Introduction to Basic function

INITGRAPH :- Initializes the graphics system

Declaration:- Initgraph (int * graphdriver, int * graphmode, char * pathodriver);

Remarks:

To start the graphic system, you must first call initgraph.

- Initgraph Initializes the graphic system by loading a graphics from disk then putting the system into graphics mode.

Integer that specifies the initial graphics mode (unless * graphdriver = DETECT). If * graphdriver = DETECT, initgraph sets * graphmode to the highest resolution available for the detected driver. you can give * graphmode a value using a constant of the graphics - modes enumeration type.

Close graph :- Closegraph (void)

Remark:- Close graph deallocates all memory allocated by the graphic system.

Return value: None.

Getpixel , Putpixel

- Getpixel get the colour of a specified pixel.
- Putpixel places a pixel at a specified point.

Declaration:-

Getpixel (int x, int y)

Putpixel (int x, int y, int colour)

Return value:-

Getpixel return the colour of the given pixel

Putpixel does not return.

Clear Device:-

Declaration:- void cleardevice ();

Clear device function clears the screen in graphics mode and sets the current position to (0,0). Clearing the screen consists of filling the screen with current background colour

ARC, CIRCLE, PIESLICE

Arc draws a circular arc

Circle draws a circle

Pieslice draws and files a circular pieslice.

Declaration :-

arc (int x, int y, int stangle , int endangle, int radius);
far Circle (int x, int y, int radius)
pieslice (int x, int y, instangle , intendangle , int radius);

Remarks:-

Arc draws a circular arc in the current drawing colour.

Circle draws a circle in the current drawing color.
Pieslice draws a pieslice inthe current drawing color, and then fills it using the current fill pattern and fill color.

ELLIPSE , FILLIPSE , SECTOR

ellipse draws and elliptical arc.
fillipse draws ~~and~~ files and ellipse.
Sector draws and fills and elliptical pie slice.

Declaration:-
ellipse (int x, int y, int stangle, int endangle, int x radius int yradius);
fillellipse (int x, int y, int xradius, int yradius);
far sector (int x, ind y, int stangle, int xradius, int yradius)

Remarks:-

Ellipse draws an elliptical arc in the current drawing Colour.

Fellipse draws an elliptical arc in the current drawing and than fill with it fills colour and fill pattern.

Sector draws an elliptical pie slice in the current drawing colour and than fills it using the pattern and colour defined by setfillstyle or setfillpattern.

Floodfill :- Flood-fills a bounded region

Declaration:- Floodfill (int x, int y, int border)

Remark:-

Floodfills an enclosed area on bitmap device. The area of bounded by the colour border is flooded with the current pattern and fill colour (x,y) is a " seed point.

Floodfill does not work with IBM-8514 driver

Return value:-

If an error occurs while flooding a region, graph result returns '1'.

Getcolor , Setcolor :-

Get color return the current drawing color.
Set color returns the current drawing color.

Declaration:-

get color (void);

setcolor (int color);

Remark :- Get color returns the current drawing color.
Setcolor sets the current drawing color to color which can be range from 0 to getmaxcolor.
To set a drawing color with setcolor, you can pass either the color number or the equivalent color name.

Line , Linerel , Lineto

Line draws a line between two specified pints
Linrto draws a line from the current possition (CP) to (x,y).

Declaration :- line (int x1 , int y1 , x2 , y2);
lineto (int x , int y)

Remarks: Line draws a line from (x1,y1) to (x2,y2) using the current color, line style and thickness.
Lineto draws a line from the CP to (x,y) then moves the the CP to (x,y)

Return : None.

Rectangle:- Draw a rectangle in graphics mode

Declaration:- rectangle (int left, int top, int right, int bottom

Remarks:- It draws a rectangle in the current line style, thickness and drawing colour is the upper left corner of the rectangle, and is its lower right corne

Return value:- None

Text height :- Textheight function returns the height of a string pixel.

Declaration:- int textheight (char* height);

Textwidth:- Textwidth function returns the width of a string pixel.

Declaration:- int textwidth (char* string);

Outtext:- outtext function displays text at current position.

Declaration:- void outtext (char* string);

outtextxy:- outtextxy function display text or string at a specified point (x,y) on the screen.

