
	Ans	Step 1: Scan end points for the line P1(x1, y1) and P2(x2, y2) Step 2: Scan corners for the window as (Wx1, Wy1) and (Wx2, Wy2) Step 3: Assign the region codes for endpoints P1 and P2 by Bit 1 - if (x < Wx1) Bit 2 - if (x < Wx2) Bit 3 - if (x < Wy2)	Correct algorithm 4 Marks

Bit 4 - if (x < Wy1)

Step 4: Check for visibility of line P1, P2

- If region codes for both end points are zero then the line is visible, draw it and jump to step 9.
- If region codes for end points are not zero and the logical and operation of them is also not zero then the line is invisible, reject it and jump to step 9.
- If region codes for end points does not satisfies the condition in 4(i) and 4(ii) then line is partly visible.

Step 5: Determine the intersecting edge of the clipping window by inspecting the region codes for endpoints.

- If region codes for both the end points are non-zero, find intersection points P1 and P2 with boundary edges of clipping window with respect to point P1 and P2.
- If region code for any one end point is non zero then find intersection point P1 or P2 with the boundary edge of the clipping window with respect to it.

Step 6: Divide the line segments by considering intersection points.

Step 7: Reject the line segment if any of the end point of it appear outside the window.

Step 8: Draw the remaining line.

Step 9: Exit

d	**Explain Koch curve with diagram.**	4M

Ans

Koch Curve: - In Koch curve, begin at a line segment. Divide it into third and replace the center by the two adjacent sides of an equilateral triangle as shown below.

Description 3 Marks; Diagram 1 Mark

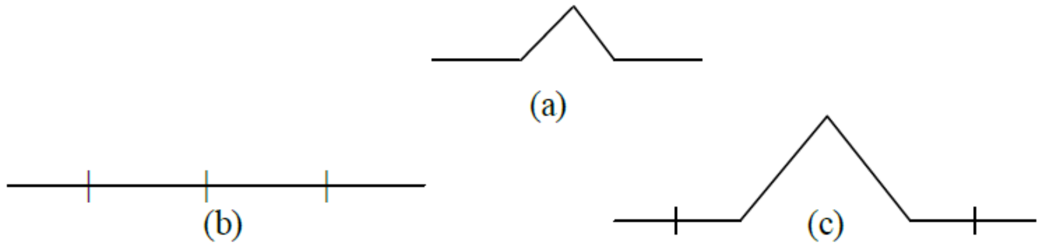


Fig 6.3 Replacement of Line Segment for Koch Curve

This will give the curve which starts and ends at same place as the original segment but is built of 4 equal length segments, with each 1/3rd of the original length. So the new curve has 4/3 the length of original segments. Repeat same process for each of the 4 segment which will give curve more wiggles and its length become 16/9 times the original. Suppose repeating the replacements indefinitely, since each repetition increases the length by a factor of 4/3, the length of the curve will be infinite but it is folded in lots of tiny wiggles.

4 **Attempt any THREE of the following:** **12 M**

a **Compare Bitmap Graphics and Vector based graphics.** 4 M

Ans

Bitmap Graphics	Vector Based Graphic
It is pixel based image	It is Mathematical based image
Images are resolution dependent.	Images are formula based / dependent.
These images are not easily scalable.	Easily scalable with the help of formula.
Poor quality of image as oppose to Vector based Graphics.	Better image quality as compare to Bitmap Graphics.
Size of image is high.	Size of image is low.

Any 4 Points of comparison; 1 Mark each

b **Consider line from (4, 4) to (12 9). Use Bresenham's algorithm to rasterize this line.** 4 M

Ans

x1 = 4 | y1 = 4 | & | x2 = 12 | y2 = 9	
Calculation	**Result**
dx = abs(x1 - x2)	8 = abs(4 - 12)
dy = abs(y1 - y2)	5 = abs(4 - 9)
p = 2 * (dy - dx)	-6 = 2 * (5 - 8)
ELSE	x = x1 | y = y1 | end = x2
	x = 4 | y = 4 | end = 12

STEP	while(x < end)	x = x + 1	if(p < 0) { p = p + 2 * dy } else{ p = p + 2 * (dy - dx) }	OUTPUT
1	5 < 12	5 = 4 + 1	IF 4 = -6 + 2 * 5	x = 5 | y = 4
2	6 < 12	6 = 5 + 1	ELSE -2 = 4 + 2 * (5 - 8)	x = 6 | y = 5
3	7 < 12	7 = 6 + 1	IF 8 = -2 + 2 * 5	x = 7 | y = 5

Any Suitable method can be consider

Correct steps and result: 4 Marks

4	8 < 12	8 = 7 + 1	ELSE 2 = 8 + 2 * (5 - 8)	x = 8 | y = 6
5	9 < 12	9 = 8 + 1	ELSE -4 = 2 + 2 * (5 - 8)	x = 9 | y = 7
6	10 < 12	10 = 9 + 1	IF 6 = -4 + 2 * 5	x = 10 | y = 7
7	11 < 12	11 = 10 + 1	ELSE 0 = 6 + 2 * (5 - 8)	x = 11 | y = 8
8	12 < 12	12 = 11 + 1	ELSE -6 = 0 + 2 * (5 - 8)	x = 12 | y = 9

c **Use Cohen-Sutherland algorithm to clip two lines PI (40, 15) -- P2 (75. 45) and P3 (70, 20) — P4 (100, 10) against a window A (50, 10), B (80, 10). C(80, 40) & D(50,40)** 4 M

Ans Solution :

Any suitable method can be consider

Computation for Line 1: 2 Marks; Computation for Line 2 : 2 Marks

Line 1 : P1 (40, 15) - P2 (75, 45) Wxi = 50 Wy2 = 40 Wx2 = 80 Wy2 = 10

Point	Endeode	ANDing	
P1	0001	0000	(Partially visible)
P2	0000		

$y_1 = m(x_L – x) + y = \frac{6}{7}(50-40)+15$ $m = \frac{45-15}{75-40}$

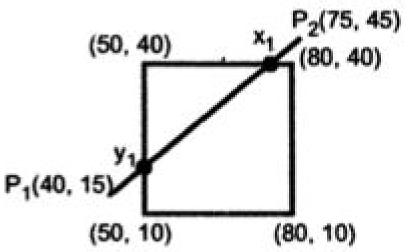
$= 23.57$

$x_1 = \frac{1}{m}(y_T – y) + x = \frac{7}{6}(40-50)+40 = 69.16$

$y_2 = m(x_R – x) + y = \frac{6}{7}(80-40)+15 = 49.28$

$x_2 = \frac{1}{m}(y_B – y) + x = \frac{7}{6}(10-15)+40 = 34.16$

Hence:



Line 2 : P3 (70,20) – P4 (100,10) Wxi = 50 Wy2 = 40 Wx2 = 80 Wy2 = 10

Point	Endeode	ANDing	
P3 0000	0000		(Partially visible)
P4 0010			

Slope m\`= $\frac{10-20}{100-70} = \frac{-10}{30} = \frac{-1}{3}$

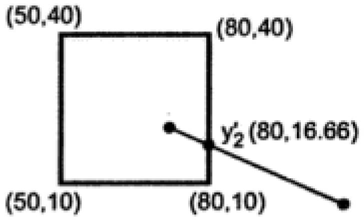
y\`$_1$ = m(x_L – x) + y = $\frac{-1}{3}$(50-70)+20 = 26.66

x\`$_1$ = $\frac{1}{m}$ (y_T – y) + x = - 3 (40-20)+70 = 10

y\`$_2$ = m(x_R – x) + y = $\frac{-1}{3}$(80-70)+20 = 16.66

x\`$_2$ = $\frac{1}{m}$ (y_B – y) + x = - 3 (10-20)+70 = 100

Hence:



d **Consider the square A (1, 0), B (0, 0), C (O, 1), D (1, I). Rotate the square ABCD by 45° anticlockwise about point A (1. 0).** 4 M

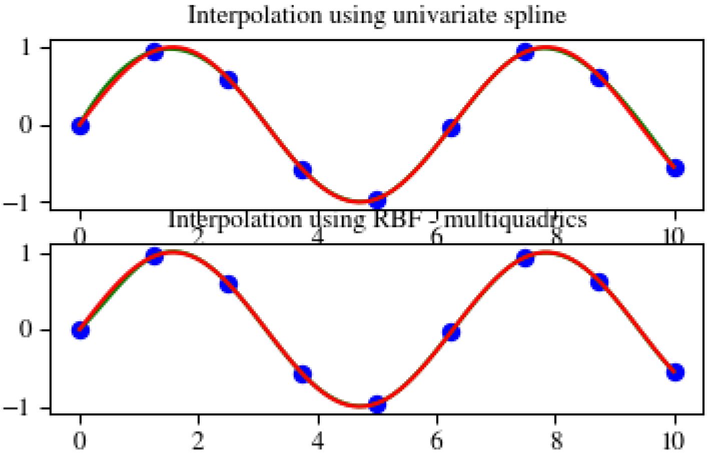
Ans

$$\begin{bmatrix} \cos\theta & \sin\theta\theta & 0 \\ -\sin\theta\theta & \cos\theta & 0 \\ -Xp\cos\theta + Yp\sin\theta + Xp & -Xp\sin\theta - Yp\cos\theta + Yp & 1 \end{bmatrix}$$

Matrix formation 2 Marks; Matrix calculation 2 Marks

Here, θ = 45°, X_p =1 Y_p = 0

$$[T_1.R.T_2] = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ \frac{-1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ \frac{1}{\sqrt{2}}+1 & -\frac{1}{\sqrt{2}} & 1 \end{bmatrix}$$

		$\begin{bmatrix} A' \\ B' \\ C' \\ D' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} & 0 \\ -1/\sqrt{2} & 1/\sqrt{2} & 0 \\ \frac{-1}{\sqrt{2}}+1 & -\frac{1}{\sqrt{2}} & 1 \end{bmatrix}$ $= \begin{bmatrix} 1 & 0 & 1 \\ \frac{-1}{\sqrt{2}}+1 & -\frac{1}{\sqrt{2}} & 1 \\ 1-\sqrt{2} & 0 & 1 \\ 1-\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 1 \end{bmatrix}$	
	e	**Explain curve generation using Interpolation technique.**	4 M
	Ans	Specify a spline curve by giving a set of coordinate positions, called control points, which indicates the general shape of the curve These, control points are then fitted with piecewise continuous parametric polynomial functions in one of two ways. When polynomial sections are fitted so that the curve passes through each control point, the resulting curve is said to interpolate the set of control points. On the other hand, when the polynomials are fitted to the general control-point path without necessarily passing through any control point, the resulting curve is said to approximate the set of control points interpolation curves are commonly used to digitize drawings or to specify animation paths. Approximation curves are primarily used as design tools to structure object surfaces an approximation spline surface credited for a design application. Straight lines connect the control-point positions above the surface. 	Description 2 Marks; Example/Diagram 2 Marks

5		**Attempt any two of the following:**	**12 M**
	a	**Rotate a triangle defined by A(0,0), B(6,0), & C(3,3) by 90^0 about origin in anti-clock wise direction**	6 M

Ans

The new position of point A (0, 0) will become A' (0,0)
The new position of point B (6,0) will become B' (0, 6)
The new position of point C (3, 3) will become C' (-3, 3)

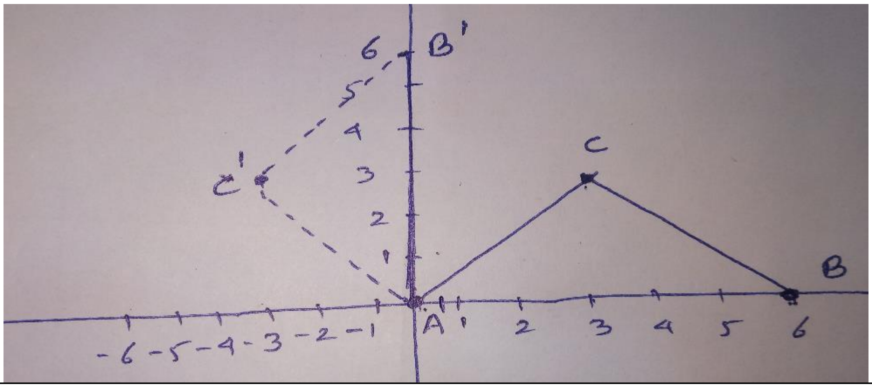
Matrix 2 Marks

Correct answer 4 marks

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

$$\begin{bmatrix} 0 & 0 & 1 \\ 6 & 0 & 1 \\ 3 & 3 & 1 \end{bmatrix} \quad x \quad \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

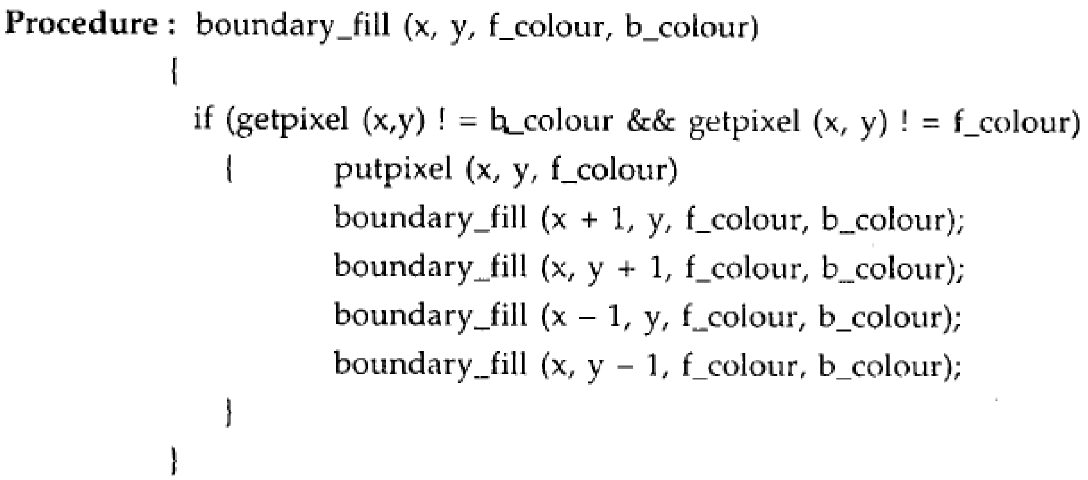
$$= \begin{bmatrix} 0 & 0 & 1 \\ 0 & 6 & 1 \\ -3 & 3 & 1 \end{bmatrix}$$



	b	**Explain boundary fill algorithm with pseudo code. Also mention its limitations if any.**	6 M

Ans

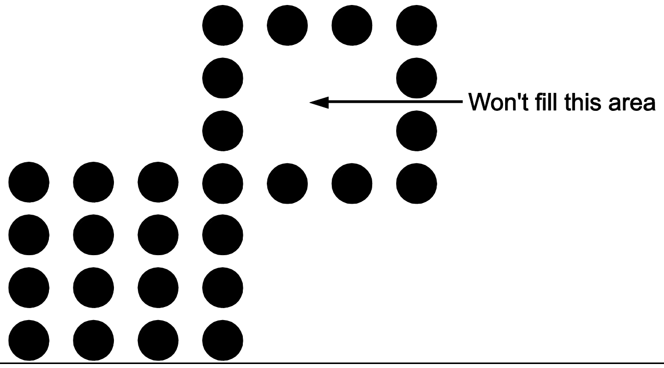
4m algorithm, 2m for limitations



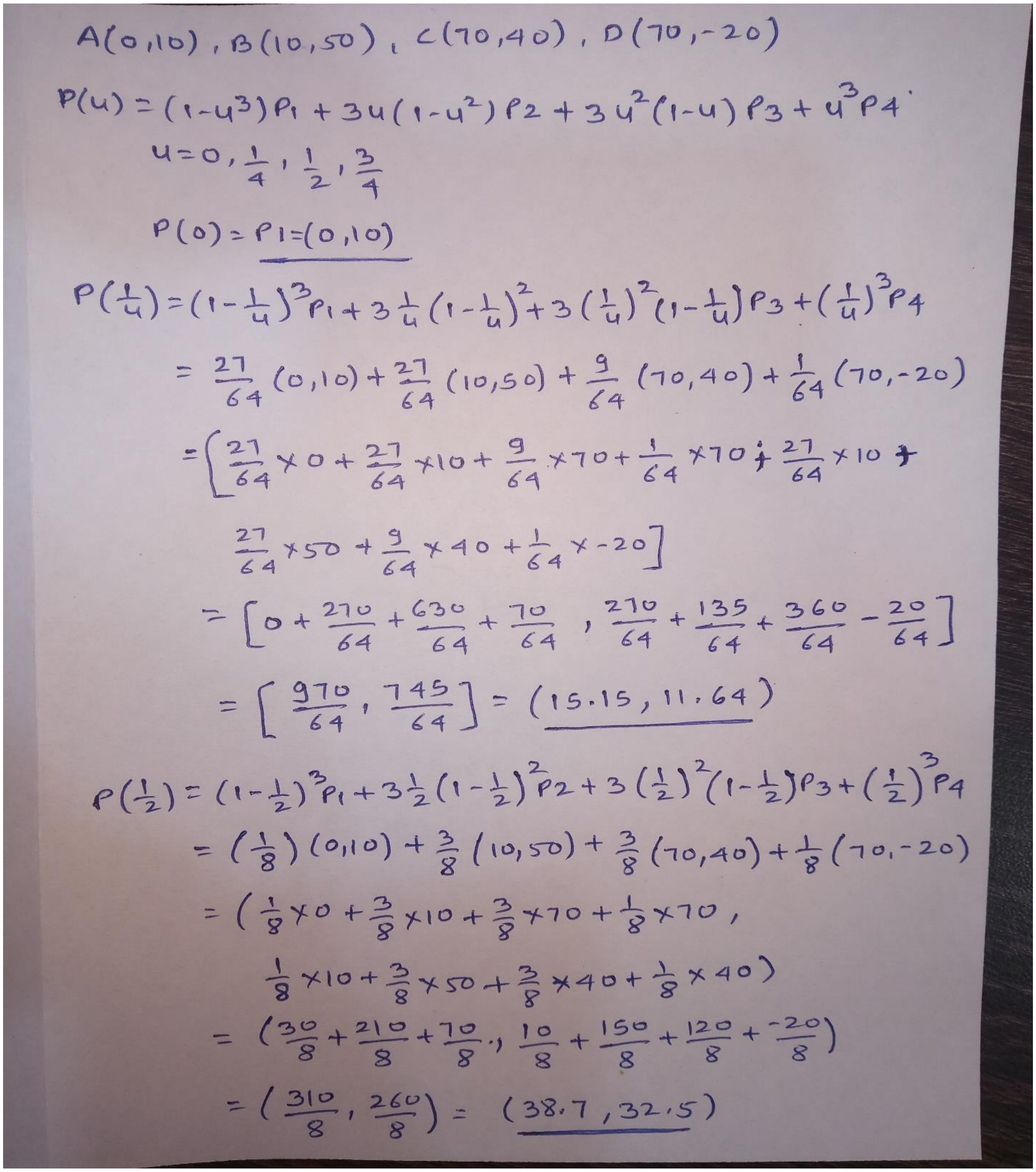
```
Procedure : boundary_fill (x, y, f_colour, b_colour)
        {
          if (getpixel (x,y) ! = b_colour && getpixel (x, y) ! = f_colour)
            {       putpixel (x, y, f_colour)
                    boundary_fill (x + 1, y, f_colour, b_colour);
                    boundary_fill (x, y + 1, f_colour, b_colour);
                    boundary_fill (x – 1, y, f_colour, b_colour);
                    boundary_fill (x, y – 1, f_colour, b_colour);
            }
        }
```

Limitations:

- There is a problem with this technique. Consider the case following, where we tried to fill the

		entire region. Here, the image is filled only partially. In such cases, 4-connected pixels technique cannot be used.  	
	c	**obtain the curve parameters for drawing a smooth Bezier curve for the following points A(0,10), B(10,50), C(70,40) &D(70,-20)**	6 M

Ans



A(0,10), B(10,50), C(70,40), D(70,-20)

$P(u) = (1-u^3)P_1 + 3u(1-u^2)P_2 + 3u^2(1-u)P_3 + u^3P_4$

$u = 0, \frac{1}{4}, \frac{1}{2}, \frac{3}{4}$

$P(0) = P_1 = (0,10)$

$P(\frac{1}{4}) = (1-\frac{1}{4})^3P_1 + 3\frac{1}{4}(1-\frac{1}{4})^2 + 3(\frac{1}{4})^2(1-\frac{1}{4})P_3 + (\frac{1}{4})^3P_4$

$= \frac{27}{64}(0,10) + \frac{27}{64}(10,50) + \frac{9}{64}(70,40) + \frac{1}{64}(70,-20)$

$= \left[\frac{27}{64}\times 0 + \frac{27}{64}\times 10 + \frac{9}{64}\times 70 + \frac{1}{64}\times 70 ; \frac{27}{64}\times 10 + \frac{27}{64}\times 50 + \frac{9}{64}\times 40 + \frac{1}{64}\times -20\right]$

$= \left[0 + \frac{270}{64} + \frac{630}{64} + \frac{70}{64}, \frac{270}{64} + \frac{135}{64} + \frac{360}{64} - \frac{20}{64}\right]$

$= \left[\frac{970}{64}, \frac{745}{64}\right] = (15.15, 11.64)$

$P(\frac{1}{2}) = (1-\frac{1}{2})^3P_1 + 3\frac{1}{2}(1-\frac{1}{2})^2P_2 + 3(\frac{1}{2})^2(1-\frac{1}{2})P_3 + (\frac{1}{2})^3P_4$

$= (\frac{1}{8})(0,10) + \frac{3}{8}(10,50) + \frac{3}{8}(70,40) + \frac{1}{8}(70,-20)$

$= (\frac{1}{8}\times 0 + \frac{3}{8}\times 10 + \frac{3}{8}\times 70 + \frac{1}{8}\times 70,$
$\frac{1}{8}\times 10 + \frac{3}{8}\times 50 + \frac{3}{8}\times 40 + \frac{1}{8}\times 40)$

