Scheme and key MID-I

1a) 6 points and diagram 8M

b) general classifications for graphics 7M

**Coordinate Representation**

**Output primitives**

**Software Standards**

**2a)Define CG 1M**

**ANY 4 APPLICATIOC 7M**

**b) Plasma Display explain 5M**

**Diagram 2M**

**3a)Definition 1M**

**Explain about Symmetric 1M**

**Function of circle with conditions 1M**

**Decision Parameter 2M**

**Difference of decision parameter 2M**

**Three cases 3M**

**b) Bundled attributes 2M**

**Inquiry Functions 2M**

**Examples 1M**

**4A) Three cases 3M**

**Algorithm 7M**

b) Four Types with examples 5M

5) Fixed Point Rotation and derivation 5M

b) Example-

Centroid 1M

transformation 1M

Rotation 2M

Order of implementation 1M

6). Reflection matrix line Y=-X 2M

Reflection y axis 2M

Anticlock rotation matrix 2M

Multiplication 2M

Proof 2M

1A)

|  |  |
| --- | --- |
| **Random Scan** | **Raster Scan** |
| 1. Picture definition is stored as a set of line-drawing commands in an area of memory referred to as the **refresh display file**.   Random Scan | 1.the electron beam is swept across the screen, one row at a time from top to bottom. Picture definition is stored in memory area called the **Refresh Buffer** or **Frame Buffer.**  Random Scan and Raster Scan Display |
| 1. It has high Resolution | 1. Its resolution is low. |
| 2. It is more expensive | 2. It is less expensive |
| 3. Any modification if needed is easy | 3.Modification is tough |
| 4. Solid pattern is tough to fill | 4.Solid pattern is easy to fill |
| 5. Refresh rate depends or resolution | 5. Refresh rate does not depend on the picture. |
| 6. Only screen with view on an area is displayed. | 6. Whole screen is scanned. |
| 7. Beam Penetration technology come under it. | 7. Shadow mark technology came under this. |
| 8. It does not use interlacing method. | 8. It uses interlacing |
| 9. It is restricted to line drawing applications | 9. It is suitable for realistic display. |

1b) Graphics Software

A graphics software is an intermediary between an application program & the graphics hardware. The output primitives & interaction devices that a graphics package supports can range from rudimentary to extremely rich.

There are two general classifications for graphics software:

**General Programming packages**: provides an extensive set of graphics functions that can be used in a high-level programming language, such as C or FORTRAN. Basic functions in a general package include those for generating picture components (straight line, circle, polygon etc),setting color and intensity values, & applying transformations.

**Special-purpose applications packages:**Designed for nonprogrammers, so that users can generate displays without worrying about how graphics operations work. Example of such application packages are the artist’s painting programs and various business,medical and CAD systems.

**Coordinate Representation General** Graphics packages are designed to be used with Cartesian coordinates. Several different Cartesian reference frames are used to construct & display a scene. Modelling ,World ,Normalized Device transformation Coordinates , Coordinate Coordinates

**Graphics Functions**

These packages provides users with a variety of functions for creating & manipulating pictures.

**Output primitives:** basic building blocks.

Attributes: properties of the output primitives.

Geometric transformations: changing size, position & orientation.

Modeling transformations: construct scene using object descriptions.

Viewing transformations: are used to specify the view that is to be presented.

Input Functions: used to control & process the data flow from the interactive devices such as mouse, tablet or joystick.

Control operations: contains no. of housekeeping tasks such as clearing a display screen & initializing parameters.

**Software Standards**

A standard graphics package such as GKS(Graphical kernal system) & PHIGS(Programmers Hierarchical Interactive graphics system) implements a specification designated as standard by an official national or international standard bodies by ISO and ANSI(American National Standard Institute).

The main purpose of such standards is to promote portability of application programs & of programmers.ν Non-official standards are also developed, promoted & licensed by individual companies or by consortia of companies eg Adobe’s Post script & MIT’s X window system are two industry standards.

GKS originally designed as a 2-D graphics packages, a 3-D GKS extension was subsequently developed.ν PHIGS is a extension of GKS having increased capabilities for object modeling, color specification, surface rendering etc.ν Extension of PHIGS called PHIGS+ provide 3-D surface shading capabilities.

**GKS primitives:**There are basic four primitives:

(a) Polyline: used to draw lines. POLYLINE(n, X, Y) n = length of an array X & Y = array of x,y coordinate

(b) Polymarker: used to plot points. POLYMARKER(n, X, Y) n = number of data points

(c) Fill Area: also used to draw line but it always connects the first and last points in the array.

FILL AREA(n, X, Y)(d) Text: used to print the “string” or “text” starting at the given coordinates.

