**arc():** arc() draws the outline of an arc in the current drawing color.

**Code:**

#include <graphics.h>

#include <conio.h>

main()

{

int gd = DETECT, gm;

initgraph(&gd, &gm, "C:\\TC\\BGI");

arc(200, 200, 2, 140, 100);

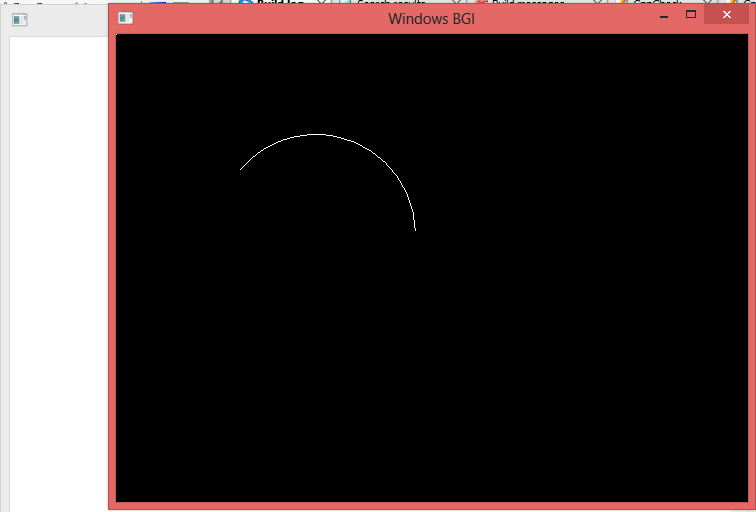
getch();

closegraph();

return 0;

}

**Output:**



**bar():** bar() draws and fills in a rectangular bar.

**Code:**

#include<stdio.h>

#include<graphics.h>

#include<conio.h>

int main()

{

int gd = DETECT,gm;

initgraph(&gd, &gm, "C:\\TC\\BGI");

/\* Draw Bar on screen \*/

bar(150, 200, 400, 350);

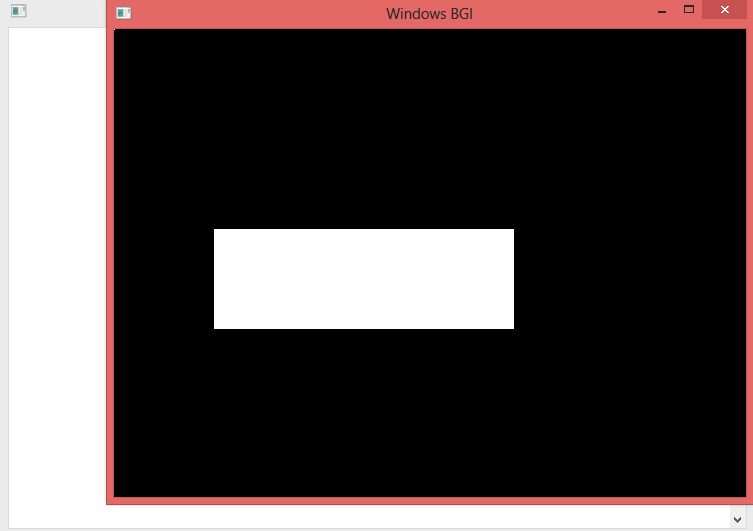
getch();

closegraph();

return 0;

}

**Output:**



**bar3d():** bar3d() draws and fills in a three-dimensional rectangular bar.

**Code:**

#include<graphics.h>

#include<conio.h>

main()

{

int gd = DETECT, gm; initgraph(&gd, &gm, "C:\\TC\\BGI");

bar3d(100, 100, 400, 400, 40, 2);