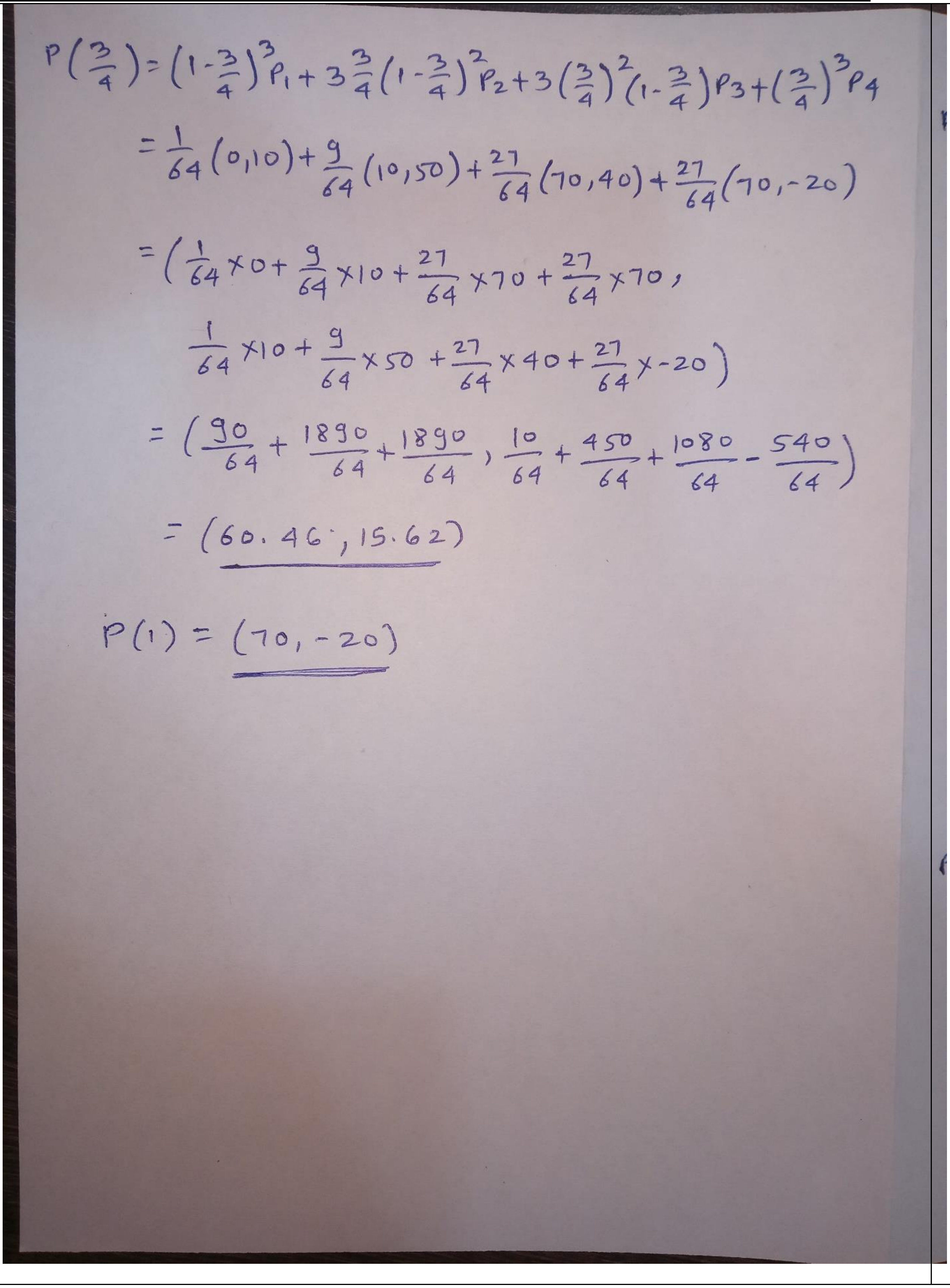
$= (\frac{30}{8} + \frac{210}{8} + \frac{70}{8}, \frac{10}{8} + \frac{150}{8} + \frac{120}{8} + \frac{-20}{8})$

$= (\frac{310}{8}, \frac{260}{8}) = (38.7, 32.5)$

Any correct method can be consider.

Calculation 3 Marks

Diagram 3Marks



$$P\left(\frac{3}{4}\right) = \left(1-\frac{3}{4}\right)^3 P_1 + 3\frac{3}{4}\left(1-\frac{3}{4}\right)^2 P_2 + 3\left(\frac{3}{4}\right)^2\left(1-\frac{3}{4}\right) P_3 + \left(\frac{3}{4}\right)^3 P_4$$

$$= \frac{1}{64}(0,10) + \frac{9}{64}(10,50) + \frac{27}{64}(70,40) + \frac{27}{64}(70,-20)$$

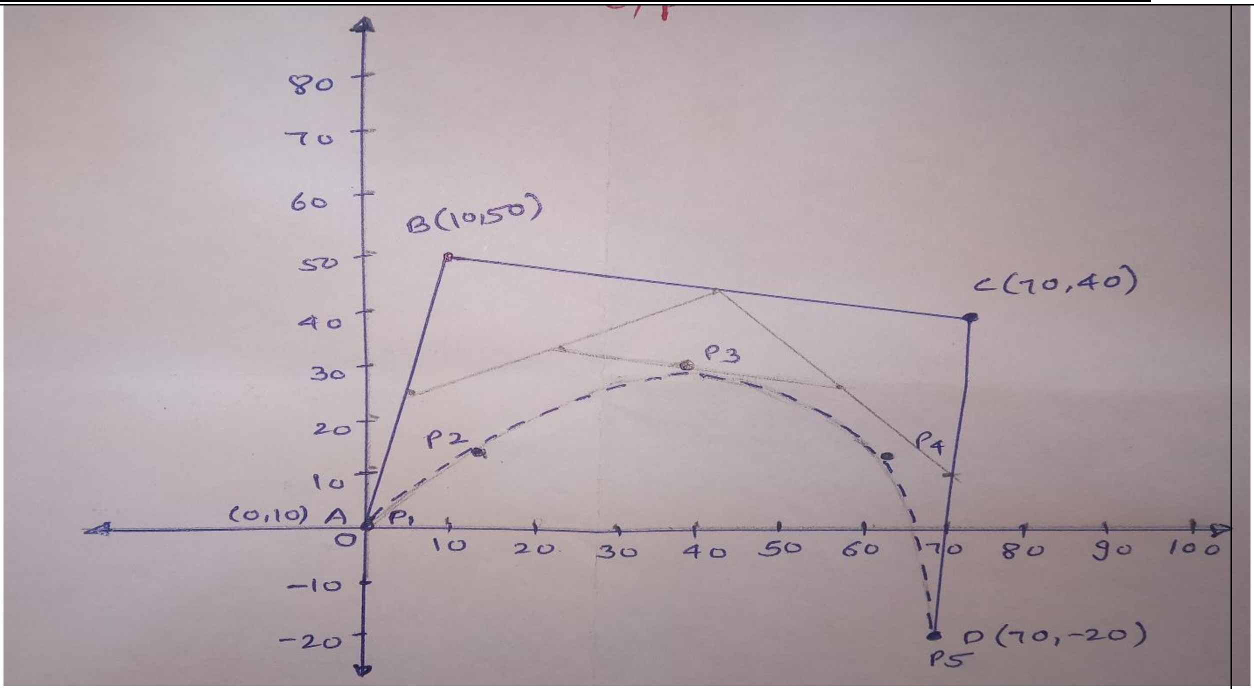
$$= \left(\frac{1}{64}\times 0 + \frac{9}{64}\times 10 + \frac{27}{64}\times 70 + \frac{27}{64}\times 70,\right.$$

$$\left.\frac{1}{64}\times 10 + \frac{9}{64}\times 50 + \frac{27}{64}\times 40 + \frac{27}{64}\times -20\right)$$

$$= \left(\frac{90}{64} + \frac{1890}{64} + \frac{1890}{64}, \frac{10}{64} + \frac{450}{64} + \frac{1080}{64} - \frac{540}{64}\right)$$

$$= \underline{(60.46, 15.62)}$$

$$P(1) = \underline{\underline{(70, -20)}}$$



OR

ITERATION 1:

Mid of AB = AB’

AB’ = [(Ax + Bx)/2, (Ay + By)/2)]

= [(0+10)/2, (10+50)/2]

= [(10)/2, (60)/2]

= (5, 30)

Mid of BC = BC’

BC’ = [(Bx + Cx)/2, (By + Cy)/2)]

= [(10+70)/2, (50+40)/2]

= [(80)/2, (90)/2]

= (40, 45)

Mid of CD = CD’

CD’ = [(Cx + Dx)/2, (Cy + Dy)/2)]

= [(70+70)/2, (40+(-20))/2]

= [(140)/2, (20)/2]

= (70, 10)

ITERATION 2:

Mid of ABC = ABC’

ABC’ = [(ABx + BCx)/2, (ABy + BCy)/2)]

= [(5+40)/2, (30+45)/2]

= [(45)/2, (75)/2]

= (22.5, 37.5)

Mid of BCD = BCD’

BCD’ = [(BCx + CDx)/2, (BCy + CDy)/2)]

= [(40+70)/2, (45+10)/2]

= [(110)/2, (55)/2]

= (55, 27.5)

ITERATION 3:

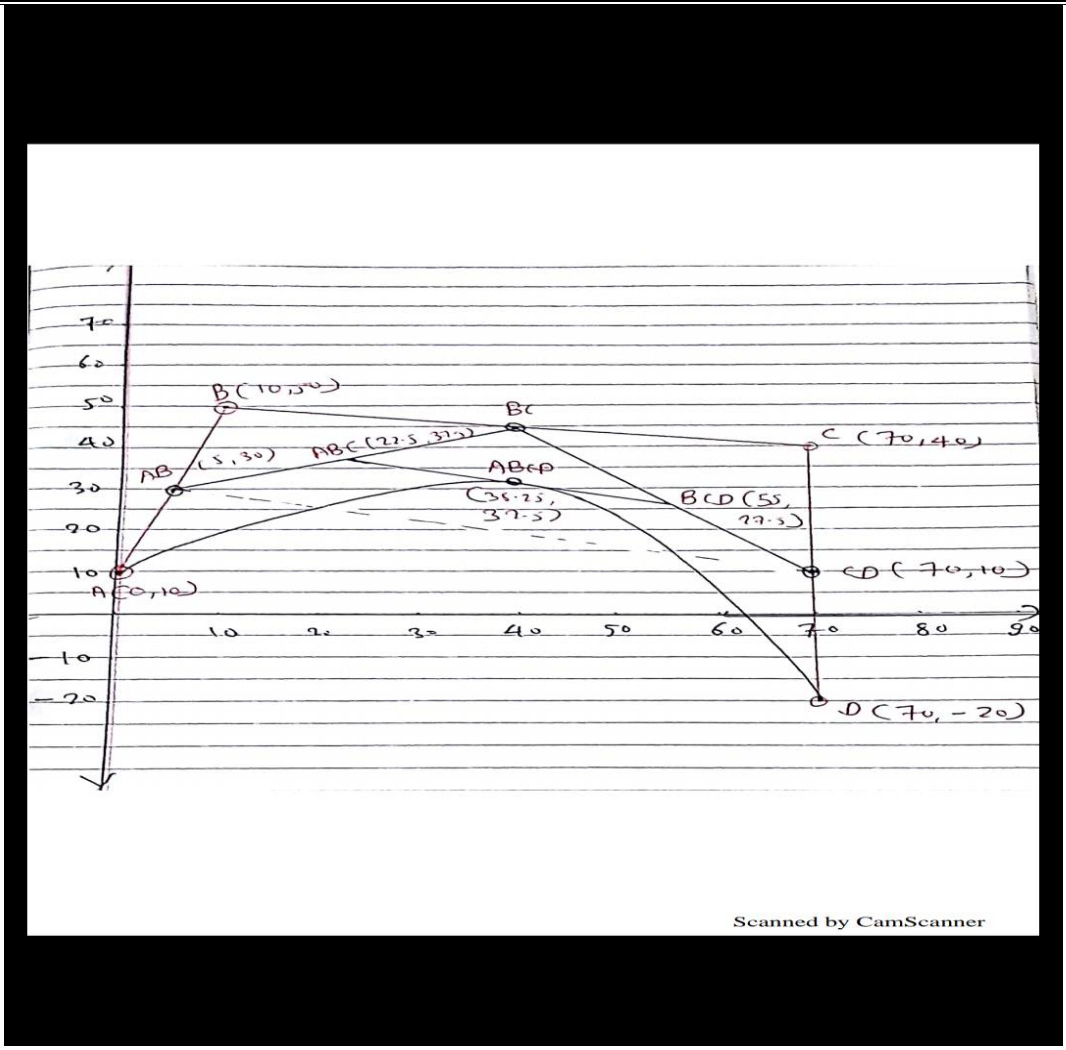
Mid of ABCD = ABCD’

ABCD’ = [(ABCx + BCDx)/2, (ABCy + BCDy)/2)]

= [(22.5+55)/2, (37.5+27.5)/2]

= [(77.5)/2, (65)/2]

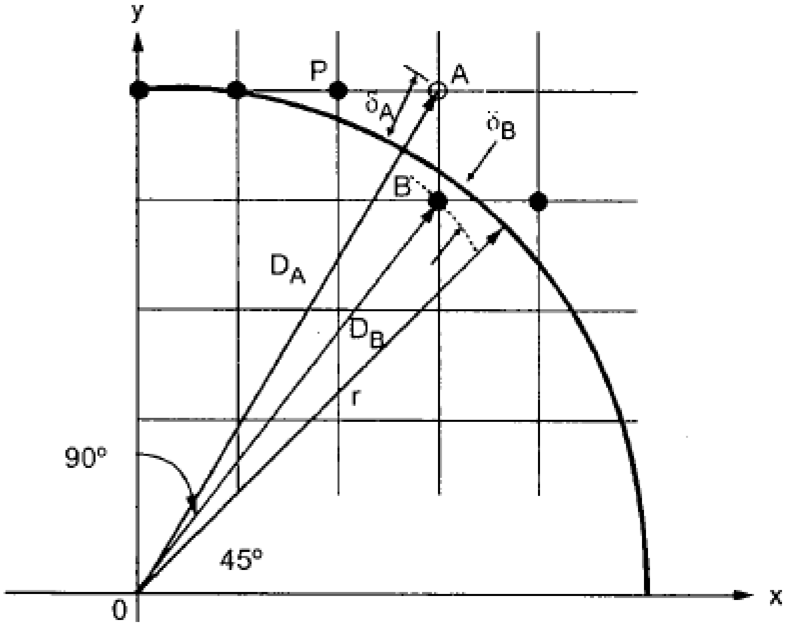
= (38.25, 32.5)

			
6		**Attempt any two of the following:**	**12 M**
	a	**Write matrices in homogeneous co-ordinates system for 3D scaling transformation.**	6M
	Ans	3D transformation matrix for scaling is as follows: $S=\begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$	Correct matrix 6 Marks

		It specifies three co-ordinates with their own scaling factors. If scale factors, $S_x = S_y = S_z = S > 1$ then the scaling is called as magnification. $S_x = S_y = S_z = S < 1$ then the scaling is called as reduction. Therefore, point after scaling with respect to origin can be calculated as, $\therefore \quad P=P \cdot S$	
	b	**Write down Cyrus-Beck line clipping algorithm.**	6M
	Ans	See algorithm below	Correct algorithm 6 marks
	c	**Derive the expression for decision parameter used in Bresenhaum's circle drawing algorithm.**	6M

```
Step 1: Read end points of line $P_1$ and $P_2$.
Step 2: Read vertex coordinates of clipping window.
Step 3: Calculate $D = P_2 – P_1$.
Step 4: Assign boundary point b with particular edge.
Step 5: Find inner normal vector for corresponding edge.
Step 6:Calculate D.n and $W = P_1 – b$
Step 7:If D.n > 0
        $t_L = – (W.n)/(D.n)$
        else
        $t_U = – (W.n)/(D.n)$
        end if
Step 8: Repeat steps 4 through 7 for each edge of clipping window.
Step 9: Find maximum lower limit and minimum upper limit.
Step 10: If maximum lower limit and minimum upper limit do not satisfy
  condition $0 \le t \le 1$ then ignore line.
Step 11: Calculate intersection points by substituting values of maximum lower
limit and minimum upper limit in parametric equation of line $P_1P_2$.
Step 12: Draw line segment $P(t_L)$ to $P(t_U)$.
Step 13: Stop.
```

Ans



The distances of pixels A and B from the origin are given as

$$D_A = \sqrt{(x_{i+1})^2 + (y_i)^2} \quad \text{and}$$
$$D_B = \sqrt{(x_{i+1})^2 + (y_i - 1)^2}$$

Now, the distances of pixels A and B from the true circle are given as

$$\delta_A = D_A - r \text{ and } \delta_B = D_B - r$$

However, to avoid square root term in derivation of decision variable, i.e. to simplify the computation and to make algorithm more efficient the δ_A and δ_B are defined as

$$\delta_A = D_A^2 - r^2 \quad \text{and}$$
$$\delta_B = D_B^2 - r^2$$

From Fig. , we can observe that δ_A is always positive and δ_B always negative. Therefore, we can define **decision variable** d_i as

$$d_i = \delta_A + \delta_B$$

and we can say that, if $d_i < 0$, i.e., $\delta_A < \delta_B$ then only x is incremented; otherwise x is incremented in positive direction and y is incremented in negative direction. In other words we can write,

For $d_i < 0$, $x_{i+1} = x_i + 1$ and
For $d_i \geq 0$, $x_{i+1} = x_i + 1$ and $y_{i+1} = y_i - 1$

The equation for d_i at starting point, i.e. at $x = 0$ and $y = r$ can be simplified as follows

$$\begin{aligned} d_i &= \delta_A + \delta_B \\ &= (x_i + 1)^2 + (y_i)^2 - r^2 + (x_i + 1)^2 + (y_i - 1)^2 - r^2 \\ &= (0 + 1)^2 + (r)^2 - r^2 + (0 + 1)^2 + (r - 1)^2 - r^2 \\ &= 1 + r^2 - r^2 + 1 + r^2 - 2r + 1 - r^2 \\ &= 3 - 2r \end{aligned}$$

Similarly, the equations for d_{i+1} for both the cases are given as

For $d_i < 0$, $d_{i+1} = d_i + 4x_i + 6$ and
For $d_i \leq 0$, $d_{i+1} = d_i + 4(x_i - y_i) + 10$

Correct method and correct equation 6 Marks

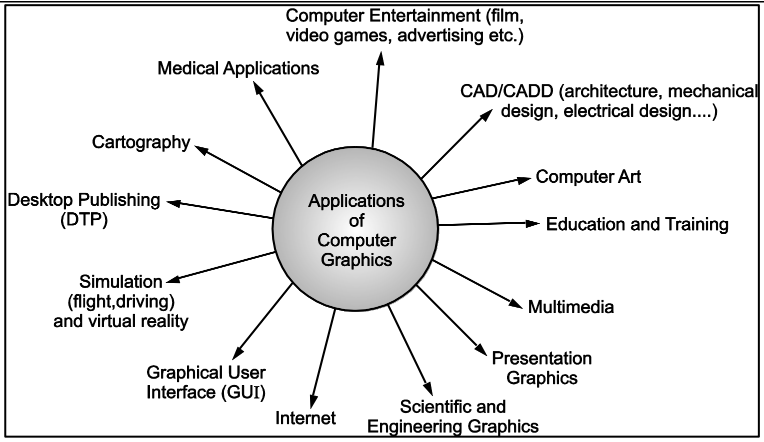
SUMMER – 19 EXAMINATION

Subject Name: Computer Graphics **Model Answer** **Subject Code: 22318**

Important Instructions to examiners:

1) The answers should be examined by key words and not as word-to-word as given in the model answer scheme.
2) The model answer and the answer written by candidate may vary but the examiner may try to assess the understanding level of the candidate.
3) The language errors such as grammatical, spelling errors should not be given more Importance (Not applicable for subject English and Communication Skills.
4) While assessing figures, examiner may give credit for principal components indicated in the figure. The figures drawn by candidate and model answer may vary. The examiner may give credit for any equivalent figure drawn.
5) Credits may be given step wise for numerical problems. In some cases, the assumed constant values may vary and there may be some difference in the candidate's answers and model answer.
6) In case of some questions credit may be given by judgement on part of examiner of relevant answer based on candidate's understanding.
7) For programming language papers, credit may be given to any other program based on equivalent concept.

Q. No.	Sub Q. N.	Answer	Marking Scheme
1		**Attempt any FIVE of the following**	**10 M**
	a	**Define aspect ratio. Give one example of an aspect ratio**	**2 M**
	Ans	**Aspect ratio:** It is the ratio of the number of vertical points to the number of horizontal points necessary to produce equal length lines in both directions on the screen. **or** In computer graphics, the relative horizontal and vertical sizes. For example, if a graphic has an aspect ratio of 2:1, it means that the width is twice as large as the height. **or** Aspect ratio is the ratio between width of an image and the height of an image. **Example:** The term is also used to describe the dimensions of a display resolution. For example, a resolution of 800x600, 1027x768, 1600x1200 has an aspect ratio of 4:3. Resolution 1280x1024 has an aspect ratio 5:4 Resolution 2160x1440, 2560x1700 has an aspect ratio 3:2	Definition-1M Example-1M
	b	**List any four applications of computer graphics.**	**2 M**

<table>
<tr><td></td><td>Ans</td><td>


- DTP (Desktop Publishing)
- Graphical User Interface (GUI)
- Computer-Aided Design
- Computer-Aided Learning (Cal)
- Animations
- Computer Art
- Entertainment
- Education and training
- Image processing
- Medical Applications
- Presentation and Business Graphics
- Simulation and Virtual Reality
</td><td>Listing of four applications-2 M</td></tr>
<tr><td></td><td>c</td><td>Define virtual reality. List any two advantages of virtual reality.</td><td>2 M</td></tr>
<tr><td></td><td>Ans</td><td>
Virtual reality (VR) means experiencing things through our computers that don't really exist.

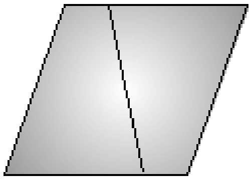
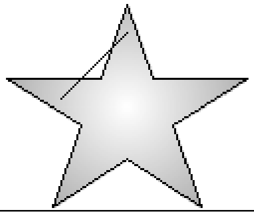
OR

Virtual Reality (VR) is the use of computer technology to create a simulated environment. Instead of viewing a screen in front of them, users are immersed and able to interact with 3D worlds.

Advantages:

- Virtual reality creates a realistic world
- Through Virtual Reality user can experiment with an artificial environment.
- Virtual Reality make the education more easily and comfort.
- It enables user to explore places.
- Virtual Reality has made watching more enjoyable than reading.