Declaration:- void outtext xy (int x, int y, char* string);

DDA Line Drawing Programme

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

 void main ( )
{
 int gm, gd = DETECT , errorcode ,i;
 float x1 , x2, y1 ,y2 , dy ,dx ;
 float x, y, xinc ,yinc , l ;
   initgraph ( &gd, &gm , "C:\\TURBOC3\\BGI");
printf ("DDA line drawing Algorithm\n");
printf ("Enter x1 :");
  scanf ("%f" ,&x1 );
Printf (" Enter y1 : ");
 scanf ( "%f", & y1);

 Printf (" Enter x2:");
  Scanf ("%f", & x2 );
 printf ("Enter y2 :");
  scanf (" %f", & y2);
  dy = y2- y1
  dx = x2 - x1

 If ( abs (dy )> =abs (dx) )
     l =abs (dy);
else
        l =  abs (dx);
   xinc  =  dx / l ;
   yinc  =  dy / l ;

   x = x1
   y = y1

 for (i=1 ; i<l ; i++)
{
     x =  x + xinc ;
     y =  y + yinc  ;
    putpixel (x, y, 6);
}
    getch ( );
   closegraph ( );
}
```

Output

```
Enter x1 : 100
Enter y1 : 100
Enter x2 : 200
Enter ~~x~~y2 : 200
```

Bresenham line drawing programme

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

 void main ()
{
   int  gm , gd = DETECT ;
   float  x1 , x2, y1 , y2 , dx , dy ,temp ,p ;
   float  x , y, xend , yend ;
   initgraph ( &gm ,&gd ,"C://TURBOC3// BGI");
Printf ("Bresenham's line drawing algorithm\n\n");

 printf ("Enter x1 :");
 scanf ("%f",& x1);
 printf ("Enter y1 :");
 scanf ("%f" ,& y1);
 printf ("Enter x2: ");
 scanf (" %f",& x2);
 printf("Enter y2;");
 scanf(" %f ",& y2);

   dx = abs ( x2- x1 );
   dy = abs ( y2 -y1);
  If (dy> dx)
 {
   temp = dy;
   dy = dx ;
   dx = temp ;
 }
```

```
P = dx - (2* dy);
if (x1< x2)
{
    x = x1 ;
    y = y1 ;
    xend = x2;
}

else
  {
     x = x2 ;
     y = y2 ;
     xend = x1;
while (x< xend)
{
 x++ ;
 If (P<0)
 {
   P= p+ 2* dx-2* dy ;
   y++;
 }
 else
   {
      P= p-2* dy ;
   } Putpixel ( ceil (x), ceil (y), 5);
      }
 } getch ();
```

Out-put:

```
Enter x1 : 100
Enter y1 : 100
Enter x2 : 200
Enter y2 : 200
```

Midpoint circle generating programme

```
# include <Stdio.h>
# include <graphics.h>
# include < math.h>

  void main()
{
int gm, gd, =DETECT ;
int  p,x,y,r ;
int xcenter =0 , ycenter =0;

 initgraph(&gm,&gd,"C://TURBOC3//BGI");
printf("Midpoint circle drawing algorithm\n\n\n");
printf(" Enter the radius :");
 scanf ("%d",&r );
printf ("Enter center coordinates:");
 scanf("%d %d ,&xcenter, & ycenter);


 x=0 ;
 y=r ;
 p= 1-r :
 while (x<y)
{
   If (P<0)
  {
     x= x+1 ;
     P =P+2*x+1;
  }
else
  {
       x= x+1 ;
       y= y-1 ;
      p=p+2*(x-y)+1 ;
  }
      else
 putpixel(xcenter+x, ycenter+y ,6);
 putpixel (xcenter-x , ycenter+y ,6);
  putpixel(xcenter+x, ycenter-y,6 );
  putpixel( xcenter -x, ycenter-y,6);

  putpixel(xcenter+y , ycenter+x ,6 );
  putpixel(xcenter-y , ycenter+x ,6);
   put pixel(xcenter+y, ycenter-x ,6);
   putpixel(xcenter-y, ycenter - x ,6);
}
  getch ();
 }
```

Output :

```
Enter radius
Enter center coordinates
```