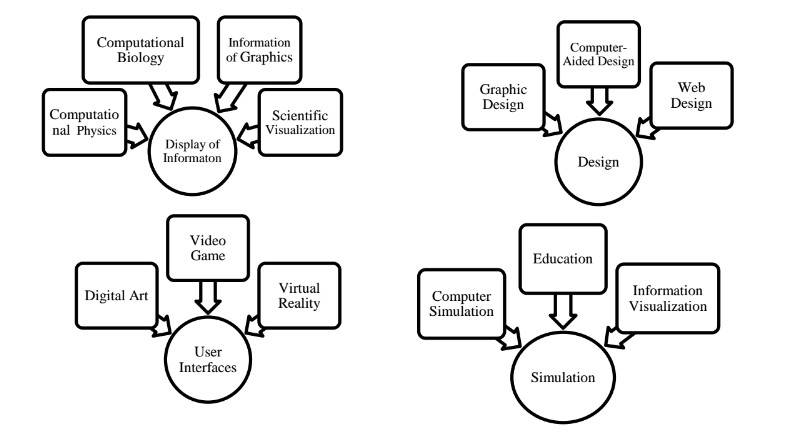
TEXT(x, y, “String”)

(OR)

2a) Computer graphics is an art of drawing pictures, lines, charts, etc using computers with the help of programming. Computer graphics is made up of number of pixels. Pixel is the smallest graphical picture or unit represented on the [computer](http://ecomputernotes.com/fundamental/introduction-to-computer/what-is-computer) screen.

[Computer-generated imagery](http://simple.wikipedia.org/wiki/Computer-generated_imagery) is used for [movie](http://simple.wikipedia.org/wiki/Movie) making, [video game](http://simple.wikipedia.org/wiki/Video_game) and [computer program](http://simple.wikipedia.org/wiki/Computer_program) development, [scientific modeling](http://simple.wikipedia.org/wiki/Scientific_modeling), and design for [catalogs](http://simple.wikipedia.org/w/index.php?title=Catalog&action=edit&redlink=1" \t "_blank) and other [commercial](http://simple.wikipedia.org/wiki/Commercial) art. Some people even make computer graphics as [art](http://simple.wikipedia.org/wiki/Art). We can classify applications of computer graphics into four main areas:

* Display of information
* Design
* User interfaces
* Simulation



2b)

Plasma is referred to be the main element of a fluorescent light. It is actually a gas including ions and electrons. Under normal conditions, the gals has only uncharged particles. That is, the number of positive charged particles [protons] will be equal to the number of negative charged particles [electrons]. This gives the gas a balanced position.

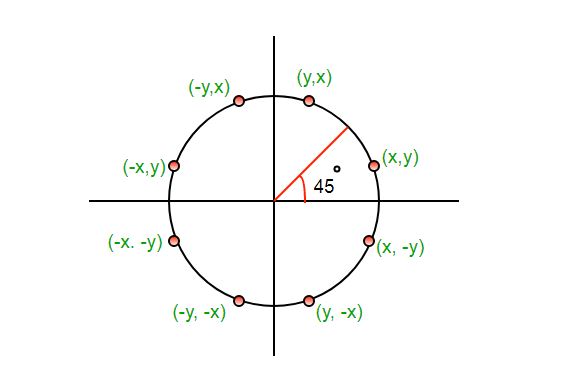
In plasma, photons of energy are released, if an electrical current is allowed to pass through it. Both the electrons and ions get attracted to each other causing inter collision. This collision causes the energy to be produced. Plasma displays mostly make use of the Xenon and neon atoms. When the energy is liberated during collision, light is produced by them.



3a) Circle derivation

We need to plot the perimeter points of a circle whose center co-ordinates and radius are given using the Mid-Point Circle Drawing Algorithm.

We use the above algorithm to calculate all the perimeter points of the circle in the **first octant** and then print them along with their mirror points in the other octants. This will work only because a circle is symmetric about it’s centre.



The algorithm is very similar to the [Mid-Point Line Generation Algorithm](https://www.geeksforgeeks.org/mid-point-line-generation-algorithm/). Here, only the boundary condition is different.

For any given pixel (x, y), the next pixel to be plotted is either **(x, y+1)** or **(x-1, y+1)**. This can be decided by following the steps below.

1. Find the mid-point **p** of the two possible pixels i.e (x-0.5, y+1)
2. If **p** lies inside or on the circle perimeter, we plot the pixel (x, y+1), otherwise if it’s outside we plot the pixel (x-1, y+1)

**Boundary Condition :** Whether the mid-point lies inside or outside the circle can be decided by using the formula:-

*Given a circle centered at (0,0) and radius r and a point p(x,y)****F(p) = x2 + y2 – r2***

*if F(p)<0, the point is inside the circle  
  
F(p)=0, the point is on the perimeter  
  
F(p)>0, the point is outside the circle*

### ‘**Decision parameter**

**pk =fcircle(xk+1,yk-1/2)** where **pk** is a decision parameter and in this **½** is taken because it is a midpoint value through which it is easy to calculate value of **yk** and **yk-1**.