Virtual reality widely used in video games, engineering, entertainment, education, design, films, media, medicine and many more.
</td><td>Definition-1M
Any two Advantages-1 M</td></tr>
<tr><td></td><td>d</td><td>List any two line drawing algorithms. Also, list two merits of any line drawing algorithm.</td><td>2 M</td></tr>
<tr><td></td><td>Ans</td><td>Line drawing algorithms:</td><td>Listing-1 M</td></tr>
</table>

		• Digital Differential Analyzer (DDA) algorithm • Bresenham’s algorithm **Merits of DDA algorithms:** • It is the simplest algorithm and it does not require special skills for implementation. • It is a faster method for calculating pixel positions than the direct use of equation y = mx + b. It eliminates the multiplication in the equation by making use of raster characteristics, so that appropriate increments are applied in the x or v direction to find the pixel positions along the line path • Floating point Addition is still needed. **Merits of Bresenham’s Algorithm:** • Bresenhams algorithm is faster than DDA algorithm • Bresenhams algorithm is more efficient and much accurate than DDA algorithm. • Bresenham's line algorithm is a highly efficient incremental method over DDA. • Bresenhams algorithm can draw circles and curves with much more accuracy than DDA algorithm. It produces mathematically accurate results using only integer addition, subtraction, and multiplication by 2, which can be accomplished by a simple arithmetic shift operation.	Two merits-1 M
	e	**Define convex and concave polygons.**	**2 M**
	Ans	**Convex Polygon:** It is a polygon in which if you take any two positions of polygon then all the points on the line segment joining these two points fall within the polygon itself.  **Concave Polygon:** It is a polygon in which if you take any two positions of polygon then all the points on the line segment joining these two points does not fall entirely within the polygon. 	Each 1 M
	f	**What is homogeneous co-ordinate? Why is it required?**	**2 M**
	Ans	Homogeneous coordinates are another way to represent points to simplify the way	Definition-1

in which we express affine transformations.

Normally, book-keeping would become tedious when affine transformations of the form $A\bar{p} + \vec{t}$ are composed. With homogeneous coordinates, affine transformations become matrices, and composition of transformations is as simple as matrix multiplication.

With homogeneous coordinates, a point $\bar{p}$ is augmented with a 1, to form $\hat{p} = \begin{bmatrix} \bar{p} \\ 1 \end{bmatrix}$

.

All points $(\alpha\bar{p}, \alpha)$ represent the same point $\bar{p}$ for real $\alpha \neq 0$.

M
Why required-1 M

OR

We have to use 3×3 transformation matrix instead of 2×2 transformation matrix. To convert a 2×2 matrix to 3×3 matrix, we have to add an extra dummy coordinate W. In this way, we can represent the point by 3 numbers instead of 2 numbers, which is called Homogenous Coordinate system.

- Homogeneous coordinates are used extensively in computer vision and graphics because they allow common operations such as translation, rotation, scaling and perspective projection to be implemented as matrix operations

3D graphics hardware can be specialized to perform matrix multiplications on 4x4 matrices.

g **Write the transformation matrix for y-shear.** **2 M**

Ans The Y-Shear can be represented in matrix from as:

$$Y_{sh} \begin{bmatrix} 1 & 0 & 0 \\ shy & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$X' = X + Sh_x . Y$$

$$Y' = Y$$

For matrix-2 M

2 **Attempt any THREE of the following** **12 M**

a **Compare vector scan display and raster scan display (write any 4 points)** **4M**

Ans

Raster	**Vector**
Raster graphics are composed of pixels.	Vector graphics are composed of paths.
Raster graphics are resolution dependent.	Vector graphics are resolution independent
More expensive	Less expensive.
Graphics primitives are specified in terms of their endpoints and must be scan converted into their	Scan conversion is not required

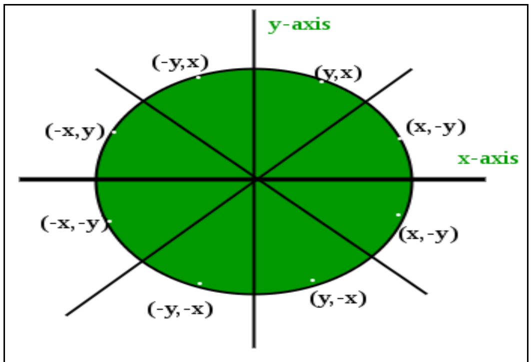
Any four point-4 M

corresponding points in the frame buffer.	
It required separate scan conversion hardware.	Scan conversion hardware is not required.
Raster display has ability to display areas filled with solid colors or patterns.	Vector display only draws lines and characters
It uses interlacing	It does not used interlacing
This displays have lower resolution	This displays have higher resolution
They occupies more space which depends on image quality.	They occupies less space
File extensions are: .bmp, .gif, .jpg, .tif	File extensions are: .pdf, .ai, .svg, .eps, .dxf

b **Rephrase the Bresenham’s algorithm to plot 1/8th of the circle and write the algorithm required to plot the same.** **4M**

Ans The key feature of circle that it is highly symmetric. So, for whole 360 degree of circle we will divide it in 8-parts each octant of 45 degree. In order to that we will use Bresenham’s Circle Algorithm for calculation of the locations of the pixels in the first octant of 45 degrees. It assumes that the circle is centered on the origin. So for every pixel (x, y) it calculates, we draw a pixel in each of the 8 octants of the circle as shown below:

Rephrase-2 M
Algorithm-2 M



For a pixel (x,y) all possible pixels in 8 octants.

Algorithm:

```
Step 1: Read the radius of circle (r).
Step 2: Set decision parameter $d = 3 – 2r$.
Step 3: x=0 and y=r.
Step 4:     do
              {
              Plot (x,y)
              If(d<0) then
```

```
  {
d = d + 4x + 6
  }
Else
  {
d=d + 4(x-y) + 10
y=y-1
  }
X=x-1
  }
While(x<y)
```

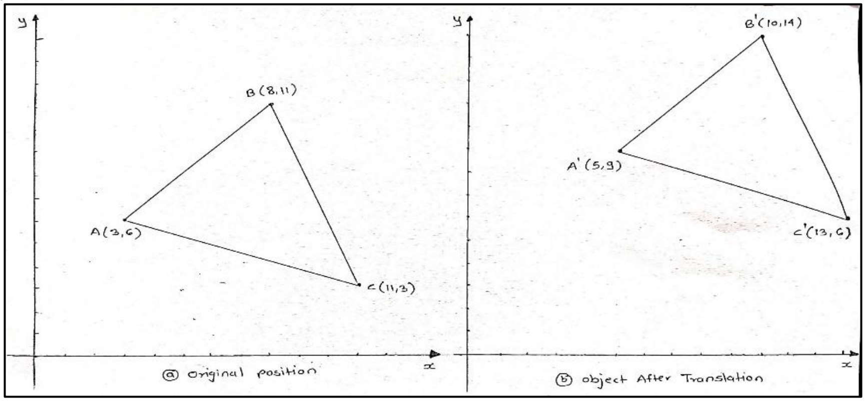
Step 5: stop

Plotting 8 points, each point in one octant

Call Putpixel (X + h, Y + k).
Call Putpixel (-X + h, Y + k).
Call Putpixel (X + h, -Y + k).
Call Putpixel (-X + h, -Y + k).
Call Putpixel (Y + h, X + k).
Call Putpixel (-Y + h, X + k).
Call Putpixel (Y + h, -X - k).
Call Putpixel (-Y + h,-X + k).

	c	**Translate the polygon with co-ordinates A (3, 6), B (8, 11), & C (11, 3) by 2 units in X direction and 3 units in Y direction.**	**4M for proper solution**
	Ans	X’=x+tx Y’=y+ty tx=2 ty=3 for point A(3,6) x’=3+2=5 y’=6+3=9 for point B(8,11) x’=8+2=10 y’=11+3=14 for point C(11,3) x’=11+2=13 y’=3+3=6 A’=(x’,y’)=(5,9)	

B’=(x’,y’)=(10,14)
C’=(x’,y’)=(13,6)



d **Write the midpoint subdivision algorithm for line clipping.** **4M**

Ans

Algorithm-4 M

Step 1: Scan two end points for the line $P_1(x_1, y_1)$ and $P_2(x_2, y_2)$.

Step 2: Scan corners for the window as $(\omega x_1, \omega y_1)$ and $(\omega x_2, \omega y_2)$.

Step 3: Assign the region codes for endpoints P_1 and P_2 by initializing code with 0000.

Bit 1 - if $(x < \omega x_1)$
Bit 2 - if $(x > \omega x_2)$
Bit 3 - if $(y < \omega y_2)$
Bit 4 - if $(y > \omega y_1)$

Step 4: Check for visibility of line P_1, P_2.

- If region codes for both end points are zero then the line is visible, draw it and jump to step 6.
- If region codes for end points are not zero and the logical Anding operation of them is also not zero then the line is invisible, reject it and jump to step 6.
- If region codes for end points does not satisfies the condition in 4 (i) and 4 (ii) then line is partly visible.

Step5: Find midpoint of line and divide it into two equal line segments and repeat steps 3 through 5 for both subdivided line segments until you get completely visible and completely invisible line segments.

Step 6: Exit.

3 **Attempt any THREE of the following** **12 M**

a **State the different character generation methods. Describe any one with diagram.** **4 M**

Ans

Listing-1 M and any one method-3 M

Character Generator Methods:

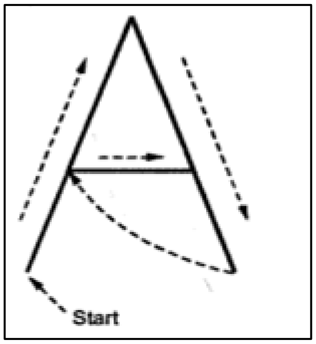
1) Stroke Method

2) Bitmap Method

3) Starburst Method

1) STROKE METHOD

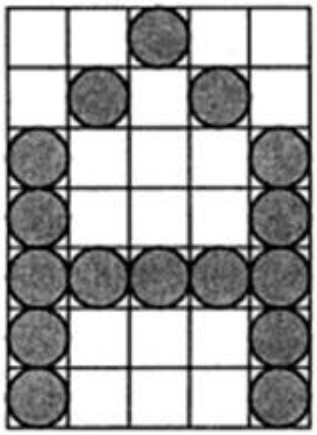
- Stroke method is based on natural method of text written by human being. In this method graph is drawing in the form of line by line.
- Line drawing algorithm DDA follows this method for line drawing.
- This method uses small line segments to generate a character. The small series of line segments are drawn like a stroke of pen to form a character.
- We can build our own stroke method character generator by calls to the line drawing algorithm. Here it is necessary to decide which line segments are needed for each character and then drawing these segments using line drawing algorithm.



2)BITMAP METHOD

- Bitmap method is a called dot-matrix method as the name suggests this method use array of bits for generating a character. These dots are the points for array whose size is fixed.
- In bit matrix method when the dots is stored in the form of array the value 1 in array represent the characters i.e. where the dots appear we represent that position with numerical value 1 and the value where dots are not present is represented by 0 in array.
- It is also called dot matrix because in this method characters are represented by an array of dots in the matrix form. It is a two dimensional array having columns and rows.

A 5x7 array is commonly used to represent characters. However 7x9 and 9x13 arrays are also used. Higher resolution devices such as inkjet printer or laser printer may use character arrays that are over 100x100.



Character A in 5 × 7 dot matrix format

3) Starbust method:

In this method a fix pattern of line segments are used to generate characters. Out of these 24 line segments, segments required to display for particular character are highlighted. This method of character generation is called starbust method because of its characteristic appearance

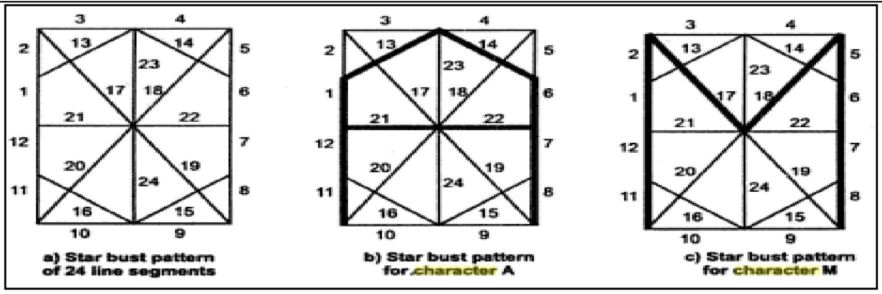
The starbust patterns for characters A and M. the patterns for particular characters are stored in the form of 24 bit code, each bit representing one line segment. The bit is set to one to highlight the line segment; otherwise it is set to zero. For example, 24-bit code for Character A is 0011 0000 0011 1100 1110 0001 and for character M is 0000 0011 0000 1100 1111 0011.

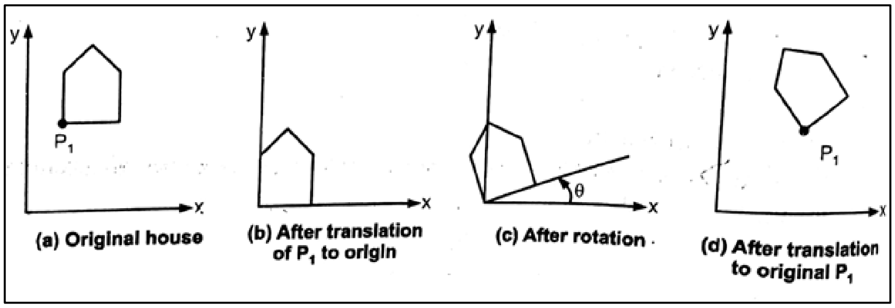
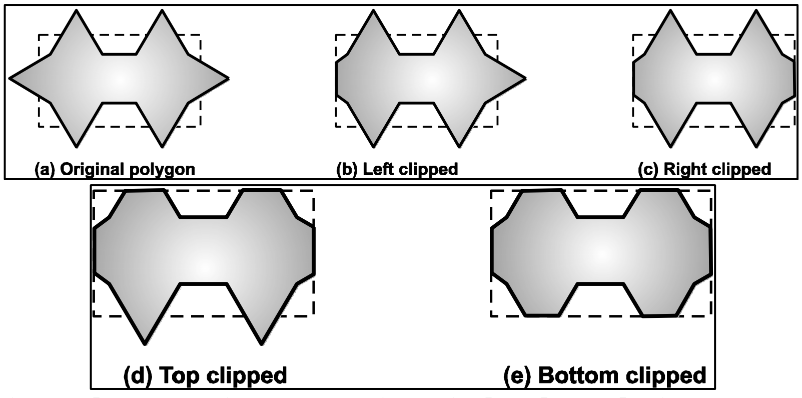
This method of character generation has some disadvantages. They are

1. The 24-bits are required to represent a character. Hence more memory is required.

2. Requires code conversion software to display character from its 24-bit code.

3. Character quality is poor. It is worst for curve shaped characters.

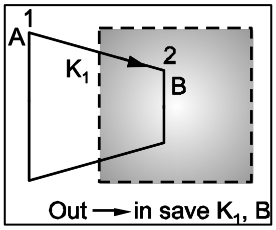
		 a) Star bust pattern of 24 line segments b) Star bust pattern for character A c) Star bust pattern for character M Character A : 0011 0000 0011 1100 1110 0001 Character M:0000 0011 0000 1100 1111 0011	
	b	**Obtain a transformation matrix for rotating an object about a specified pivot point.**	**4 M**
	Ans	To do rotation of an object about any selected arbitrary point P1(x1 ,y1), following sequence of operations shall be performed. **1. Translate:** Translate an object so that arbitrary point P1 is moved to coordinate origin. **2. Rotate:** Rotate object about origin. **3. Translate:** Translate object so that arbitrary point P1 is moved back to the its original position. Note: Here to do one operation we are doing the sequence of three operations. So it is called as composite transformation or concatenation. Rotate about point P1(x1,y1). 1) Translate P1 to origin. 2) Rotate 3) Translate back to P1. Equation for this composite transformation matrix form is as follows: $P' = T(x_1, y_1) \,.\, R(\theta).\, T(-x_1, -y_1)$ $P' = \begin{bmatrix} 1 & 0 & x_1 \\ 0 & 1 & y_1 \\ 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} 1 & 0 & -x_1 \\ 0 & 1 & -y_1 \\ 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$ Here (x1,y1) are coordinates of point P1 and hence are translation factors tx and ty; we want to move P1 to origin , x1 and y1 are x and y distances to P1and hence it is translation factor.	Proper Explanation 4 M

		$$P' = \begin{bmatrix} \cos\theta & -\sin\theta & x_1\,(1-\cos\theta)+y_1\,\sin\theta \\ \sin\theta & \cos\theta & y_1\,(1-\cos\theta)-x_1\,\sin\theta \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$ It is demonstrated in following figure: 	
	c	**Describe Sutherland-Hodgeman algorithm for polygon clipping.**	
	Ans	• In Sutherland-Hodgeman, a polygon is clipped by processing the polygon boundary as a whole against each window edge. Clipping window must be convex. • This could be accomplished by processing all polygon vertices against each clip rectangle boundary in turn beginning with the original set of polygon vertices, first clip the polygon against the left rectangle boundary to produce a new sequence of vertices. • The new set of vertices could then be successively passed to a right boundary clipper, a top boundary clipper and a bottom boundary clipper. • At each step a new set of polygon vertices us generated and passed to the next window boundary clipper. This is the logic used in Sutherland-Hodgeman algorithm.  **Fig. Clipping polygon against successive window boundaries** • The output of algorithm is a list of polygon vertices all of which are on the visible side of clipping plane. Such each edge of the polygon is individually compared with the clipping plane. • This is achieved by processing two vertices of each edge of the polygon	Explanation-1 M and Algorithm-3 M

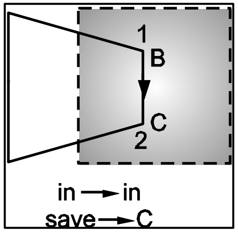
around the clipping boundary or plane.

- This results in four possible relationships between the edge and clipping plane.

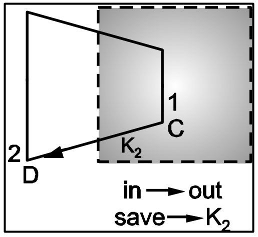
1. If first vertex of polygon edge is outside and second is inside window boundary, then intersection point of polygon edge with window boundary and second vertex are added to output vertices set as shown in Fig. 6.13.



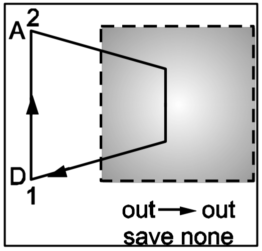
2. If both vertices of edge are inside window boundary, then add only second vertex to output set as shown in Fig. 6.14.



3. If first vertex of edge is inside and second is outside of window boundary then point of intersection of edge with window boundary is stored in output set as shown in Fig. 6.15.



4. If both vertices of edges are outside of window boundary then those vertices are rejected as shown in Fig. 6.16.



- Going through above four cases we can realize that there are two key processes in this algorithm:
  1. Determine the visibility of point or vertex (Inside – Outside Test)
  2. Determine the intersection of the polygon edge and clipping plane.
- The second key process in Sutherland-Hodgeman polygon clipping algorithm is to determine the intersection of the polygon edge and clipping plane.

- Assume that we're clipping a polygon's edge with vertices at (x_1, y_1) and (x_2, y_2) against a clip window with vertices at (x_{min}, y_{min}) and (x_{max}, y_{max}).
  1. The location (I_X, I_Y) of the intersection of the edge with the left side of the window is:
     (i) $I_X = x_{min}$
     (ii) $I_Y = slope*(x_{min} – x_1) + y_1$, where the slope = $(y_2 – y_1)/(x_2 – x_1)$.
  2. The location of the intersection of the edge with the right side of the window is:
     (i) $I_X = x_{max}$
     (ii) $I_Y = slope*(x_{max} – x_1) + y_1$, where the slope = $(y_2 – y_1)/(x_2 – x_1)$
  3. The intersection of the polygon's edge with the top side of the window is:
     (i) $I_X = x_1 + (y_{max} – y_1) / slope$
     (ii) $I_Y = y_{max}$
  4. Finally, the intersection of the edge with the bottom side of the window is:
     (i) $I_X = x_1 + (y_{min} – y_1) / slope$
     (ii) $I_Y = y_{min}$

Algorithm for Sutherland-Hodgeman Polygon Clipping:

Step 1: Read co-ordinates of all vertices of the polygon.

Step 2: Read co-ordinates of the clipping window.

Step 3: Consider the left edge of window.

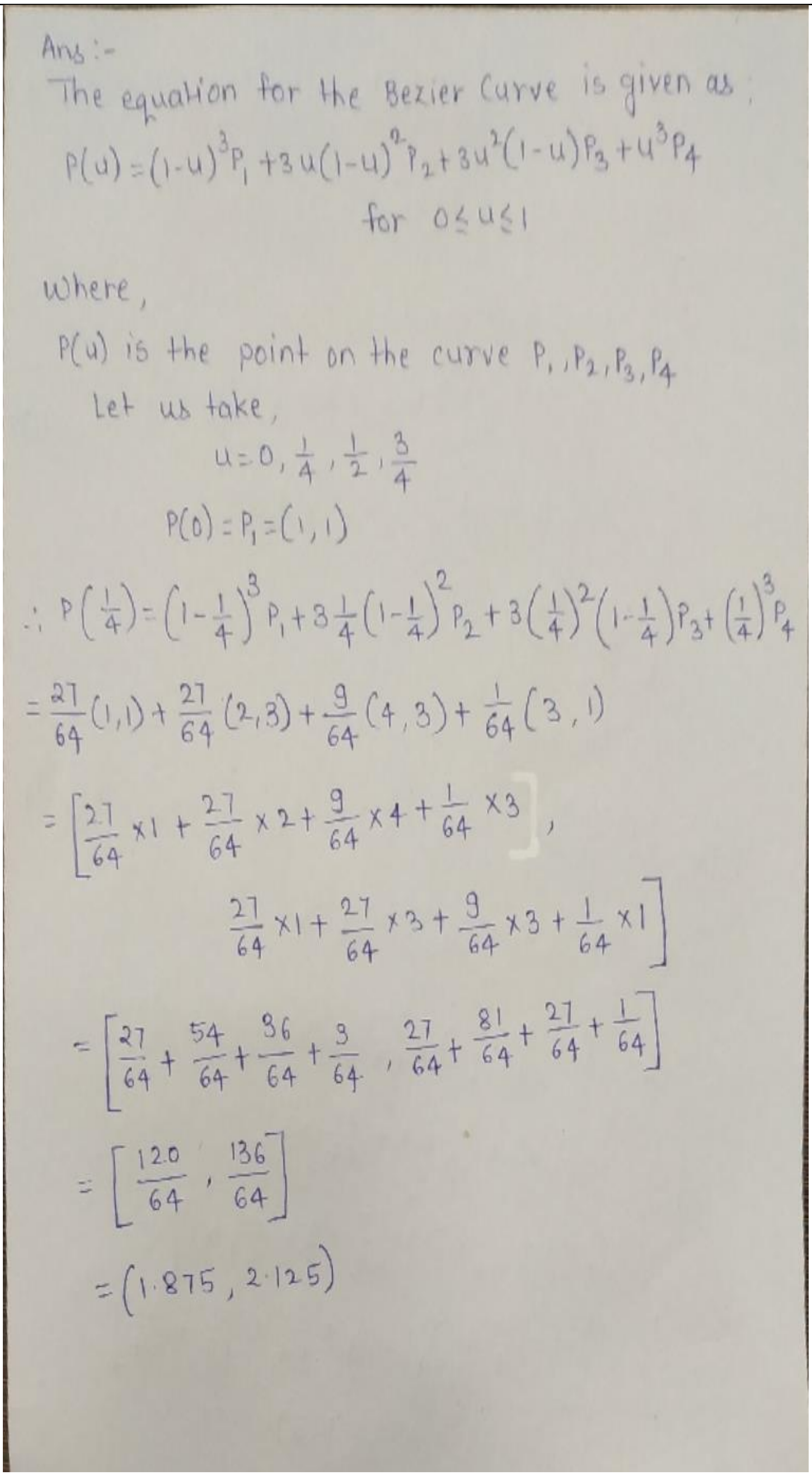
Step 4: Compare vertices of each of polygon, individually with the clipping plane.