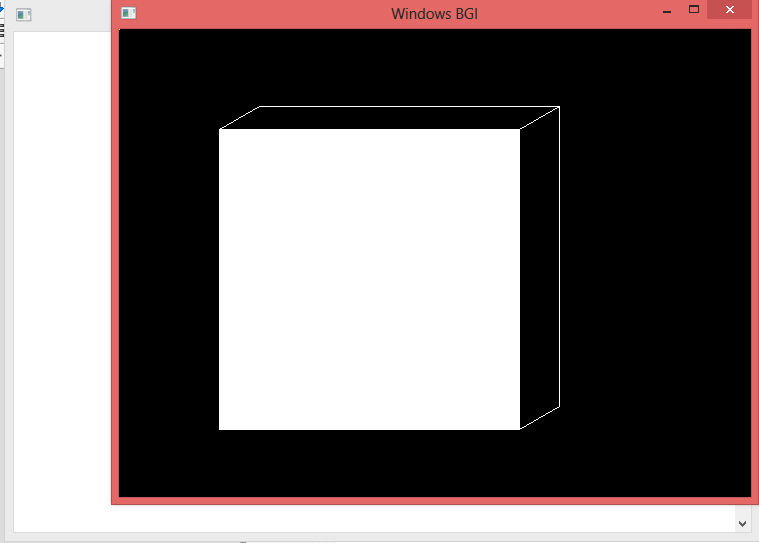
getch();

closegraph();

return 0;

}

**Output:**



**Circle:**Circle draws a circle.

**Code:**

#include<conio.h>

#include<graphics.h>

main()

{

int gdriver=DETECT,gmode;

initgraph(&gdriver,&gmode,"C:\TC\BGI");

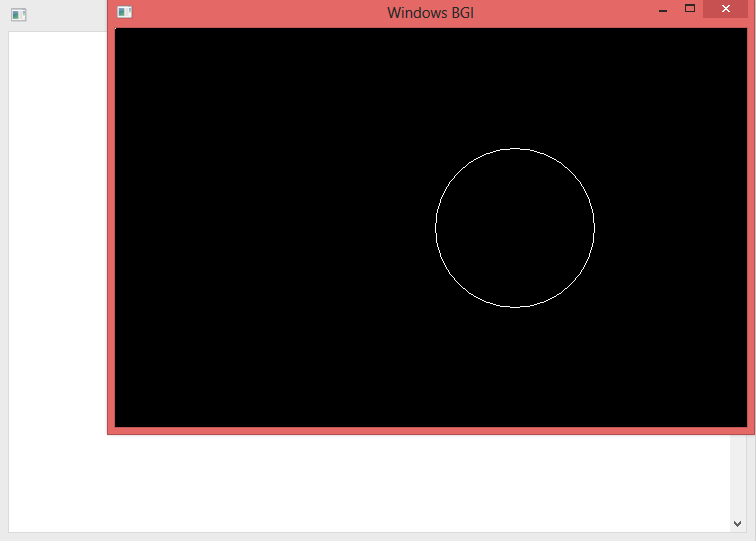
circle(400,200,80);

getch();

closegraph();

}

**Output:**



**Ellipse:** Ellipse draws an elliptical arc.

**getmaxx():**getmaxx() returns the maximum x screen coordinate for the current graphics mode. This function is very useful for positioning text and graphics on the screen.

**getmaxy() :**getmaxy() returns the maximum y screen coordinate for the current graphics mode. This function is very useful for positioning text and graphics on the screen.

**Code:**

#include<stdio.h>

#include<graphics.h>

#include<conio.h>

int main(){

int gd = DETECT,gm;

int x ,y;

initgraph(&gd, &gm, "X:\\TC\\BGI");

/\* Initialize center of ellipse with center of screen \*/

x = getmaxx()/2;

y = getmaxy()/2;

outtextxy(x-50, 100, "ELLIPSE Using Graphics in C++");

/\* Draw ellipse on screen \*/

ellipse(x, y, 0, 360, 120, 60);

getch();

closegraph();

return 0;

}

**Output:**



**Cleardevice():**

cleardevice() erases the entire graphics screen and moves the current

position(cp) to home position which is the upper left hand corner of the screen with coordinates(0,0).