I.e. **pk= (xk+1)2+ (yk-1/2)2-r2**

If **pk <0** then midpoint is inside the circle in this condition we select **y** is **yk** otherwise we will select next **y** as **yk-1** for the condition of **pk > 0**.

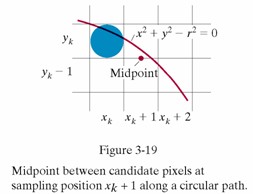
**Midpoint Circle Algorithm**

◼ For a point in the interior of the circle, the circle function is negative and for a point outside the circle, the function is positive

◼ Thus,fcircle(x,y) < 0 if (x,y) is inside the circle boundary

◼circle(x,y) = 0 if (x,y) is on the circle boundary  
◼fcircle(x,y) > 0 if (x,y) is outside the circle boundary

Assuming we have just plotted the pixel at *(xk,yk)* , we next need to determine whether the pixel at position *(xk + 1, yk-1)* is closer to the circle

◼ Our decision parameter is the circle function evaluated at the midpoint between 

*pk = fcircle (xk +1, yk-1/2) = (xk +1)2 + (yk -1/2)2 – r2*

If *pk < 0* , this midpoint is inside the circle and the pixel on the scan line *yk* is closer to the circle boundary. Otherwise, the

mid position is outside or on the circle boundary, and we select the pixel on the scan line *yk-1*

◼Successive decision parameters are obtained using incremental calculations *Pk+1 = fcircle(xk+1+1, yk+1-1/2)*

*= [(xk+1)+1]2 + (yk+1 -1/2)2 –r2*

**Difference between pk and pk+1**

**pk+1= pk** +*[(xk+1)+1]2 + (yk+1 -1/2)2 –r2* -[ *(xk +1)2 + (yk -1/2)2 – r2* ]

Case 1: if(**pk <0** the the next points are(*xk* +1, *yk)*

**pk+1= pk** +*[(xk+1)+1]2 + (yk -1/2)2 –r2* -[ *(xk +1)2 + (yk -1/2)2 – r2* ]

= **pk+2** *xk+3*

Case 2: if(**pk >=0** the the next points are(*xk* +1, *yk-1)*

**pk+1= pk** +*[(xk+1)+1]2 + ((yk-1)-1/2)2 –r2* -[ *(xk +1)2 + (yk -1/2)2 – r2* ]

= **pk+2(** *xk-yk)+5*

Case 3:

◼At the start position *(0,r)* , these two terms have the values 2 and *2r-2* respectively ◼Initial decision parameter is obtained by evaluating the circle function at the start

position *(x0,y0) = (0,r)  
p0 = fcircle(1, r-1/2) = 1+ (r-1/2)2-r2*

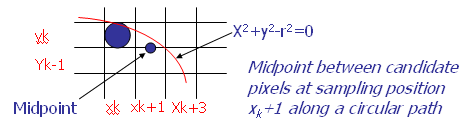
*=5/4 -r*

◼If radius r is specified as an integer, we can round p0 to *p0 = 1-r*

The actual algorithm  
1: Input radius r and circle center (xc,yc) and obtain the first point on the circumference of the circle centered on the origin as

(x0,y0) = (0,r)  
2: Calculate the initial value of the decision parameter as

P0 = 5/4 - r  
3: At each xk position starting at k = 0 , perform the following test:  
If pk < 0 , the next point along the circle centered on (0,0) is (xk+1, yk) and



pk+1 = pk + 2xk+1 + 1

= pk + 2xk + 3

Otherwise the next point along the circle is (xk+1, yk-1) and

pk+1 = pk + 2xk+1 +1 -2yk+1

Where 2xk+1 = 2xk+2 and 2yk+1 = 2yk-2  
4: Determine symmetry points in the other seven octants  
5: Move each calculated pixel position (x,y) onto the circular path centered on (x,yc) and plot the coordinate values

x=x+xc , y=y+yc  
6: Repeat steps 3 through 5 until x >= y

3b) **BUNDLED ATTRIBUTES**

The procedures considered so far each function reference a single attribute that specifies exactly how a primitive is to be displayed these specifications are called individual attributes.

A particular set of attributes values for a primitive on each output device is chosen by specifying appropriate table index. Attributes specified in this manner are called bundled attributes. The choice between a bundled or an unbundled specification is made by setting a switch called the aspect source flag for each of these attributes

**setIndividualASF( attributeptr, flagptr)**

where parameter attributer ptr points to a list of attributes and parameter flagptr points to the corresponding list of aspect source flags. Each aspect source flag can be assigned a value of individual or bundled.