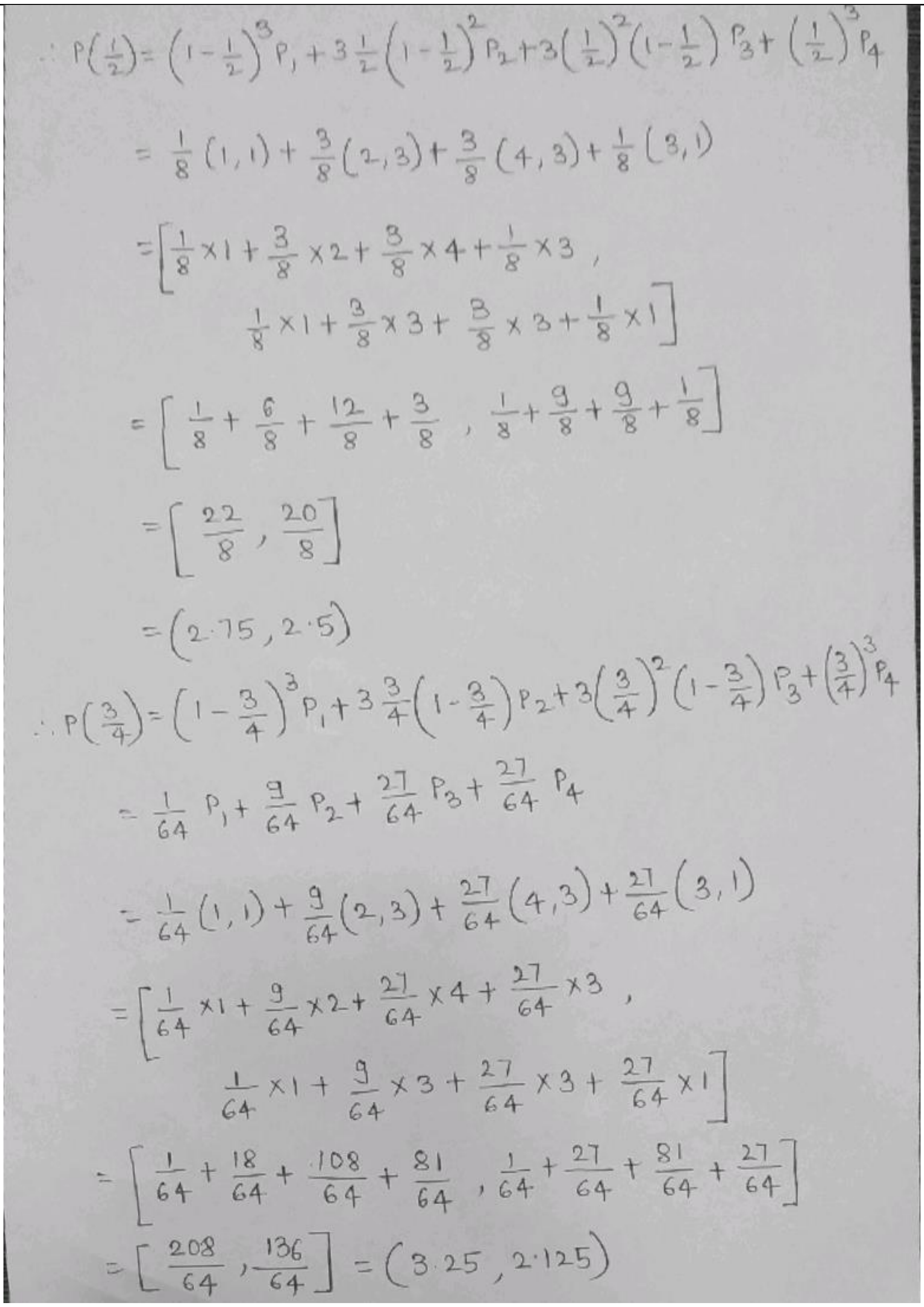
Step 5: Save the resulting intersections and vertices in the new list of vertices according to four possible relationships between the edge and the clipping boundary.

Step 6: Repeat the steps 4 and 5 for remaining edges of clipping window. Each time resultant list of vertices is successively passed to process next edge of clipping window.

Step 7: Stop.

	d	**Given the vertices of Bezier Polygon as $P_0(1, 1)$, $P_1(2,3)$, $P_2(4,3)$, $P_3(3,1)$, determine five points on Bezier Curve.**	**4 M**

	Ans		 Ans :- The equation for the Bezier Curve is given as : $P(u) = (1-u)^3 P_1 + 3u(1-u)^2 P_2 + 3u^2(1-u) P_3 + u^3 P_4$ for $0 \le u \le 1$ where, P(u) is the point on the curve P_1, P_2, P_3, P_4 Let us take, $u = 0, \frac{1}{4}, \frac{1}{2}, \frac{3}{4}$ $P(0) = P_1 = (1,1)$ $\therefore P\left(\frac{1}{4}\right) = \left(1-\frac{1}{4}\right)^3 P_1 + 3\frac{1}{4}\left(1-\frac{1}{4}\right)^2 P_2 + 3\left(\frac{1}{4}\right)^2\left(1-\frac{1}{4}\right)P_3 + \left(\frac{1}{4}\right)^3 P_4$ $= \frac{27}{64}(1,1) + \frac{27}{64}(2,3) + \frac{9}{64}(4,3) + \frac{1}{64}(3,1)$ $= \left[\frac{27}{64} \times 1 + \frac{27}{64} \times 2 + \frac{9}{64} \times 4 + \frac{1}{64} \times 3\right.,$ $\left.\frac{27}{64} \times 1 + \frac{27}{64} \times 3 + \frac{9}{64} \times 3 + \frac{1}{64} \times 1\right]$ $= \left[\frac{27}{64} + \frac{54}{64} + \frac{36}{64} + \frac{3}{64}, \frac{27}{64} + \frac{81}{64} + \frac{27}{64} + \frac{1}{64}\right]$ $= \left[\frac{120}{64}, \frac{136}{64}\right]$ $= (1.875, 2.125)$	Proper result 4 M



$$\therefore P\left(\tfrac{1}{2}\right) = \left(1-\tfrac{1}{2}\right)^3 P_1 + 3\,\tfrac{1}{2}\left(1-\tfrac{1}{2}\right)^2 P_2 + 3\left(\tfrac{1}{2}\right)^2\left(1-\tfrac{1}{2}\right) P_3 + \left(\tfrac{1}{2}\right)^3 P_4$$

$$= \tfrac{1}{8}(1,1) + \tfrac{3}{8}(2,3) + \tfrac{3}{8}(4,3) + \tfrac{1}{8}(3,1)$$

$$= \left[\tfrac{1}{8}\times 1 + \tfrac{3}{8}\times 2 + \tfrac{3}{8}\times 4 + \tfrac{1}{8}\times 3,\ \tfrac{1}{8}\times 1 + \tfrac{3}{8}\times 3 + \tfrac{3}{8}\times 3 + \tfrac{1}{8}\times 1\right]$$

$$= \left[\tfrac{1}{8} + \tfrac{6}{8} + \tfrac{12}{8} + \tfrac{3}{8},\ \tfrac{1}{8} + \tfrac{9}{8} + \tfrac{9}{8} + \tfrac{1}{8}\right]$$

$$= \left[\tfrac{22}{8}, \tfrac{20}{8}\right]$$

$$= (2.75, 2.5)$$

$$\therefore P\left(\tfrac{3}{4}\right) = \left(1-\tfrac{3}{4}\right)^3 P_1 + 3\,\tfrac{3}{4}\left(1-\tfrac{3}{4}\right) P_2 + 3\left(\tfrac{3}{4}\right)^2\left(1-\tfrac{3}{4}\right) P_3 + \left(\tfrac{3}{4}\right)^3 P_4$$

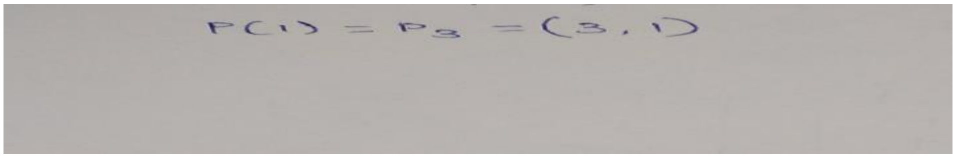
$$= \tfrac{1}{64} P_1 + \tfrac{9}{64} P_2 + \tfrac{27}{64} P_3 + \tfrac{27}{64} P_4$$

$$= \tfrac{1}{64}(1,1) + \tfrac{9}{64}(2,3) + \tfrac{27}{64}(4,3) + \tfrac{27}{64}(3,1)$$

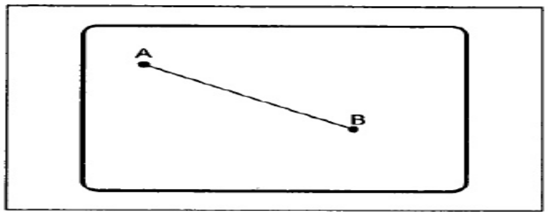
$$= \left[\tfrac{1}{64}\times 1 + \tfrac{9}{64}\times 2 + \tfrac{27}{64}\times 4 + \tfrac{27}{64}\times 3,\ \tfrac{1}{64}\times 1 + \tfrac{9}{64}\times 3 + \tfrac{27}{64}\times 3 + \tfrac{27}{64}\times 1\right]$$

$$= \left[\tfrac{1}{64} + \tfrac{18}{64} + \tfrac{108}{64} + \tfrac{81}{64},\ \tfrac{1}{64} + \tfrac{27}{64} + \tfrac{81}{64} + \tfrac{27}{64}\right]$$

$$= \left[\tfrac{208}{64}, \tfrac{136}{64}\right] = (3.25, 2.125)$$



$$P(1) = P_3 = (3, 1)$$

4		**Attempt any THREE of the following**	**12 M**
	a	**Describe the vector scan display techniques with neat diagram.**	**4 M**
	Ans	• A pen plotter operates in a similar way and is an example of a random-scan, hard-copy device. • When operated as a random-scan display unit, a CRT has the electron beam directed only to the parts of the screen where a picture is to be drawn. • Random scan monitors draw a picture one line at a time and for this reason are also referred to as vector displays (or stroke-writing or calligraphic displays).  • Here the electron gun of a CRT illuminate's points and / or straight lines in any order. If we want a line connecting point A with point B on vector graphics display, we simply drive the beam reflection circuitry, which will cause beam to go directly from point A to point B. • Refresh rate on a random-scan system depends on the number of lines to be displayed. • Picture definition stored as a set of line drawing commands in an area of memory called "***refresh display file***" or also called as ***display list*** or ***display program*** or ***refresh buffer***. • To display a given picture, the system cycles through the set of commands in the display file, drawing each component line by line in turn. After all line drawing commands have been processed, the system cycles back to the first line drawing command in the list. And repeats the procedure of scan, display and retrace. • This displays to draw all the component lines of picture 30 to 60 frames/second • Random scan system is designed for line drawing applications; hence cannot display realistic shaded scenes. • Vector displays produces smooth line drawings but raster produces jagged lines that are plotted points • Random scan suitable for applications like engineering and scientific drawings • Graphics patterns are displayed by directing the electron beam along the component lines of the picture • A scene is then drawn one line at a time by positioning the beam to fill in the line between specified end points.	Explanation 3 M Diagram 1 M

b **Consider the line from (0,0) to (4,6). Use the simple DDA algorithm to rasterize this line.** **4 M**

Ans Evaluating steps 1 to 5 in the DDA algorithm we have,

Proper result 4 M

$X_l = 0, Y_1 = 0$

$X_2 = 4, Y_2 = 6$

Length $= |Y_2 – Y_1| = 6$

$\Delta X=|X_2 – X_1| / \text{Length} = 4/6$

$\Delta Y =|Y_2 – Y_l| / \text{Length} = 6/6 = 1$

Initial value for,

$X=0 + 0.5 \times (4/6) = 0.5$

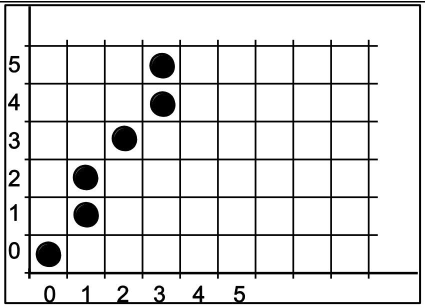
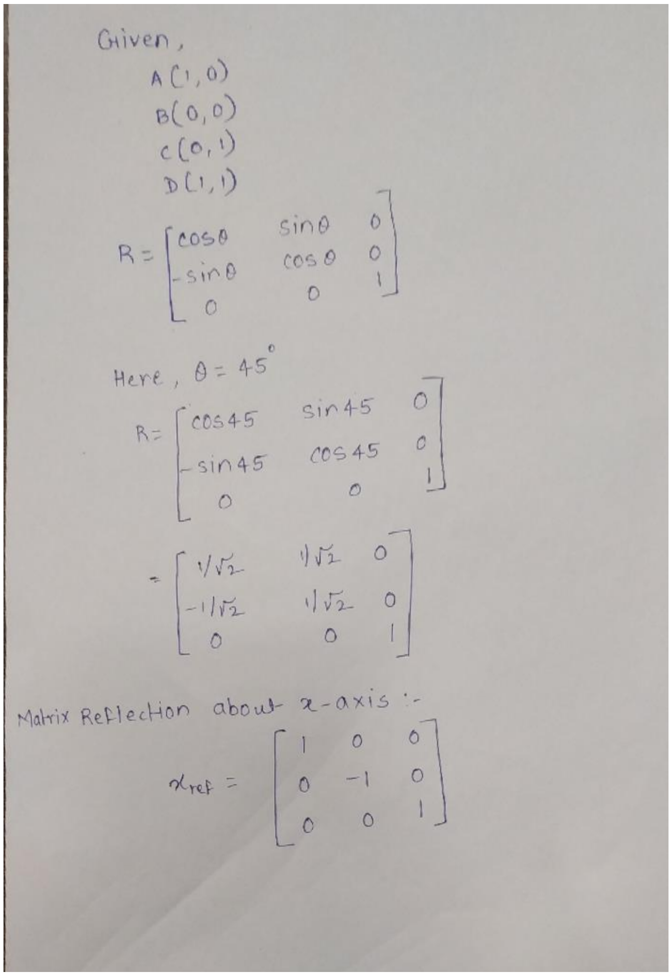
$Y =0 + 0.5 \times (1) = 0.5$

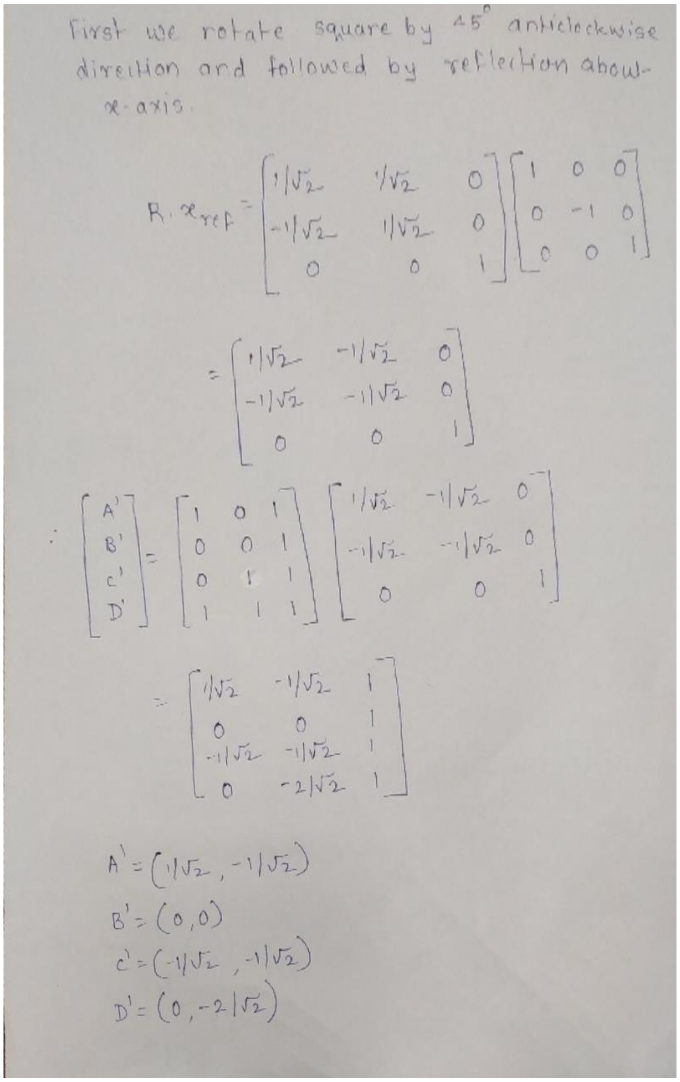
Plot integer now:

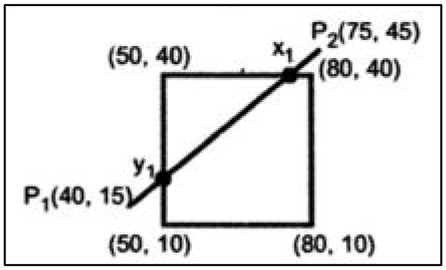
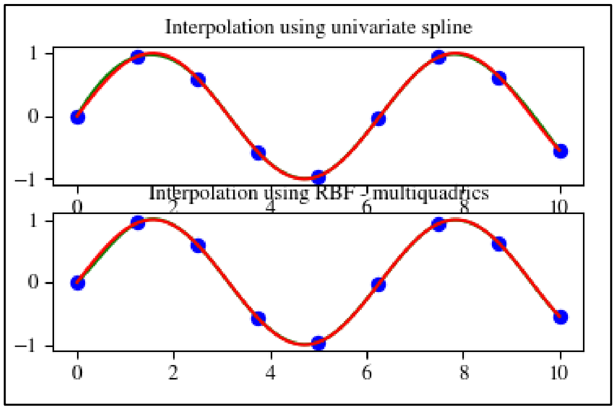
1. Plot (0,0), $x=x+\Delta X=0.5+4/6=1.167$, $y=y+ \Delta Y=0.5+1=1.5$
2.Plot (1,1), $x=x+\Delta X=1.167+4/6=1.833$ $y=y+ \Delta Y=1.5+1=2.5$
3.Plot (1,2), $x=x+\Delta X=1.833+4/6=2.5$ $y=y+ \Delta Y=2.5+1=3.5$
4.Plot (2,3), $x=x+\Delta X=2.5+4/6=3.167$ $y=y+ \Delta Y=3.5+1=4.5$
5.Plot (3,4), $x=x+\Delta X=3.167+4/6=3.833$ $y=y+ \Delta Y=4.5+1=5.5$
6.Plot (3,5), $x=x+\Delta X=3.833+4/6=4.5$ $y=y+ \Delta Y=5.5+1=6.5$

Tabulating the results of each iteration in the step 7 we get,

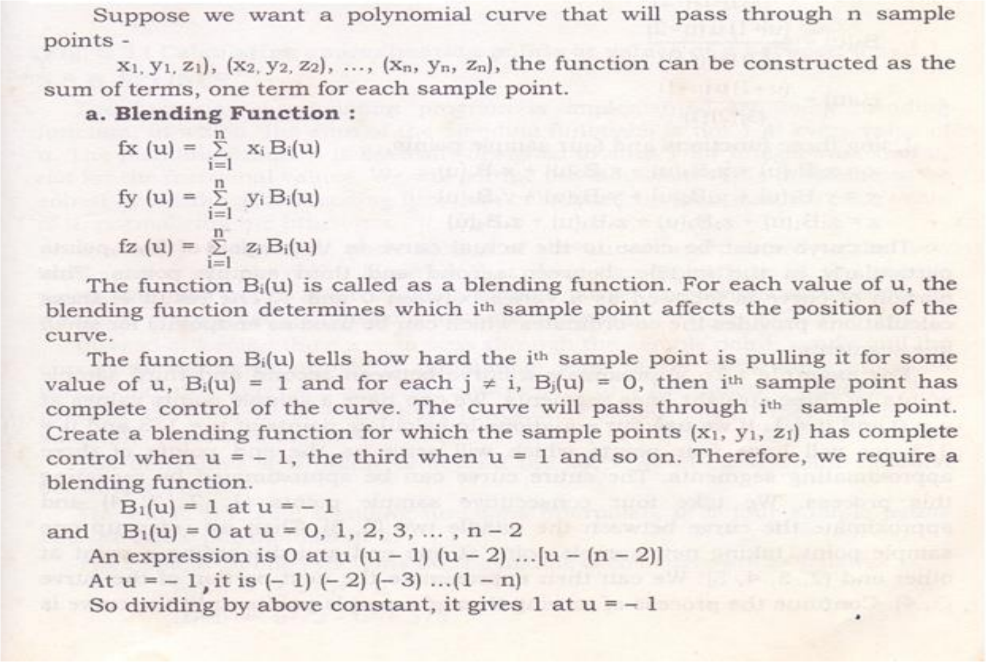
i	Plot	x	y
		0.5	0.5
1	(0,0)	1.167	1.5
2	(1,1)	1.833	2.5
3	(1,2)	2.5	3.5
4	(2,3)	3.167	4.5
5	(3,4)	3.833	5.5
6	(3,5)	4.5	6.5

		 **Fig. 2.2** • The results are plotted as shown in the Fig. 2.2. It shows that the rasterized line lies to both sides of the actual line, i.e. the algorithm is orientation dependent.	
	c	**Consider a square A(1,0), B(0,0), C(0,1), D(1,1). Rotate the square by 45^0 anti-clockwise direction followed by reflection about X-axis.**	**4 M**
	Ans	 Given, A(1,0) B(0,0) C(0,1) D(1,1) $R = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$ Here, $\theta = 45^0$ $R = \begin{bmatrix} \cos 45 & \sin 45 & 0 \\ -\sin 45 & \cos 45 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ $= \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} & 0 \\ -1/\sqrt{2} & 1/\sqrt{2} & 0 \\ 0 & 0 & 1 \end{bmatrix}$ Matrix Reflection about x-axis :- $x_{ref} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	Rotation + Reflection Matrix 1 M final Result= 3 M

		 First we rotate square by 45° anticlockwise direition and followed by reflection abow x-axis. $R.x_{ref} = \begin{bmatrix}1/\sqrt{2} & 1/\sqrt{2} & 0\\ -1/\sqrt{2} & 1/\sqrt{2} & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0 & 0\\ 0 & -1 & 0\\ 0 & 0 & 1\end{bmatrix}$ $= \begin{bmatrix}1/\sqrt{2} & -1/\sqrt{2} & 0\\ -1/\sqrt{2} & -1/\sqrt{2} & 0\\ 0 & 0 & 1\end{bmatrix}$ $\therefore \begin{bmatrix}A'\\ B'\\ C'\\ D'\end{bmatrix} = \begin{bmatrix}1 & 0 & 1\\ 0 & 0 & 1\\ 0 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}\begin{bmatrix}1/\sqrt{2} & -1/\sqrt{2} & 0\\ -1/\sqrt{2} & -1/\sqrt{2} & 0\\ 0 & 0 & 1\end{bmatrix}$ $= \begin{bmatrix}1/\sqrt{2} & -1/\sqrt{2} & 1\\ 0 & 0 & 1\\ -1/\sqrt{2} & -1/\sqrt{2} & 1\\ 0 & -2/\sqrt{2} & 1\end{bmatrix}$ $A' = (1/\sqrt{2}, -1/\sqrt{2})$ $B' = (0,0)$ $C' = (-1/\sqrt{2}, -1/\sqrt{2})$ $D' = (0, -2/\sqrt{2})$	
	d	**Use Cohen-Sutherland outcode algorithm to clip line PI (40, 15) -- P2 (75. 45) against a window A (50, 10), B (80, 10). C(80, 40) & D(50,40).**	**4 M**
	Ans	P1 (40, 15) - P2 (75, 45) Wxi = 50 Wy2 = 40 Wx2 = 80 Wy2 = 10 Point Endcode ANDing P1 0001 0000 (Partially visible) P2 0000 $y_1 = m(x_L – x) + y = \frac{6}{7}(50-40)+15$ $m = \frac{45-15}{75-40}$	Proper result 4 M

		= 23.57 $x_1 = \frac{1}{m}(y_T - y) + x = \frac{7}{6}(40\text{-}50)+40 = 69.16$ $y_2 = m(x_R - x) + y = \frac{6}{7}(80\text{-}40)+15 = 49.28$ $x_2 = \frac{1}{m}(y_B - y) + x = \frac{7}{6}(10\text{-}15)+40 = 34.16$ Hence:  	
	e	**What is interpolation? Describe the Lagrangian Interpolation method.**	**4 M**
	Ans	Specify a spline curve by giving a set of coordinate positions, called control points, which indicates the general shape of the curve These, control points are then fitted with piecewise continuous parametric polynomial functions in one of two ways. When polynomial sections are fitted so that the curve passes through each control point, the resulting curve is said to interpolate the set of control points. On the other hand, when the polynomials are fitted to the general control-point path without necessarily passing through any control point, the resulting curve is said to approximate the set of control points interpolation curves are commonly used to digitize drawings or to specify animation paths. Approximation curves are primarily used as design tools to structure object surfaces an approximation spline surface credited for a design application. Straight lines connect the control-point positions above the surface  	Definition- 1 M Description of Lagrangian method- 3 M

Lagrangian Interpolation Method:



Suppose we want a polynomial curve that will pass through n sample points -

x_1, y_1, z_1), (x_2, y_2, z_2), …, (x_n, y_n, z_n), the function can be constructed as the sum of terms, one term for each sample point.

a. Blending Function :

$$fx\ (u) = \sum_{i=1}^{n} x_i B_i(u)$$

$$fy\ (u) = \sum_{i=1}^{n} y_i B_i(u)$$

$$fz\ (u) = \sum_{i=1}^{n} z_i B_i(u)$$

The function $B_i(u)$ is called as a blending function. For each value of u, the blending function determines which i^{th} sample point affects the position of the curve.

The function $B_i(u)$ tells how hard the i^{th} sample point is pulling it for some value of u, $B_i(u) = 1$ and for each $j \neq i$, $B_j(u) = 0$, then i^{th} sample point has complete control of the curve. The curve will pass through i^{th} sample point. Create a blending function for which the sample points (x_1, y_1, z_1) has complete control when $u = -1$, the third when $u = 1$ and so on. Therefore, we require a blending function.

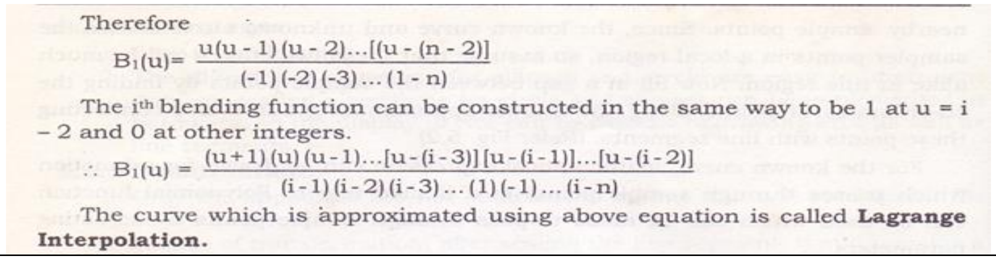
$B_1(u) = 1$ at $u = -1$

and $B_1(u) = 0$ at $u = 0, 1, 2, 3, \ldots, n-2$

An expression is 0 at $u\ (u-1)\ (u-2) \ldots [u-(n-2)]$

At $u = -1$, it is $(-1)\ (-2)\ (-3) \ldots (1-n)$

So dividing by above constant, it gives 1 at $u = -1$



Therefore

$$B_1(u) = \frac{u(u-1)\,(u-2)\ldots[(u-(n-2)]}{(-1)\,(-2)\,(-3)\ldots(1-n)}$$

The i^{th} blending function can be constructed in the same way to be 1 at u = i – 2 and 0 at other integers.

$$\therefore\ B_1(u) = \frac{(u+1)\,(u)\,(u-1)\ldots[u-(i-3)]\,[u-(i-1)]\ldots[u-(i-2)]}{(i-1)\,(i-2)\,(i-3)\ldots(1)\,(-1)\ldots(i-n)}$$

The curve which is approximated using above equation is called **Lagrange Interpolation.**

5		**Attempt any TWO of the following :**	**12 M**
	a	**Consider the line from (5,5) to (13,9). Use the Bresenham's algorithm to rasterize the line.**	**6 M**
	Ans	**Bresenham Line Drawing Calculator** By putting x1,x2 and y1,y2 Value it Show The Result In Step By Step order, and Result Brief Calculation Which Is Calculated by Bresenham Line Drawing Algorithm. Bresenham Line Drawing Algorithm display result in tables. Starting Points is x1,y1 and Ending points is x2,y2. **Preliminary Calculations:**	Remark: Preliminary Calculations 2 M; Step wise plot 4 M

x1 = 5 | y1 = 5 | & | x2 = 13 | y2 = 9

Calculation	Result

dx = abs(x1 - x2)	8 = abs(5 - 13)
dy = abs(y1 - y2)	4 = abs(5 - 9)
p = 2 * (dy - dx)	-8 = 2 * (4 - 8)
ELSE	x = x1 | y = y1 | end = x2
	x = 5 | y = 5 | end = 13

Stepwise Plot:

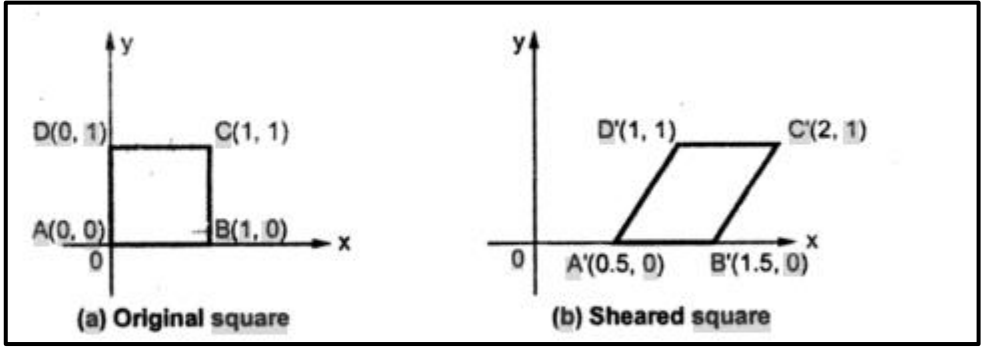
STEP	**while(x < end)**	**x = x + 1**	**if(p < 0) { p = p + 2 * dy } else{ p = p + 2 * (dy - dx) }**	**OUTPUT**
1	6 < 13	6 = 5 + 1	IF 0 = -8 + 2 * 4	x = 6 | y = 5
2	7 < 13	7 = 6 + 1	ELSE -8 = 0 + 2 * (4 - 8)	x = 7 | y = 6
3	8 < 13	8 = 7 + 1	IF 0 = -8 + 2 * 4	x = 8 | y = 6
4	9 < 13	9 = 8 +	ELSE -8 = 0 + 2 * (4 -	x = 9 | y = 7

		1	8)	
5	10 < 13	10 = 9 + 1	IF 0 = -8 + 2 * 4	x = 10 | y = 7
6	11 < 13	11 = 10 + 1	ELSE -8 = 0 + 2 * (4 - 8)	x = 11 | y = 8
7	12 < 13	12 = 11 + 1	IF 0 = -8 + 2 * 4	x = 12 | y = 8
8	13 < 13	13 = 12 + 1	ELSE -8 = 0 + 2 * (4 - 8)	x = 13 | y = 9

	b	**Apply the shearing transformation to square with A(0,0), B(1,0), C(1,1), D(0,1) as given below.** **(i) Shear Parameter value of 0.5 relative to the line yref = -1.** **(ii) Shear Parameter value of 0.5 relative to the line xref = -1**.	**6 M**
	Ans	We can represent the given square ABCD, in matrix form, using homogeneous coordinates of vertices as: $\begin{bmatrix} A & 0 & 0 & 1 \\ B & 1 & 0 & 1 \\ C & 1 & 1 & 1 \\ D & 0 & 1 & 1 \end{bmatrix}$ i) Here $Sh_x = 0.5$ and $y_{ref} = -1$ $\begin{bmatrix} A` \\ B` \\ C` \\ D` \end{bmatrix} = \begin{bmatrix} A \\ B \\ C \\ D \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 \\ Shx & 1 & 0 \\ -Shx * yref & 0 & 1 \end{bmatrix}$ $= \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 \\ 0.5 & 1 & 0 \\ 0.5 & 0 & 1 \end{bmatrix}$	Each sub problem – 3 M

$$= \begin{bmatrix} 0.5 & 0 & 1 \\ 1.5 & 0 & 1 \\ 2 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Shearing Transformation Result:-



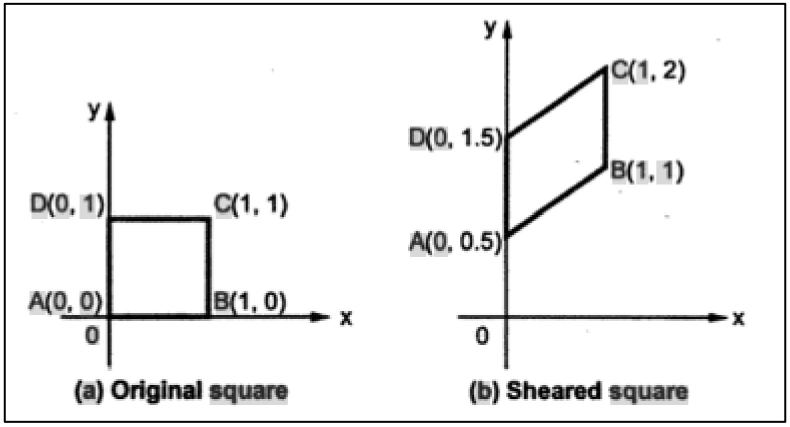
ii) Here $Sh_y = 0.5$ and $x_{ref} = -1$

$$\begin{bmatrix} A` \\ B` \\ C` \\ D` \end{bmatrix} = \begin{bmatrix} A \\ B \\ C \\ D \end{bmatrix} * \begin{bmatrix} 1 & Shy & 0 \\ 0 & 1 & 0 \\ 0 & -Shy * xref & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0.5 & 0 \\ 0 & 1 & 0 \\ 0 & 0.5 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0.5 & 1 \\ 1 & 1 & 1 \\ 1 & 2 & 1 \\ 0 & 1.5 & 1 \end{bmatrix}$$

Shearing Transformation Result:-



	c	**Write a program in 'C' to generate Hilbert's curve.**	**6 M**
	Ans	Correct logic – 6 Marks)	

```
#include <stdio.h>
#include <stdlib.h>
#include <graphics.h>
#include <math.h>

void move(int j,int h,int &x,int &y)
{
        if(j==1)
        y-=h;
        else if(j==2)
        x+=h;
        else if(j==3)
        y+=h;
        else if(j==4)
        x-=h;
        lineto(x,y);
}

void hilbert(int r,int d,int l,int u,int i,int h,int &x,int &y)
{
        if(i>0)
        {
                i--;
                hilbert(d,r,u,l,i,h,x,y);
                move(r,h,x,y);
                hilbert(r,d,l,u,i,h,x,y);
                move(d,h,x,y);
                hilbert(r,d,l,u,i,h,x,y);
                move(l,h,x,y);
                hilbert(u,l,d,r,i,h,x,y);
        }
}

int main()
{
        int n,x1,y1;
        int x0=50,y0=150,x,y,h=10,r=2,d=3,l=4,u=1;

        printf)"\nGive the value of n: ");
        scanf("%d",&n);
        x=x0;y=y0;
        int gm,gd=DETECT;
        initgraph(&gd,&gm,NULL);
        moveto(x,y);
        hilbert(r,d,l,u,n,h,x,y);
```

```
        delay(10000);
        closegraph();
        return 0;
}
```

6		**Attempt any TWO of the following**	**12 M**
	a	**Write a Program in 'C' for DDA Circle drawing algorithm**	**6 M**
	Ans		**Correct Program 6 marks**