**Bundled line attributes**Entries in the bundle table for line attributes on a specified workstation are set with the function

**setPolylineRepresentation (ws, li, lt, lw, lc)**

Parameter ws is the workstation identifier and line index parameter li defines the bundle table position. Parameter lt, lw, tc are then bundled and assigned values to set the line type, line width, and line color specifications for designated table index.

Example

**setPolylineRepresentation(1,3,2,0.5,1)**

**setPolylineRepresentation (4,3,1,1,7)**

A poly line that is assigned a table index value of 3 would be displayed using dashed lines at half thickness in a blue color on work station 1; while on workstation 4, this same index generates solid, standard-sized white lines

**Bundle area fill Attributes**

Table entries for bundled area-fill attributes are set with

**setInteriorRepresentation (ws, fi, fs, pi, fc)**

Which defines the attributes list corresponding to fill index fi on workstation ws. Parameter fs, pi and fc are assigned values for the fill style pattern index and fill color.

**Bundled Text Attributes**

**setTextRepresentation (ws, ti, tf, tp, te, ts, tc)**

bundles values for text font, precision expansion factor size an color in a table position for work station ws that is specified by value assigned to text index parameter ti.

**Bundled marker Attributes**

**setPolymarkerRepresentation (ws, mi, mt, ms, mc)**

That defines marker type marker scale factor marker color for index mi on workstation ws.

4a)

4b) **LINE ATTRIBUTES**

Basic attributes of a straight line segment are its type, its width, and its color. In some graphics packages, lines can also be displayed using selected pen or brush options

* Line Type
* Line Width
* Pen and Brush Options
* Line color

**Line type**

Possible selection of line type attribute includes solid lines, dashed lines and dotted lines.To set line type attributes in a PHIGS application program, a user invokes the function

**setLinetype (lt)**

Where parameter lt is assigned a positive integer value of 1, 2, 3 or 4 to generate lines that are solid, dashed, dash dotted respectively. Other values for line type parameter it could be used to display variations in dot-dash patterns.

**Line width**

Implementation of line width option depends on the capabilities of the output device to set the line width attributes.

**setLinewidthScaleFactor(lw)**

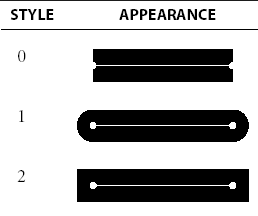
Line width parameter lw is assigned a positive number to indicate the relative width of line to be displayed. A value of 1 specifies a standard width line. A user could set lw to a value of 0.5 to plot a line whose width is half that of the standard line. Values greater than 1 produce lines thicker than the standard.

**Line Cap**

We can adjust the shape of the line ends to give them a better appearance by adding line caps.

There are three types of line cap. They are

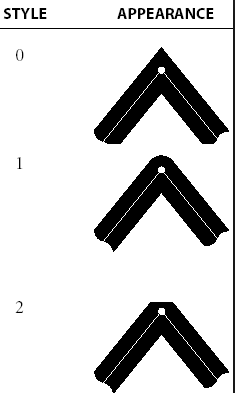
* Butt cap
* Round cap
* Projecting square cap



Butt cap obtained by adjusting the end positions of the component parallel lines so that the thick line is displayed with square ends that are perpendicular to the line path.Round cap obtained by adding a filled semicircle to each butt cap. The circular arcs are centered on the line endpoints and have a diameter equal to the line thicknessProjecting Square cap extend the line and add butt caps that are positioned one half of the line with beyond the secified end points.

Three possible methods for smoothly joining two line segments

* Mitter Join
* Round Join
* Bevel Join



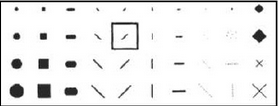
A miter join accomplished by extending the outer boundaries of each of the two lines until they meet.

A round join is produced by capping the connection between the two segments with a circular boundary whose diameter is equal to the width.

A bevel join is generated by displaying the line segment with but caps and filling in tri angular gap where the segments meet.

**Pen and Brush Options**

With some packages, lines can be displayed with pen or brush selections. Options in this category include shape, size, and pattern.



A poly line routine displays a line in the current color by setting this color value in the frame buffer at pixel locations along the line path using the set pixel procedure. We set the line color value in PHlGS with the function

**setPolylineColourIndex (lc)**

Nonnegative integer values, corresponding to allowed color choices, are assigned to the line color parameter lc

Example : Various line attribute commands in an applications program is given by the following sequence of statements

setLinetype(2);

setLinewidthScaleFactor(2);

setPolylineColourIndex (5);

polyline(n1,wc points1);

setPolylineColorIindex(6);

poly line (n2, wc points2);

This program segment would display two figures, drawn with double-wide dashed lines. The first is displayed in a color corresponding to code 5, and the second in color 6.