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<math.h>
void main()
{
 int gdriver=DETECT,gmode,errorcode,tmp,i=1,rds;
 float st_x,st_y,x1,x2,y1,y2,ep;
 initgraph(&gdriver,&gmode,"C:\\TC\\BGI");
      printf("Enter Radius:");
 scanf("%d",&rds);
 while(rds>pow(2,i))
  i++;
   ep=1/pow(2,i);
 x1=rds;  y1=0;
 st_x=rds; st_y=0;
 do
 { x2=x1+(y1*ep);
  y2=y1-(x2*ep);
  putpixel(x2+200,y2+200,10);
  x1=x2;
  y1=y2;
 }while((y1-st_y)<ep || (st_x-x1)>ep);
    getch();
}
```

	b	**Perform a 45^o rotation of triangle A(0,0), B(1,1), C(5,2)** **(i) About the origin (ii) About P(-1,-1)**	**6 M**
	Ans	About the Origin: -	Each Sub problem – 3 M

<table>
<tr><td></td><td></td><td>

Solution: We can represent the given triangle, in matrix form, using homogeneous coordinates of the vertices:

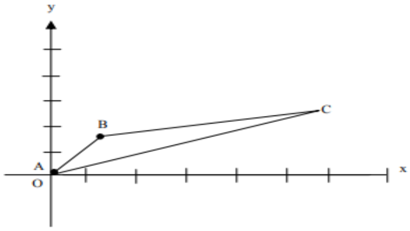
$$[ABC]=\begin{pmatrix} A & 0 & 0 & 1 \\ B & 1 & 1 & 1 \\ C & 5 & 2 & 1 \end{pmatrix}$$

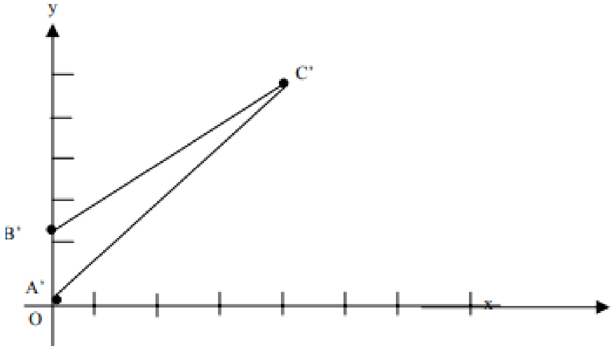
The matrix of rotation is: $R_\theta = R_{45^0} = \begin{pmatrix} \cos45^0 & \sin45^0 & 0 \\ -\sin45^0 & \cos45^0 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} \sqrt{2}/2 & \sqrt{2}/2 & 0 \\ -\sqrt{2}/2 & \sqrt{2}/2 & 0 \\ 0 & 0 & 1 \end{pmatrix}$

So the new coordinates A'B'C' of the rotated triangle ABC can be found as:

$$[A'B'C']=[ABC].\ R_{45^\circ} = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 1 & 1 \\ 5 & 2 & 1 \end{bmatrix} \begin{bmatrix} \sqrt{2}/2 & \sqrt{2}/2 & 0 \\ -\sqrt{2}/2 & \sqrt{2}/2 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & \sqrt{2} & 1 \\ 3\sqrt{2}/2 & 7\sqrt{2}/2 & 1 \end{bmatrix}$$

Thus A'=(0,0), B'=(0,√2), C'=(3√2/2,7√2/2)





</td><td></td></tr>
<tr><td></td><td>c</td><td>

**Apply the Liang-Barsky algorithm to the line with co-ordinate (30,60) & (60,25) against the window:
(Xmin, Ymin) = (10.10) & (Xmax,Ymax) = (50,50)**

</td><td>6 M</td></tr>
</table>

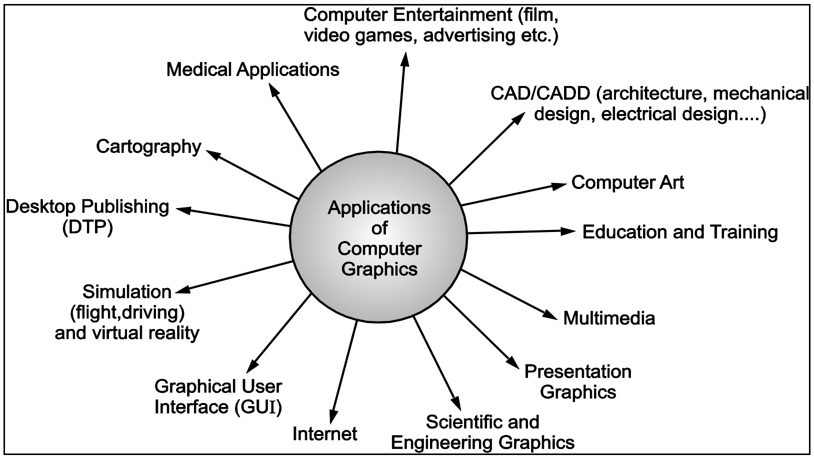
	Ans	**Given:** (X_{min},Y_{min})=(10,10) and (X_{max},Y_{max})=(50,50) P1 (30, 60) and P2 = (60, 25) **Solution:** Set Umin = 0 and Umax = 1 ULeft= q1 / p1 = X1 – Xmin / - Δ X = 30 – 10 / - (60 - 30) = 20 / - 30 = -0.67 URight= q2 / p2 = Xmax – X1/ ΔX = 50 – 30 / (60 - 30) = 20 / 30 = 0.67 UBottom= q3 / p3 = Y1 – Ymin / - ΔY = 60 – 10 / - (25 – 60) = 50 / 35 = 1.43 UTop= q4 / p4 = Ymax – Y1 / ΔY = 50 – 60 / (25 - 60) = -10 / - 35 = 0.29 Since ULeft=−0.57which is less than Umin. Therefore we ignore it. Similarly UBottom=1.43 which is greater than Umax. So we ignore it. URight=Umin = 0.67 (Entering) UTop=Umax = 0.29 (Exiting) We have UTop= 0.29 and URight= 0.67 Q – P = (ΔX, ΔY) = (30, -35) Since Umin>Umax, there is no line segment to draw.	Remark: Calculation of each side 1 M; Decision of displaying line coordinates with justification 2 M

WINTER – 19 EXAMINATION

Subject Name: Computer Graphics **Model Answer** **Subject Code: 22318**

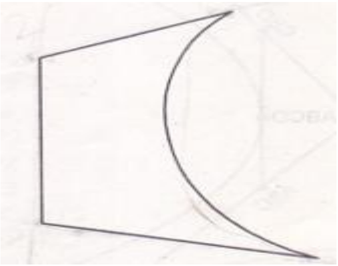
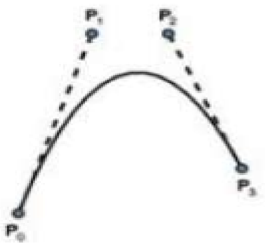
Important Instructions to examiners:

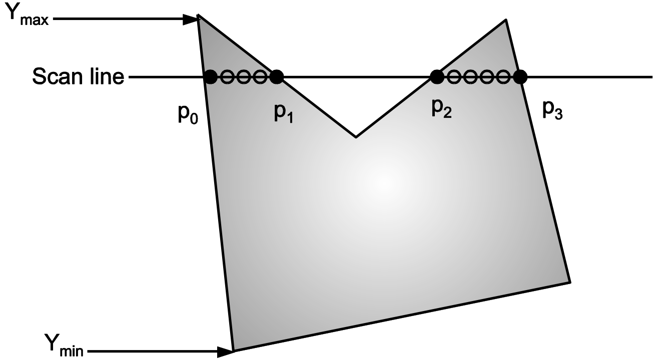
1) The answers should be examined by key words and not as word-to-word as given in the model answer scheme.
2) The model answer and the answer written by candidate may vary but the examiner may try to assess the understanding level of the candidate.
3) The language errors such as grammatical, spelling errors should not be given more Importance (Not applicable for subject English and Communication Skills.
4) While assessing figures, examiner may give credit for principal components indicated in the figure. The figures drawn by candidate and model answer may vary. The examiner may give credit for any equivalent figure drawn.
5) Credits may be given step wise for numerical problems. In some cases, the assumed constant values may vary and there may be some difference in the candidate's answers and model answer.
6) In case of some questions credit may be given by judgement on part of examiner of relevant answer based on candidate's understanding.
7) For programming language papers, credit may be given to any other program based on equivalent concept.

Q. No.	Sub Q. N.	Answer	Marking Scheme
1		**Attempt any FIVE of the following :**	**10 M**
	a	**Give two applications of computer graphics.**	**2 M**
	Ans	 - **DTP (Desktop Publishing)** Used for common paper and book publishing are sometimes used to create graphics for point of sale displays, presentations, infographics, brochures, business cards, promotional items, trade show exhibits, retail package designs and outdoor signs. - **Graphical User Interface (GUI)**	**Any two applications : 2 M**

The use of pictures, images, icons, pop-up menus, graphical objects helps in creating a user friendly environment where working is easy and pleasant, using computer graphics we can create such an atmosphere where everything can be automated and anyone can get the desired action performed in an easy fashion.

- **Computer-Aided Design**
  Designing of buildings, automobile, aircraft is done with the help of computer aided drawing, this helps in providing minute details to the drawing and producing more accurate and sharp drawings with better specifications.
- **Computer-Aided Learning (Cal)**
  Computer Aided Learning (CAL) is the application of computers as an integral part of the learning system for learning and teaching process.
- **Animations**
  Used for creating motion pictures, music video, television shows, **cartoon animation** films.
- **Computer Art**
  Using computer graphics we can create fine and commercial art which include animation packages, paint packages.
- **Entertainment**
  Computer graphics finds a major part of its utility in the movie industry and game industry. Used for creating motion pictures, music video, television shows, cartoon animation films.
- **Education and training**
  Computer generated models are extremely useful for teaching huge number of concepts and fundamentals in an easy to understand and learn manner.
- **Image processing**
  Various kinds of photographs or images require editing in order to be used in different places.
- **Medical Applications**
  The use of **computer graphics** for **medical** diagnosis has provided an extraordinary ability to visualize measure and evaluate structures in a non-intrusive manner.
- **Presentation and Business Graphics**
  For the preparation of reports or summarizing the financial, statistical, mathematical, scientific, economic data for research reports, managerial reports, moreover creation of bar graphs, pie charts, time chart, can be done using the tools present in computer graphics.
- **Simulation and Virtual Reality**
  A simulation can also take the form of a computer-graphics image that represents dynamic processes in an animated sequence. Virtual reality applications are applications that make use of virtual

		reality (VR). VR is an immersive sensory experience that digitally simulates a remote environment.	
	b	**List / name two line drawing algorithms.**	**2 M**
	Ans	• Digital Differential Analyzer (DDA) Algorithm • Bresenham's Line Drawing Algorithm	**Any two names: 2 M**
	c	**Explain the need of homogeneous co-ordinates matrix.**	**2 M**
	Ans	Homogeneous coordinates are used extensively in computer vision and graphics because they allow common operations such as translation, rotation, scaling and perspective projection to be implemented as matrix operations.	**Explanation: 2 M**
	d	**Define polygon clipping.**	**2 M**
	Ans	A set of connected lines are considered as polygon; polygons are clipped based on the window and the portion which is inside the window is kept as it is and the outside portions are clipped. **OR** Polygon clipping is removal of part of an object outside a polygon.	**Any suitable definition: 2 M**
	e	**Draw Cubic Bezier Curve.**	**2 M**
	Ans	 **OR** 	**Any similar type of curve: 2 M**
	f	**Define Bitmap Graphics.**	**2 M**
	Ans	• A **bitmap** is an image or shape of any kind-a picture, a text character, a photo-that's composed of a collection of tiny individual dots. A wild landscape on your screen is a **bitmapped** graphic, or simply a bitmap. • It is a pixel based image, not scalable and size of image is high.	**Any suitable definition: 2 M**
	g	**List various character generation methods.**	**2 M**
	Ans	• Stroke Method • Bitmap Method • Starburst Method	**Any two names: 2 M**
2		**Attempt any THREE of the following :**	**12 M**
	a	**Write short note on Augmented Reality.**	**4 M**
	Ans	• Augmented reality (AR) is made up of the word "augment" which means to make something great by adding something to it. • **A***ugmented* **R***eality* is a type of virtual reality that aims to duplicate the world's environment in a computer. • Augmented reality is a method by which we can alter our real world by adding some digital elements to it.	**Explanation: 4M**

		• This is done by superimposing a digital image on the person's current view thus it enhances the experience of reality. • Virtual reality makes a virtual environment and puts the user in it whereas Augmented reality just adds the virtual components into the user's real-world view. • For Augmented reality you only need a modern smartphone then you can easily download an AR app like Google's "**just a line**" and try this technology. • One of the most popular ways AR has infiltrated everyday life is through mobile games. In 2016, the AR game "Pokémon Go" became a sensation worldwide, with over 100 million estimated users at its peak, according to CNET. • The goal of Augmented Reality is to create a system in which the user cannot tell the difference between the real world and the virtual augmentation of it. Today Augmented Reality is used in entertainment, military training, engineering design, robotics, manufacturing and other industries.	
	b	**Explain scan line algorithm of polygon clipping.**	**4 M**
	Ans	• For each scan line crossing a polygon, the area-fill algorithm locates the intersection points of the scan line with the polygon edges. • These intersection points are then sorted from left to right, and the corresponding frame-buffer positions between each intersection pair are set to the specified fill color. • Scan line algorithm works by intersecting scan line with polygon edges and fills the polygon between pairs of intersections. The following steps depict how this algorithm works. **Step 1 :** Find out the Y_{min} and Y_{max} from the given polygon.  • Step 2 : ScanLine intersects with each edge of the polygon from Ymin to Ymax. Name each intersection point of the polygon. As per the Fig. 2.21 shown, they are named as p0, p1, p2, p3. • Step 3 : Sort the intersection point in the increasing order of X coordinate i.e. (p0, p1), (p1, p2), and (p2, p3).	**Algorithm: 4 M**

- Step 4 : Fill all those pair of coordinates that are inside polygons and ignore the alternate pairs.

c **Write 2D and 3D scaling matrix.** **4 M**

Ans **2D Scaling**

2D matrix: 2 M,
3D matrix: 2 M

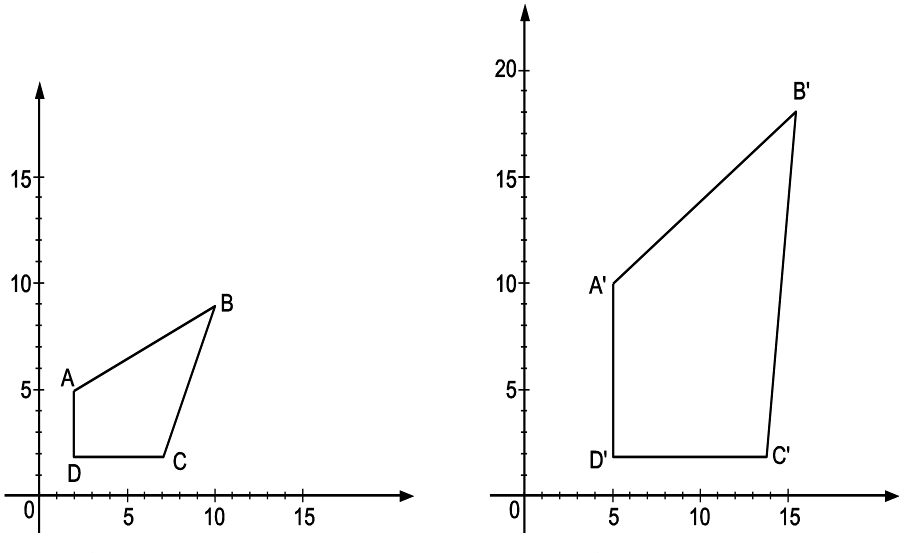
- Scaling means to change the size of object. This change can either be positive or negative.
- To change the size of an object, scaling transformation is used. In the scaling process, you either expand or compress the dimensions of the object.
- Scaling can be achieved by multiplying the original co-ordinates of the object with the scaling factor to get the desired result.
- Let us assume that the original co-ordinates are (X, Y), the scaling factors are (S_X, S_Y), and the produced co-ordinates are (X', Y'). This can be mathematically represented as shown below:
  - $X' = X \cdot S_X$ and $Y' = Y \cdot S_Y$
- The scaling factor S_X, S_Y scales the object in X and Y direction respectively. The above equations can also be represented in matrix form as below:

$$\begin{bmatrix} X' \\ Y' \end{bmatrix} = \begin{bmatrix} X \\ Y \end{bmatrix} \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix}$$

OR

$$P' = P \cdot S$$

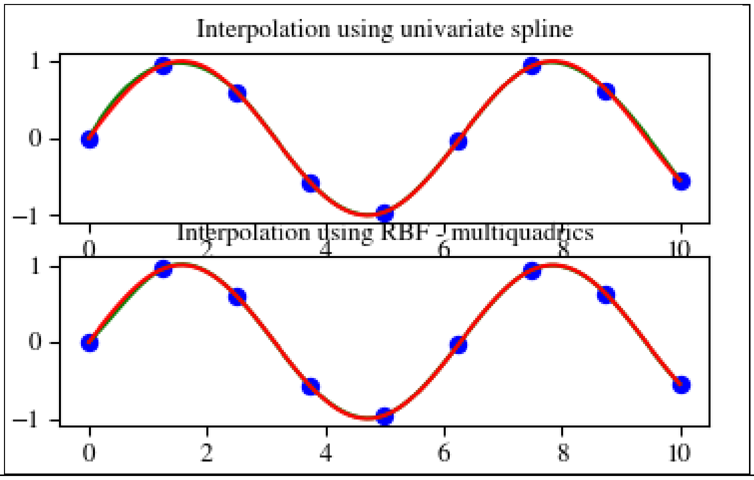
- Where, S is the scaling matrix.
- The scaling process is shown in the Fig

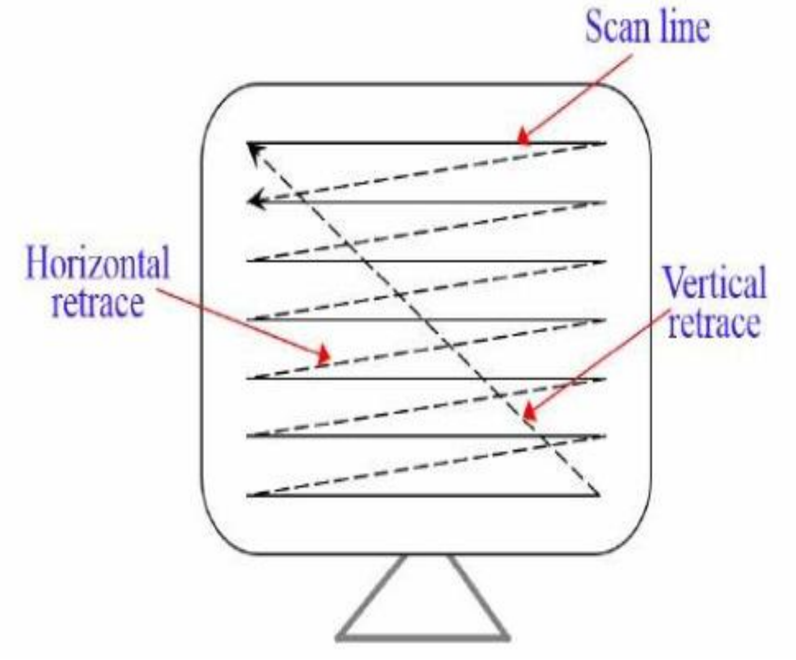


(a) Before Scaling **(b) After Scaling**

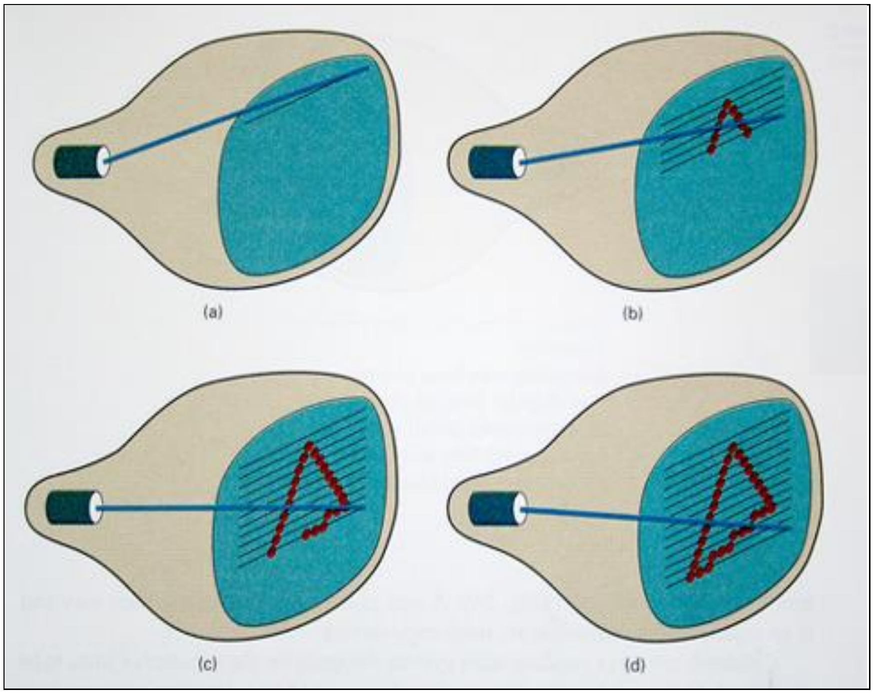
3D Scaling Matrix

		$$S = \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$ It specifies three co-ordinates with their own scaling factors. If scale factors, Sx = Sy = Sz = S > 1 then the scaling is called as magnification. Sx = Sy = Sz = S < 1 then the scaling is called as reduction. Therefore, point after scaling with respect to origin can be calculated as, P=P . S	
	d	**Explain midpoint subdivision line clipping algorithm.**	**4 M**
	Ans	**Step 1:** Scan two end points for the line P1(x1, y1) and P2(x2, y2). **Step 2:** Scan corners for the window as (Wx1, Wy1) and (Wx2, Wy2). **Step 3:** Assign the region codes for endpoints P1 and P2 by initializing code with 0000. Bit 1 - if (x < Wx1) Bit 2 - if (x > Wx2) Bit 3 - if (y < Wy1) Bit 4 - if (y > Wy2) **Step 4:** Check for visibility of line P1, P2. • If region codes for both end points are zero then the line is visible, draw it and jump to step 6. • If region codes for end points are not zero and the logical Anding operation of them is also not zero then the line is invisible, reject it and jump to step 6. • If region codes for end points does not satisfies the condition in 4 (i) and 4 (ii) then line is partly visible. **Step5:** Find midpoint of line and divide it into two equal line segments and repeat steps 3 through 5 for both subdivided line segments until you get completely visible and completely invisible line segments. **Step 6:** Exit.	**Algorithm: 4 M**
	e	**Explain interpolation techniques in curve generation.**	**4 M**
	Ans	Specify a spline curve by giving a set of coordinate positions, called control points, which indicates the general shape of the curve These, control points are then fitted with piecewise continuous parametric polynomial functions in one of two ways. When polynomial sections are fitted so that the curve passes through each control point, the resulting curve is said to interpolate the set of control points. On the other hand, when the polynomials are fitted to the general control -point path without necessarily passing through any control point, the resulting curve is said to approximate the set of control points interpolation curves are commonly used to digitize drawings or to specify animation paths. Approximation curves are primarily used as design tools to	**Diagram: 2 M, Explanation: 2 M**

		structure object surfaces an approximation spline sur face credited for a design application. Straight lines connect the control -point positions above the surface.  	
3		**Attempt any THREE of the following :**	**12 M**
	a	**Explain with diagram the techniques of Raster Scan Display.**	**4 M**
	Ans	• The most common type of graphics monitor employing a CRT is the Raster-scan displays, based on television technology • JPG images are raster based. Light occurs when an electron beam stimulates a phosphor. • In Raster scan, the electron beam from electron gun is swept horizontally across the phosphor one row at time from top to bottom. • The electron beam sweeps back and forth from left to right across the screen. The beam is on, while it moves from left to right. The beam is off, when it moves back from right to left. This phenomenon is called the horizontal retrace. • As soon as the beam reaches the bottom of the screen, it is turned off and is rapidly retraced back to the top to start again. This is called the vertical retrace. • Raster scan displays maintain the steady image on the screen by repeating scanning of the same image. This process is known as refreshing of screen.	**Explanation: 2 M,** **Diagram: 2 M**



Raster Scan CRT

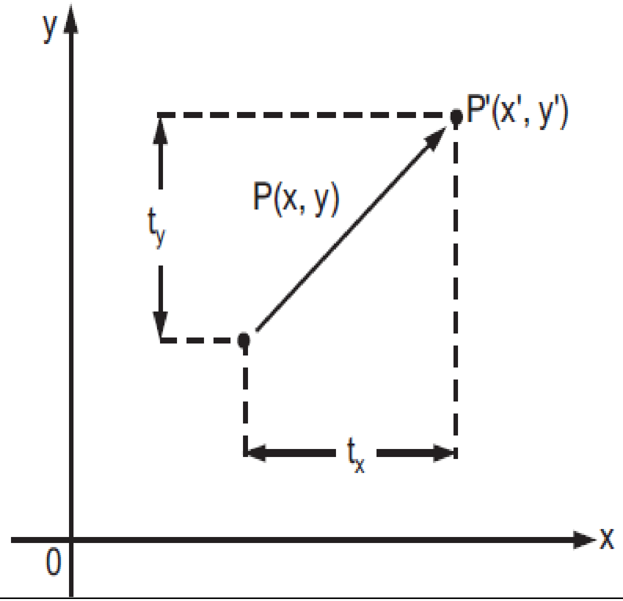


A Raster-Scan System Displays an Object as a Set of Discrete Points Across each Scan Line

- Typically, a graphics display consist of three components: frame buffer, video controller or display controller, and a TV screen or monitor.

		• Picture definition is stored in a memory area called the refresh buffer or frame buffer. This memory area holds the set of intensity values for all the screen points. The stored intensity values are then retrieved from frame buffer and painted on the screenone row at a time. Each screen point is referred as Pixel orpel. Each pixel on the screen can be specified by it row and column number. • Intensity range for pixel position depends on capability of the raster system. In black and white system, the point on screen is either on or off. Only one bit is needed to control the intensity of the screen. In case of color systems, 2 bits are requiredOne to represent ON (1), another one is OFF (0). • Refreshing on raster scan is carried out at the rate of 60 to 80 frames per seconds. The video or display controller has direct access to memory locations in the frame buffer. It is responsible for retrieving data from the frame buffer and passing it to the display device. It reads bytes of data from frame buffer and converts 0's and 1's in one line into its corresponding video signals and this is called a scan line. If the intensity is one (1) then controller sends a signal to display a dot in the corresponding position on the screen. If the intensity is zero (0) then no dot is displayed.	
	b	**Write procedure to fill polygon with flood fill.**	**4 M**
	Ans		**Correct procedure: 4 M**

```
flood_fill(x,y,old_color,new_color)

{

  if(getpixel(x,y) = old_color)

{

 putpixel(x,y,new_color);

flood_fill(x+1,y,old_color, new_color);

flood_fill(x-1,y,old_color, new_color);

flood_fill(x,y+1,old_color, new_color);

flood_fill(x,y-1,old_color, new_color);

flood_fill(x+1,y+1,old_color, new_color);

flood_fill(x-1,y-1,old_color, new_color);

flood_fill(x+1,y-1,old_color, new_color);

flood_fill(x-1,y+1,old_color, new_color);
```

		} }	
	c	**Explain 2D transformations with its types.**	**4 M**
	Ans	A transformation is a function that maps every position (x, y) into a new position (x', y'). Instead of applying the transformation function to every point in every line that makes up the object, we simply apply the function to the object vertices and then draw new lines between the resulting new endpoints. **Basic Transformations:** **1)Translation** **2)Scaling** **3)Rotation** **1)Translation:** • A translation is applied to an object by repositioning it along a straight-line path from one coordinate location to another. • Translation refers to the shifting (moving) of a point to some other place, whose distance with regard to the present point is known. • Translation can be defined as "the process of repositioning an object along a straight line path from one co-ordinate location to new co-ordinate location." • A translation moves an object to a different position on the screen. You can translate a point in 2D by adding translation coordinate (tx, ty) to the original coordinate (X, Y) to get the new coordinate (X', Y')  	**2D transformation: 1 M,** **Types: 1 M each**

From the above Fig. you can write that:

$X' = X + tx$
$Y' = Y + ty$

The pair (tx, ty) is called the translation vector or shift vector. The above equations can also be represented using the column vectors.

P = [X] [Y] p'
= [X] [Y] T = [tx] [ty]

We can write it as,
$P' = P + T$

Rotation

- Rotation as the name suggests is to rotate a point about an axis. The axis can be any of the co-ordinates or simply any other specified line also.
- In rotation, we rotate the object at particular angle θ (theta) from its origin. From the following figure, we can see that the point P(X, Y) is located at angle φ from the horizontal X coordinate with distance r from the origin.
- Let us, suppose you want to rotate it at the angle θ. After rotating it to a new location, you will get a new point P' (X', Y').

Using standard trigonometric the original coordinate of point P(X, Y) can be represented as:

$X = r \cos \phi$ (1)
$Y = r \sin \phi$ (2)

Same way we can represent the point P' (X', Y') as:

$x' = r \cos (\phi + \theta) = r \cos \phi \cos \theta - r \sin \phi \sin \theta$ (3)
$y' = r \sin (\phi + \theta) = r \cos \phi \sin \theta + r \sin \phi \cos \theta$ (4)

Substituting equation (1) and (2) in (3) and (4) respectively, we will get

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

Representing the above equation in matrix form,

$$[X'\ Y'] = [X'\ Y']\begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}$$

OR

$$P' = P \cdot R$$

Where, R is the rotation matrix

$$R = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}$$

The rotation angle can be positive and negative.

Scaling:

Scaling means to change the size of object. This change can either be positive or negative.

To change the size of an object, scaling transformation is used. In the scaling process, you either expand or compress the dimensions of the object.

Scaling can be achieved by multiplying the original co-ordinates of the object with the scaling factor to get the desired result.

Let us assume that the original co-ordinates are (X, Y), the scaling factors are (SX, SY), and the produced co-ordinates are (X', Y'). This can be mathematically represented as shown below:

X' = X .SX and Y' = Y .SY

The scaling factor SX, SY scales the object in X and Y direction respectively.

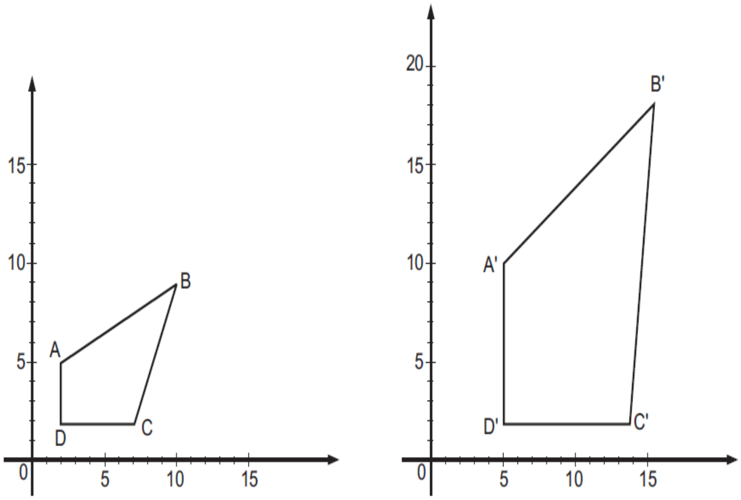
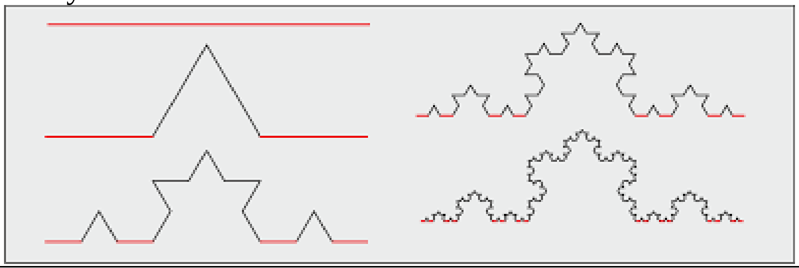
The above equations can also be represented in matrix form as below:

$$\begin{bmatrix} X' \\ Y' \end{bmatrix} = \begin{bmatrix} X \\ Y \end{bmatrix}\begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix}$$

OR

$$P' = P \cdot S$$

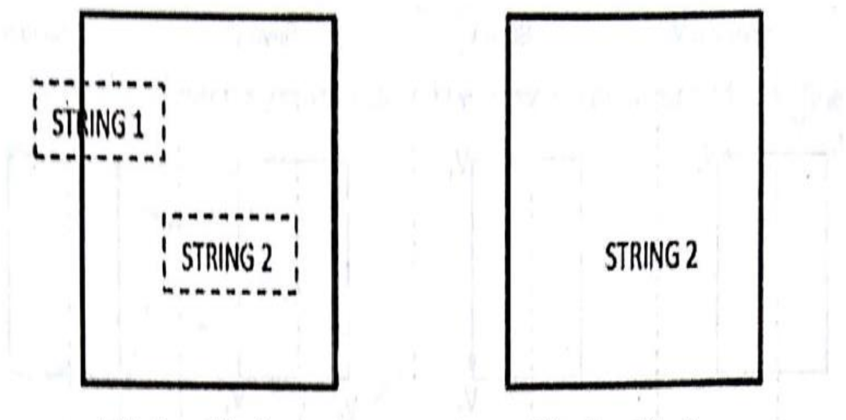
Where, S is the scaling matrix.

		 (a) Before Scaling (b) After Scaling If we provide values less than 1 to the scaling factor S, then we can reduce the size of the object. If we provide values greater than 1, then we can increase the size of the object.	
	d	**Explain Koch curve with diagram**.	**4 M**
	Ans	**Koch Curve: -** In Koch curve, begin at a line segment. Divide it into third and replace the center by the two adjacent sides of an equilateral triangle as shown below. This will give the curve which starts and ends at same place as the original segment but is built of 4 equal length segments, with each 1/3rd of the original length. So the new curve has 4/3 the length of original segments. Repeat same process for each of the 4 segment which will give curve more wiggles and its length become 16/9 times the original. Suppose repeating the replacements indefinitely, since each repetition increases the length by a factor of 4/3, the length of the curve will be infinite but it is folded in lots of tiny 	**Explanation: 2 M, Diagram: 2 M**
	e	**Explain Text Clipping.**	**4 M**
	Ans	Many techniques are used to provide text clipping in a computer graphics. It depends on the methods used to generate characters and the requirements of a particular application. There are three methods for text clipping which are listed below – 1) All or none string clipping	**Explanation: 4 M**

2) All or none character clipping

3) Text clipping

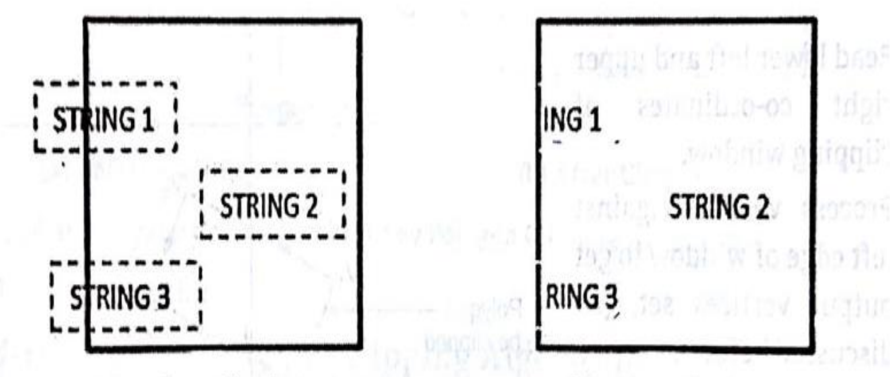
The following figure shows all or none string clipping –



(a) Before Clipping (b) After Clipping

In all or none string clipping method, either we keep the entire string or we reject entire string based on the clipping window. As shown in the above figure, STRING2 is entirely inside the clipping window so we keep it and STRING1 being only partially inside the window, we reject.

The following figure shows all or none character clipping –



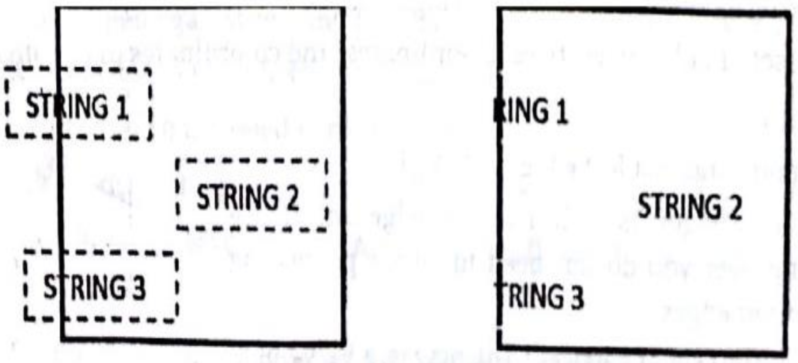
(a) Before Clipping (b) After Clipping

This clipping method is based on characters rather than entire string. In this method if the string is entirely inside the clipping window, then we keep it. If it is partially outside the window, then −

You reject only the portion of the string being outside If the character is on the boundary of the clipping window, then we discard that entire character and keep the rest string.

<table>
<tr><td></td><td></td><td>

The following figure shows text clipping –



(a) Before Clipping (b) After Clipping

This clipping method is based on characters rather than the entire string. In this method if the string is entirely inside the clipping window, then we keep it. If it is partially outside the window, then you reject only the portion of string being outside. If the character is on the boundary of the clipping window, then we discard only that portion of character that is outside of the clipping window.

</td><td></td></tr>
<tr><td></td><td></td><td></td><td></td></tr>
<tr><td>4</td><td></td><td>Attempt any THREE of the following :</td><td>12 M</td></tr>
<tr><td></td><td>a</td><td>Explain inside and outside test for polygon.</td><td>4 M</td></tr>
<tr><td></td><td>Ans</td><td>

This method is also known as counting number method. While filling an object, we often need to identify whether particular point is inside the object or outside it.

There are two methods by which we can identify whether particular point is inside an object or outside namely, Odd-Even Rule, and Non-zero winding number rule.

1. Odd-Even Rule:
In this technique, we count the edge crossing along the line from any point (x, y) to infinity. If the number of interactions is odd then the point (x, y) is an interior point. If the number of interactions is even then point (x, y) is an exterior point.
Here is the example to give you the clear idea,

</td><td>Explanation: 4 M</td></tr>
</table>

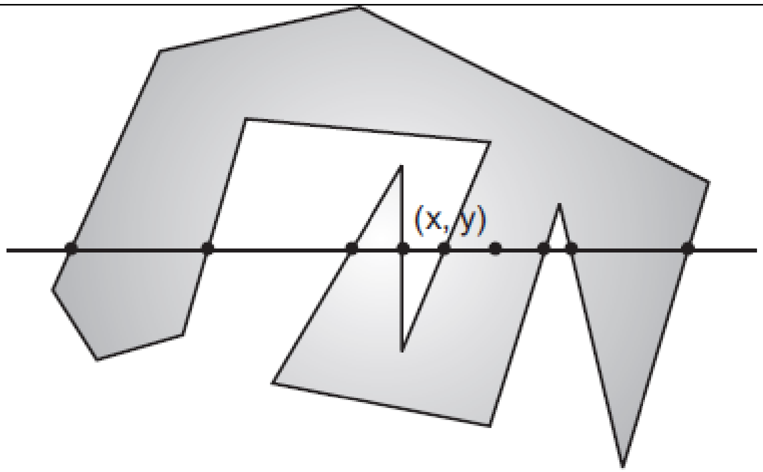


Fig a: Odd-Even Rule

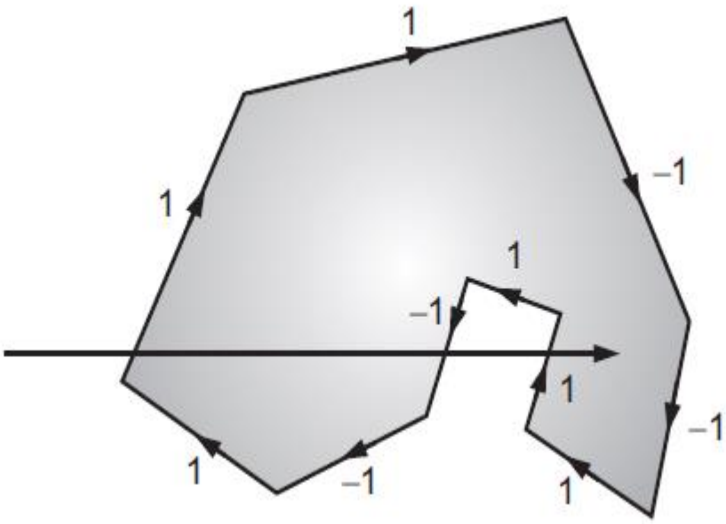
From the Fig., we can see that from the point (x, y), the number of interactions point on the left side is 5 and on the right side is 3. So the total number of interaction point is 8, which is odd. Hence, the point is considered within the object.

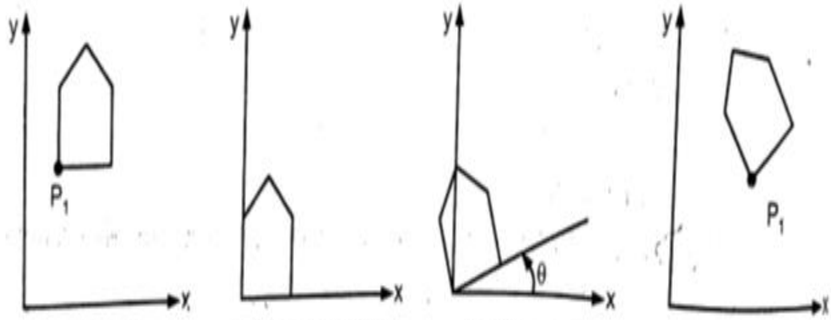
2. Non-zero Winding Number Rule:

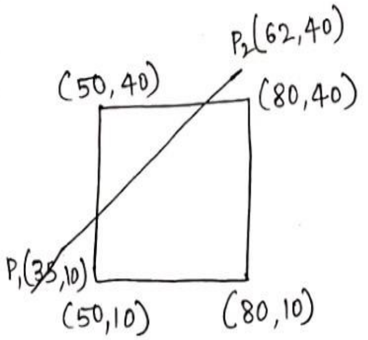
This method is also used with the simple polygons to test the given point is interior or not. It can be simply understood with the help of a pin and a rubber band.

Fix up the pin on one of the edge of the polygon and tie-up the rubber band in it and then stretch the rubber band along the edges of the polygon.

When all the edges of the polygon are covered by the rubber band, check out the pin which has been fixed up at the point to be test. If we find at least one wind at the point consider it within the polygon, else we can say that the point is not inside the polygon.

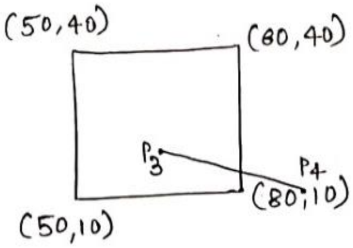
		 **Fig b: Non-zero Winding Number Rule:** In another alternative method, give directions to all the edges of the polygon. Draw a scan line from the point to be test towards the left most of X direction. Given the value 1 to all the edges which are going to upward direction and all other – 1 as direction values. Check the edge direction values from which the scan line is passing and sum up them. If the total sum of this direction value is non-zero, then this point to be tested is an interior point, otherwise it is an exterior point. In the above figure, we sum up the direction values from which the scan line is passing then the total is 1 – 1 + 1 = 1; which is non-zero. So the point is said to be an interior point.	
	b	**Explain composite transformation over arbitrary point.**	**4 M**
	Ans	To do rotation of an object about any selected arbitrary point P1(x1 ,y1), following sequence of operations shall be performed. **1. Translate:** Translate an object so that arbitrary point P1 is moved to coordinate origin. **2. Rotate:** Rotate object about origin. **3. Translate:** Translate object so that arbitrary point P1 is moved back to the its original position. Rotate about point P1(x1,y1). 1) Translate P1 to origin. 2) Rotate 3) Translate back to P1. Equation for this composite transformation matrix form is as follows:	**Explanation: 2 M,** **Matrix: 1 M,** **Diagram: 1M**

		$P' = T(x_1, y_1) \cdot R(\theta) \cdot T(-x_1, -y_1)$ $P' = \begin{bmatrix}1 & 0 & x_1\\0 & 1 & y_1\\0 & 0 & 1\end{bmatrix} \cdot \begin{bmatrix}\cos\theta & -\sin\theta & 0\\ \sin\theta & \cos\theta & 0\\0 & 0 & 1\end{bmatrix} \cdot \begin{bmatrix}1 & 0 & -x_1\\0 & 1 & -y_1\\0 & 0 & 1\end{bmatrix} \cdot \begin{bmatrix}x\\y\\1\end{bmatrix}$ Here (x1,y1) are coordinates of point P1 and hence are translation factors tx and ty; we want to move P1 to origin , x1 and y1 are x and y distances to P1and hence it is translation factor. $P' = \begin{bmatrix}\cos\theta & -\sin\theta & x_1(1-\cos\theta)+y_1\sin\theta\\ \sin\theta & \cos\theta & y_1(1-\cos\theta)-x_1\sin\theta\\0 & 0 & 1\end{bmatrix} \cdot \begin{bmatrix}x\\y\\1\end{bmatrix}$ It is demonstrated in following figure:  (a) Original house (b) After translation of P_1 to origin (c) After rotation (d) After translation to original P_1 	
	c	**Use the Cohen Sutherland algorithm to clip two lines P1(35,10)-P2(65,40) and P3(65,20)-P4(95,10) against a window A(50,10), B(80,10), C(80,40) and D(50,40).**	**4 M**

		Line 1:- $P_1=(35,10)$ $wx_1=50$ $wy_1=40$ $P_2=(62,40)$ $wx_2=80$ $wy_2=10$  $P_1=0001$ $P_2=1000$ ANDing 0000 Line is partially visible. $m=\frac{40-10}{62-35}=\frac{30}{27}$ $y_1=m(x_L-x)+y$ $=\frac{30}{27}(50-35)+10$ $=\frac{30}{27}(15)+10=26.66$ $x_1=\frac{1}{m}(y_T-y)+x$ $=\frac{27}{30}(40-10)+35$ $=\frac{27}{30}(30)+35=62$ $y_2=m(x_R-x)+y$ $=\frac{30}{27}(80-35)+10$ $=\frac{30}{27}(45)+10=60$ $x_2=\frac{1}{m}(y_B-y)+x$ $=\frac{27}{30}(10-10)+35$ $=35$		**First Line 2M,Second Line 2M**

Line 2 :-

$P_3(65,20)$ $P_4(95,10)$



P_3 0 0 0 0

P_4 0 0 1 0

ANDing 0 0 0 0

Line is partially visible.

$$m = \frac{10-20}{95-65} = \frac{-10}{30} = \frac{-1}{3}$$

$$y_1' = m(x_L - x) + y$$
$$= \frac{-1}{3}(50-65)+20$$
$$= \frac{-1}{3}(-15)+20 = 25$$

$$x_1' = \frac{1}{m}(y_T - y) + x$$
$$= -3(40-20)+65$$
$$= -3(20)+65 = -60+65 = 5$$

$$y_2' = m(x_R - x) + y$$
$$= \frac{-1}{3}(80-65)+20$$
$$= -5+20 = 15$$

$$x_2' = \frac{1}{m}(y_B - y) + x$$
$$= -3(10-20)+65$$
$$= -3(-10)+65$$
$$= 95$$

d **Write DDA Arc generation algorithm.** **4 M**

Ans

Correct algorithm: 4 M

```
1. Read the centre of curvature, say(x0,y0)
2. Read the arc angle, say Θ
3. Read the starting point of the arc, say(x,y)
4. Calculate dΘ
dΘ=min(0.01,1/3.2*(|x-x0|+|y-y0|)))
5. Initialize angle = 0
6. while (angle < Θ)
   do
{
 Plot(x,y)
x=x-(y-y0) *dΘ
y=y-(x-x0) *dΘ
Angle =Angle + dΘ
}
7. stop
```

5 **Attempt any TWO of the following :** **12 M**

a **Use Bresenham's line drawing algorithm to rasterize line from (6,5) to (15,10).** **6 M**

Ans

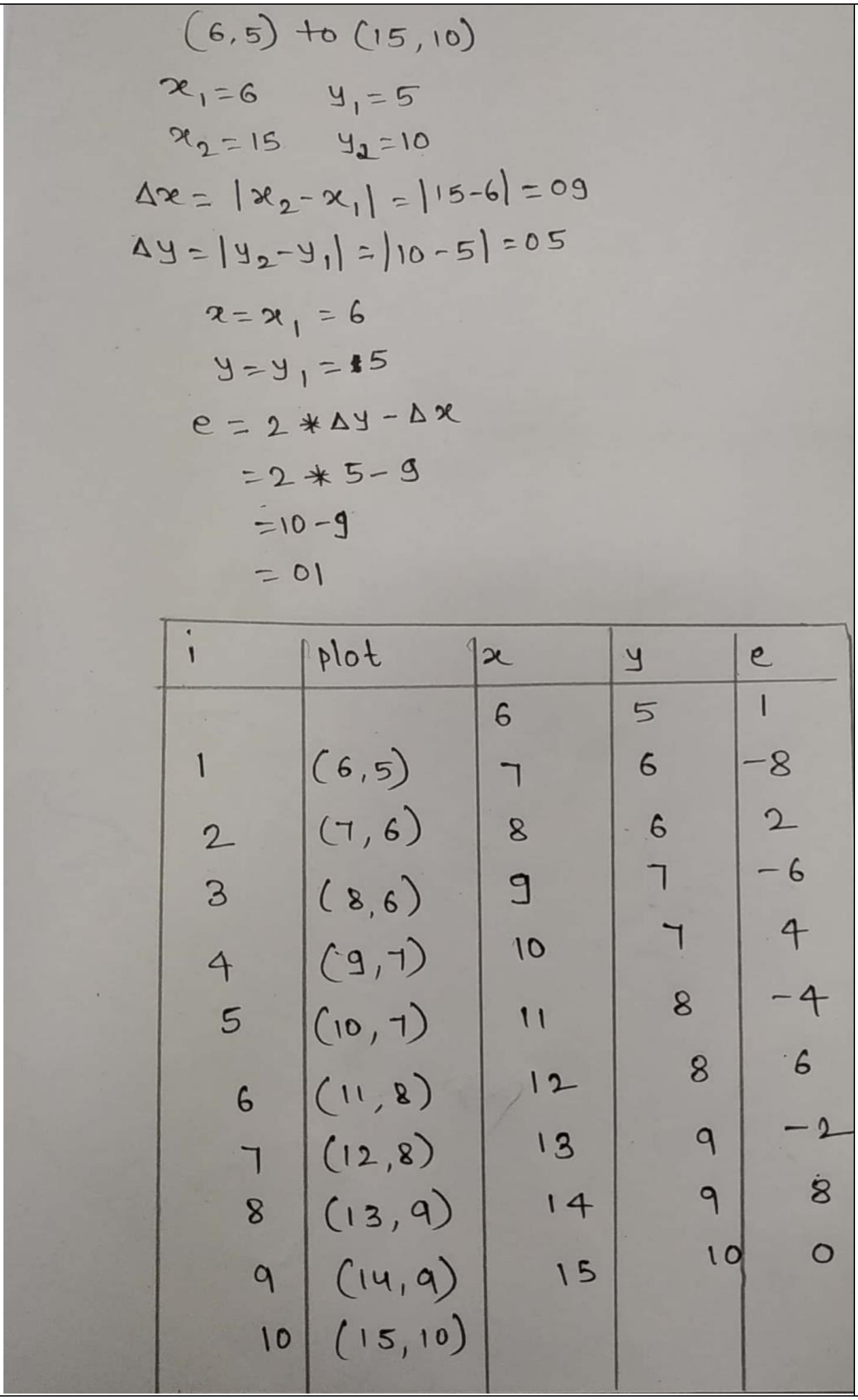
Calculations of dx, dy and p: 2 M; Calculations of steps: 4 M

x1 = 6 | y1 = 5 | & | x2 = 15 | y2 = 10	
Calculation	Result
dx = abs(x1 - x2)	9 = abs(6 - 15)
dy = abs(y1 - y2)	5 = abs(5 - 10)
p = 2 * (dy - dx)	-8 = 2 * (5 - 9)
ELSE	x = x1 | y = y1 | end = x2
	x = 6 | y = 5 | end = 15

STEP	while(x < end)	x = x + 1	if(p < 0) { p = p + 2 * dy } else{ p = p + 2 * (dy - dx) }	OUTPUT
1	7 < 15	7 = 6 + 1	IF 2 = -8 + 2 * 5	x = 7 | y = 5
2	8 < 15	8 = 7 + 1	ELSE -6 = 2 + 2 * (5 - 9)	x = 8 | y = 6

3	9 < 15	9 = 8 + 1	IF 4 = -6 + 2 * 5	x = 9 | y = 6
4	10 < 15	10 = 9 + 1	ELSE -4 = 4 + 2 * (5 - 9)	x = 10 | y = 7
5	11 < 15	11 = 10 + 1	IF 6 = -4 + 2 * 5	x = 11 | y = 7
6	12 < 15	12 = 11 + 1	ELSE -2 = 6 + 2 * (5 - 9)	x = 12 | y = 8
7	13 < 15	13 = 12 + 1	IF 8 = -2 + 2 * 5	x = 13 | y = 8
8	14 < 15	14 = 13 + 1	ELSE 0 = 8 + 2 * (5 - 9)	x = 14 | y = 9
9	15 < 15	15 = 14 + 1	ELSE -8 = 0 + 2 * (5 - 9)	x = 15 | y = 10

OR



(6, 5) to (15, 10)

$x_1 = 6 \quad y_1 = 5$

$x_2 = 15 \quad y_2 = 10$

$\Delta x = |x_2 - x_1| = |15-6| = 09$

$\Delta y = |y_2 - y_1| = |10-5| = 05$

$x = x_1 = 6$

$y = y_1 = 5$

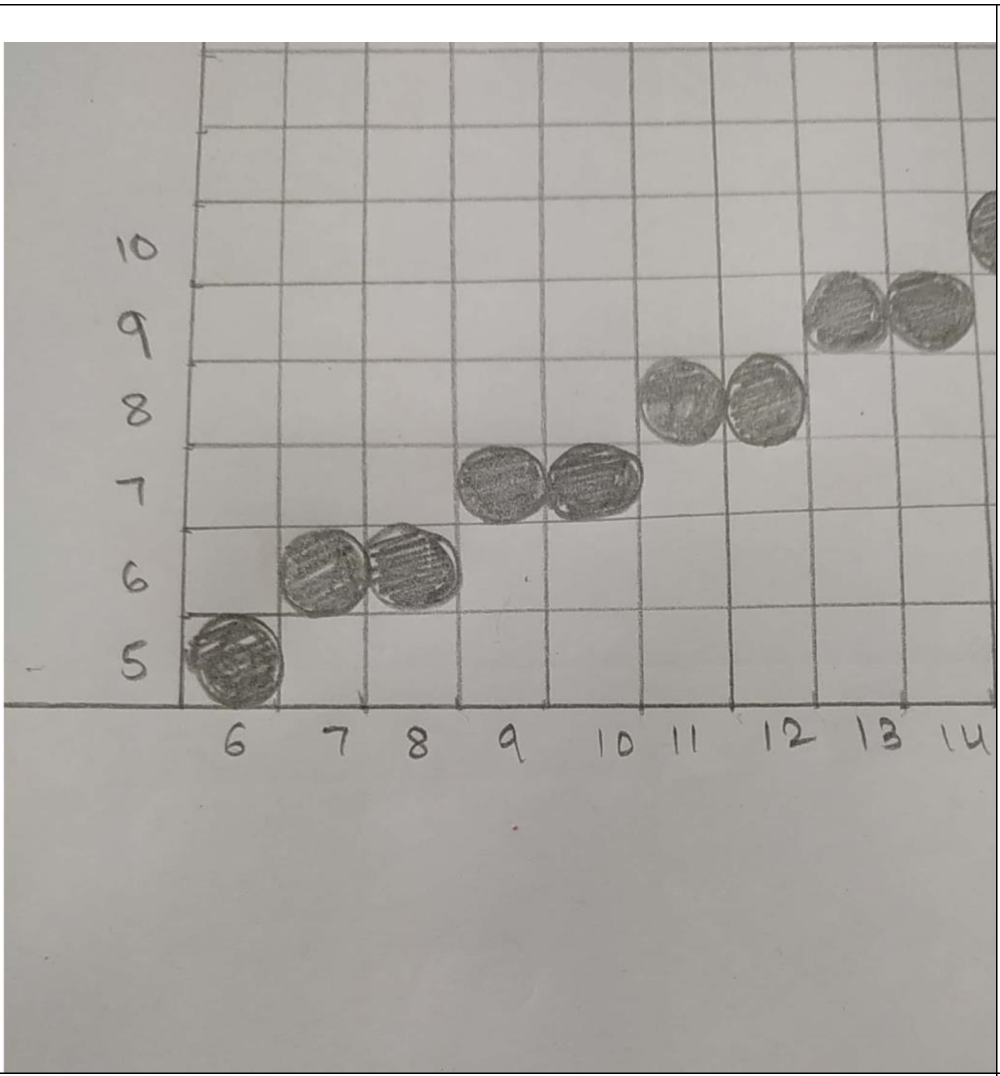
$e = 2 * \Delta y - \Delta x$

$= 2 * 5 - 9$

$= 10 - 9$

$= 01$

i	plot	x	y	e
		6	5	1
1	(6,5)	7	6	-8
2	(7,6)	8	6	2
3	(8,6)	9	7	-6
4	(9,7)	10	7	4
5	(10,7)	11	8	-4
6	(11,8)	12	8	6
7	(12,8)	13	9	-2
8	(13,9)	14	9	8
9	(14,9)	15	10	0
10	(15,10)			

			
	b	**Find the transformation of triangle A(1,0) B(0,1) C(1,1) by** **i. Rotating 30° about the origin** **ii. Translating one unit x and y direction and then rotate 45° about origin.**	**6 M**
	Ans	**i. Rotating 30° about the origin.** We can represent the given triangle, in matrix form, using homogeneous coordinates of the vertices: [ABC] = $\begin{bmatrix} A & 1 & 0 & 1 \\ B & 0 & 1 & 1 \\ C & 1 & 1 & 1 \end{bmatrix}$ The rotation matrix is Rθ = R$_{30}$ = $\begin{bmatrix} cos30 & sin30 & 0 \\ -sin30 & cos30 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \frac{\sqrt{3}}{2} & \frac{1}{2} & 0 \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} & 0 \\ 0 & 0 & 1 \end{bmatrix}$ So the new coordinates A'B'C' of the rotated triangle ABC can be found as:	**Rotation 30°: 3 M; Translation, rotation 45° and retranslation: 3 M**

A'B'C']=[ABC]. R30° $=\begin{bmatrix}1 & 0 & 1\\ 0 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}\begin{bmatrix}\frac{\sqrt{3}}{2} & \frac{1}{2} & 0\\ -\frac{1}{2} & \frac{\sqrt{3}}{2} & 0\\ 0 & 0 & 1\end{bmatrix}$ =

$$\begin{bmatrix}0.866025 & -0.5 & 0\\ 0.5 & 0.866025 & 0\\ 1.36603 & 0.366025 & 1\end{bmatrix}$$

ii. Translating one unit x and y direction and then rotate 45° about origin.

Points A, B and C, θ= 45° and about points are (1,1)

$t_x = 1;$

$t_y = 1;$

for rotation about arbitrary point we followed sequence of operation as

Translation -> Rotation about origin -> Retranslation

$$\begin{bmatrix}x'\\ y'\\ z'\end{bmatrix} = \begin{bmatrix}1 & 0 & 1\\ 0 & 1 & 1\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}\frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0\\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0 & -1\\ 0 & 1 & -1\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0 & 1\\ 0 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

$$\begin{bmatrix}x'\\ y'\\ z'\end{bmatrix} = \begin{bmatrix}\frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 1\\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 1\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0 & -1\\ 0 & 1 & -1\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0 & 1\\ 0 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

$$\begin{bmatrix}x'\\ y'\\ z'\end{bmatrix} = \begin{bmatrix}-\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 1\\ -\frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & \frac{2}{\sqrt{2}}+1\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0 & 1\\ 0 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

$$\begin{bmatrix}x'\\ y'\\ z'\end{bmatrix} = \begin{bmatrix}-\frac{1}{\sqrt{2}}+1 & \frac{1}{\sqrt{2}}+1 & 1\\ -\frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & \frac{4}{\sqrt{2}}+3\\ 1 & 1 & 1\end{bmatrix}$$

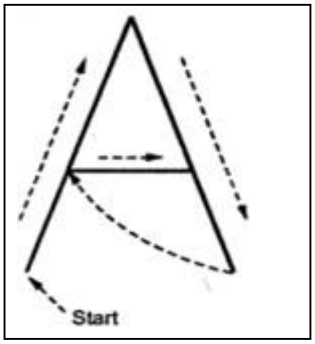
	c	**Write C program for Hilbert's Curve.**	**6 M**
	Ans	`#include <stdio.h>`	**Correct program: 6 M**

```
#define N 32
#define K 3
#define MAX N * K

typedefstruct{int x; int y; } point;

void rot(int n, point *p, int rx, int ry){
int t;
if(!ry){
if(rx == 1){
            p->x = n - 1 - p->x;
            p->y = n - 1 - p->y;
}
        t = p->x;
        p->x = p->y;
        p->y = t;
}
}

void d2pt(int n, int d, point *p){
int s = 1, t = d, rx, ry;
    p->x = 0;
    p->y = 0;
while(s < n){
        rx = 1&(t / 2);
        ry = 1&(t ^ rx);
        rot(s, p, rx, ry);
        p->x += s * rx;
        p->y += s * ry;
        t /= 4;
        s *= 2;
}
}

int main(){
int d, x, y, cx, cy, px, py;
char pts[MAX][MAX];
    point curr, prev;
for(x = 0; x < MAX; ++x)
```

```
for(y = 0; y < MAX; ++y) pts[x][y] = ' ';
    prev.x = prev.y = 0;
    pts[0][0] = '.';
for(d = 1; d < N * N; ++d){
        d2pt(N, d, &curr);
        cx = curr.x * K;
        cy = curr.y * K;
        px = prev.x * K;
        py = prev.y * K;
        pts[cx][cy] = '.';
if(cx == px ){
if(py < cy)
for(y = py + 1; y < cy; ++y) pts[cx][y] = '|';
else
for(y = cy + 1; y < py; ++y) pts[cx][y] = '|';
}
else{
if(px < cx)
for(x = px + 1; x < cx; ++x) pts[x][cy] = '_';
else
for(x = cx + 1; x < px; ++x) pts[x][cy] = '_';
}
        prev = curr;
}
for(x = 0; x < MAX; ++x){
for(y = 0; y < MAX; ++y)printf("%c", pts[y][x]);
printf("\n");
}
return0;
}
```

6		**Attempt any TWO of the following :**	**12 M**
	a	**Explain character generation methods:** **i. Stroke** **ii. Starburst** **iii. Bitmap**	**6 M**
	Ans	1) STROKE METHOD • Stroke method is based on natural method of text written by human being. In this method graph is drawing in the form of line by line.	**Each Method of character generation: 2 M**

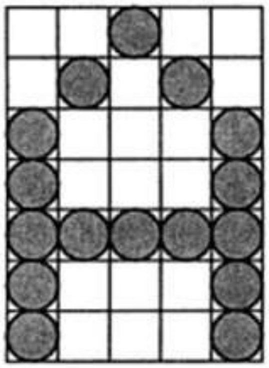
- Line drawing algorithm DDA follows this method for line drawing.
- This method uses small line segments to generate a character. The small series of line segments are drawn like a stroke of pen to form a character.
- We can build our own stroke method character generator by calls to the line drawing algorithm. Here it is necessary to decide which line segments are needed for each character and then drawing these segments using line drawing algorithm.



2)BITMAP METHOD

- Bitmap method is a called dot-matrix method as the name suggests this method use array of bits for generating a character. These dots are the points for array whose size is fixed.
- In bit matrix method when the dots is stored in the form of array the value 1 in array represent the characters i.e. where the dots appear we represent that position with numerical value 1 and the value where dots are not present is represented by 0 in array.
- It is also called dot matrix because in this method characters are represented by an array of dots in the matrix form. It is a two dimensional array having columns and rows.

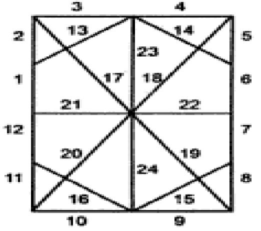
A 5x7 array is commonly used to represent characters. However 7x9 and 9x13 arrays are also used. Higher resolution devices such as inkjet printer or laser printer may use character arrays that are over 100x100.



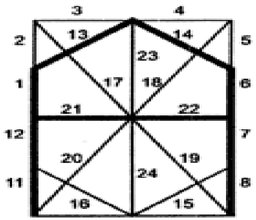
Character A in 5 × 7 dot matrix format

3) Starbust method:
In this method a fix pattern of line segments are used to generate characters. Out of these 24 line segments, segments required to display for particular character are highlighted. This method of character generation is called starbust method because of its characteristic appearance
The starbust patterns for characters A and M. the patterns for particular characters are stored in the form of 24 bit code, each bit representing one line segment. The bit is set to one to highlight the line segment; otherwise it is set to zero. For example, 24-bit code for Character A is 0011 0000 0011 1100 1110 0001 and for
character M is 0000 0011 0000 1100 1111 0011.
This method of character generation has some disadvantages. They are

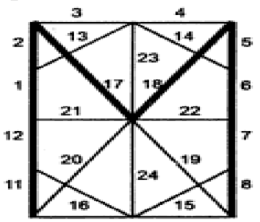
1. The 24-bits are required to represent a character. Hence more memory is required.
2. Requires code conversion software to display character from its 24-bitcode.
3. Character quality is poor. It is worst for curve shaped characters.



a) Star bust pattern of 24 line segments



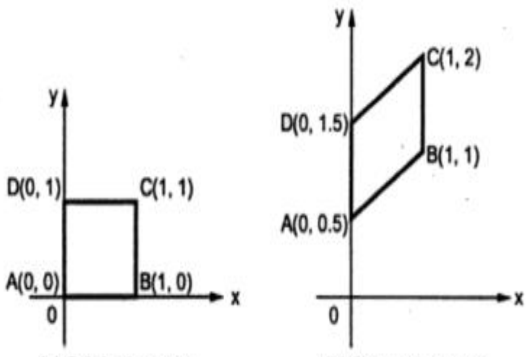
b) Star bust pattern for character A

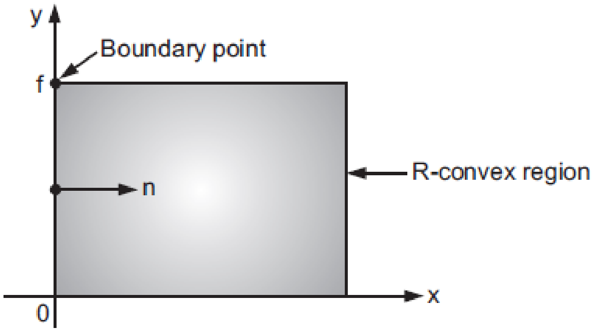


c) Star bust pattern for character M

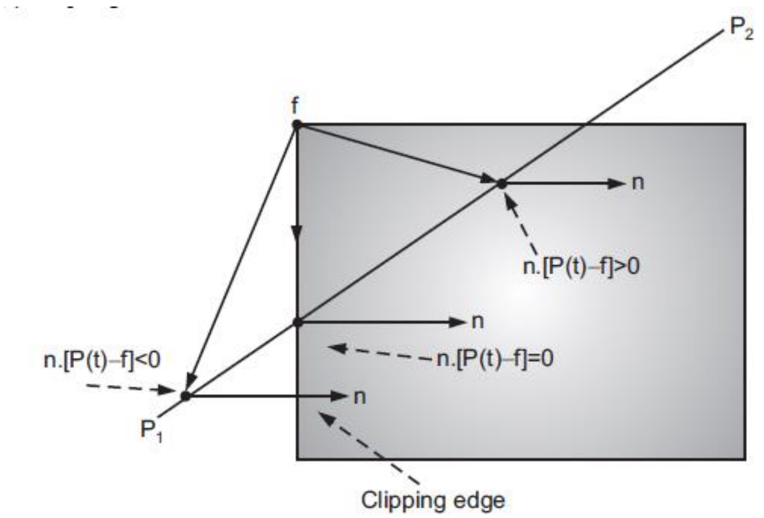
Character A : 0011 0000 0011 1100 11100001
Character M:0000 0011 0000 1100 11110011

	b	**Apply shearing transformation to square with A(0,0), B(1,0), C(1,1) and D(0,1) as shear parameter value of 0.5 relative to the line Y_{ref} = -1 and X_{ref}=-1.**	**6 M**

	Ans	a) Here $Sh_x = 0.5$ and $y_{ref} = -1$ $\begin{bmatrix}A'\\B'\\C'\\D'\end{bmatrix} = \begin{bmatrix}A\\B\\C\\D\end{bmatrix}\begin{bmatrix}1&0&0\\Sh_x&1&0\\-Sh_x\cdot y_{ref}&0&1\end{bmatrix}$ $= \begin{bmatrix}0&0&1\\1&0&1\\1&1&1\\0&1&1\end{bmatrix}\begin{bmatrix}1&0&0\\0.5&1&0\\0.5&0&1\end{bmatrix}$ $= \begin{bmatrix}0.5&0&1\\1.5&0&1\\2&1&1\\1&1&1\end{bmatrix}$ b) Here $Sh_y = 0.5$ and $x_{ref} = -1$ $\therefore \quad \begin{bmatrix}A'\\B'\\C'\\D'\end{bmatrix} = \begin{bmatrix}A\\B\\C\\D\end{bmatrix}\begin{bmatrix}1&Sh_y&0\\0&1&0\\0&-Sh_y\cdot x_{ref}&1\end{bmatrix}$ $= \begin{bmatrix}0&0&1\\1&0&1\\1&1&1\\0&1&1\end{bmatrix}\begin{bmatrix}1&0.5&0\\0&1&0\\0&0.5&1\end{bmatrix} = \begin{bmatrix}0&0.5&1\\1&1&1\\1&2&1\\0&1.5&1\end{bmatrix}$ It is important to note that shearing operations can be expressed as sequence of basic transformations. The sequence of basic transformations involve series of rotation and scaling transformations.  (a) Original square (b) Sheared square	**Shear in Yref=-1: 3 M;** **Shear in Xref = -1: 3 M**

	c	**Explain Cyrusblek line clipping algorithm.**	**6 M**
	Ans	**Cyrus Beck Line Clipping algorithm:** Cyrus Beck Line Clipping algorithm is used to clip 2D/3D lines against convex polygon/polyhedron. • Cyrus Beck Line clipping algorithm is actually, a parametric line-clipping algorithm. • The term parametric means that we require finding the value of the parameter t in the parametric representation of the line segment for the point at that the segment intersects the clipping edge. • Consider line segment P_1P_2. The parametric equation of line segment P_1P_2 is, $P(t) = P_1 + t(P_2 – P_1)$ … (1) Where, t value defines a point on the line going through P1 and P2. $0 <= t <= 1$ defines line segment between P1 and P2. If $t = 0$ then $P(0) = P_1$. If $t = 1$ then $P(1) = P_2$. • Consider a convex clipping region R, f is a boundary point of the convex region R and n is an inner normal for one of its boundaries as shown in Fig  **Convex region, boundary point and inner normal.** • Then we can distinguish in which region a point lie by looking at the value of the dot product $n.[P(t) – f]$, as shown in Fig. • If dot product is negative i.e., $n.[P(t) – f] < 0$ … (2) then the vector $P(t) – f$] is pointed away from the interior of R. • If dot product is zero i.e., $n.[P(t) – f] = 0$ … (3) then the vector $P(t) – f$] is pointed parallel to the plane containing f and perpendicular to the normal. • If dot product is positive i.e., $n.[P(t) – f] > 0$ … (4)	**Description of algorithm: 6 M** ****Cyrus Beck is considered instead of Cyrusblek**

then the vector P(t) – f] is pointed towards the interior of R.



Dot products for three points inside, outside and on the boundary of the clipping region

- As shown in Fig. if the point f lies in the boundary plane or edge for which n is the inner normal, then that point t on the line P(t) which satisfies,

  n.[P(t) – f] = 0 condition is the intersection of the line with the boundary edge.
- To get the formal statement of the Cyrus-Beck algorithm we substitute value of P(t) in equation 3.

  n . [P(t) – f] = n . [P1 + (P2 – P1)t – f] = 0 … (5)
- The relation should be applied for each boundary plane or edge of the window to get the intersection points. Thus in general form equation (5) can be written as,

  $n_i . [P_1 + (P_2 – P_1)t – f_i] = 0$ … (6)

where, i is edge number.

- Solving equation (6) we get,

  $n_i . [P1 – f_i] + n_i . (P2 – P1)t = 0$ … (7)
- Here the vector P2 – P1 defines the direction of the line. The direction of the line is important to correctly identify the visibility of the line. The vector P1 - fi is proportional to the distance from the end point of the line to the boundary point.
- Let us define,

  $D = P_2 – P_1$ as the direction of a line and

  $W_i = P_1 – f_i$ as weighting factor.
- Substituting newly defined variable D and W_i in Equation (7) we get,

  $n_i.W_i + (n_i . D)t = 0$ … (8)

$t = -(n_i.Wi)/(n_i.D)$ … (9)

where, $D \neq 0$ and i = 1, 2, 3 .....

• The equation (9) is used to obtain the value of t for the intersection of the line with each edge of the clipping window. We must select the proper value for t using following tips :

1. If t is outside the range 0<= t <= 1, then it can be ignored.
2. We know that, the line can intersect the convex window in at most two points , i.e. at two values of t. With equation (9) , there can be several values of t in the range of 0<= t <= 1 . We have to choose the largest lower limit and the smallest upper limit.
3. If $(n_i . D_i) > 0$ then equation (9) gives lower limit value for t and if $(n_i. D_i) < 0$ then equation (9) gives upper limit value for t.

Algorithm Cyrus Beck Line Clipping Algorithm:

```
Step 1 : Read end points of line $P_1$ and $P_2$.
Step 2 : Read vertex coordinates of clipping window.
Step 3 : Calculate $D = P_2 – P_1$.
Step 4 : Assign boundary point b with particular edge.
Step 5 : Find inner normal vector for corresponding edge.
Step 6 : Calculate D.n and $W = P_1 – b$
Step 7 : If D.n > 0
        $t_L = – (W.n)/(D.n)$
        else
        $t_U = – (W.n)/(D.n)$
        end if
Step 8 : Repeat steps 4 through 7 for each edge of clipping window.
Step 9 : Find maximum lower limit and minimum upper limit.
Step 10 : If maximum lower limit and minimum upper limit do not satisfy condition $0 \leq t \leq 1$ then ignore line.
Step 11 : Calculate intersection points by substituting values of maximum lower limit and minimum upper limit in parametric equation of line $P_1P_2$.
Step 12 : Draw line segment $P(t_L)$ to $P(t_U)$.
Step 13 : Stop.
```