**Code:**

#include <graphics.h>

#include <conio.h>

main()

{

int gd = DETECT, gm;

initgraph(&gd, &gm, "C:\\TC\\BGI");

outtext("Press any key to clear the screen.");

getch();

cleardevice();

outtext("Press any key to exit...");

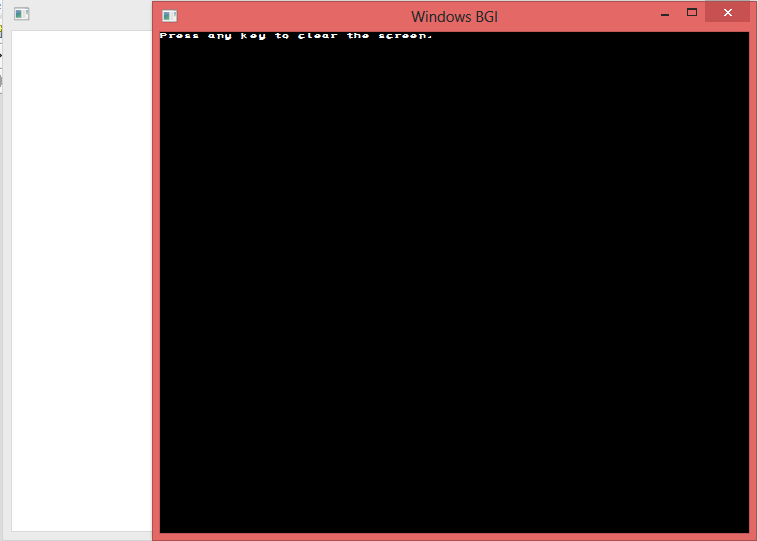
getch();

closegraph();

return 0;

}

**Output:**



**closegraph**:closegraph() shut down the graphics system by deallocating all the memory that was allocated by initgraph().closegraph() is a graphics system control function and should be the last graphics function called.

**drawpoly():**drawpoly() draws a polygon with 'npoint' points. 'ppoints' points to a sequence of pair integers.Each pair of integers represents the (‘x’, ‘y’) coordinates of ‘npoint’.

**Code:**

#include <graphics.h>

#include <conio.h>

main()

{

int gd=DETECT,gm,points[]={400,200,600,300,250,300,400,200};

initgraph(&gd, &gm, "C:\\TC\\BGI");

drawpoly(4, points);

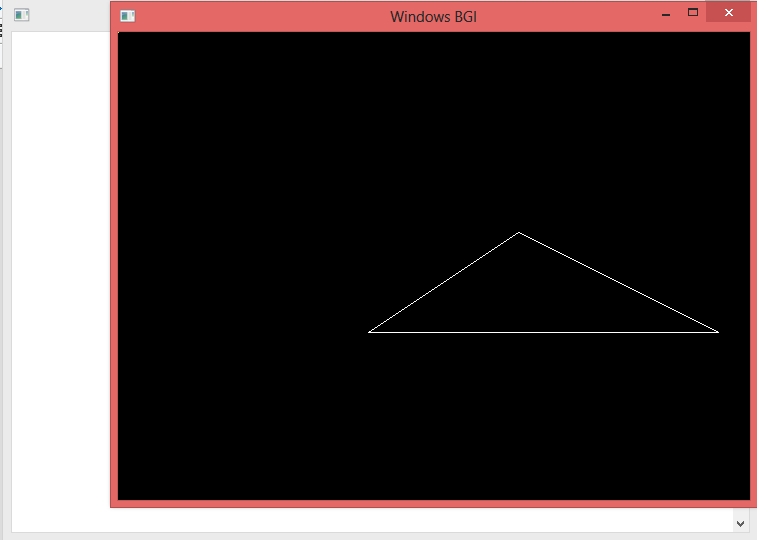
getch();

closegraph();

return 0;

}

**Output:**



**fillellipse():**The function draws and fills an ellipse with center point at (x,y),horizontal radius xrad, and vertical radius yrad.The current fill color is used.

**Code:**

#include <graphics.h>

#include <conio.h>

int main()

{

int gd = DETECT, gm;

initgraph(&gd, &gm, "C:\\TC\\BGI");

fillellipse(400, 180, 80, 40);

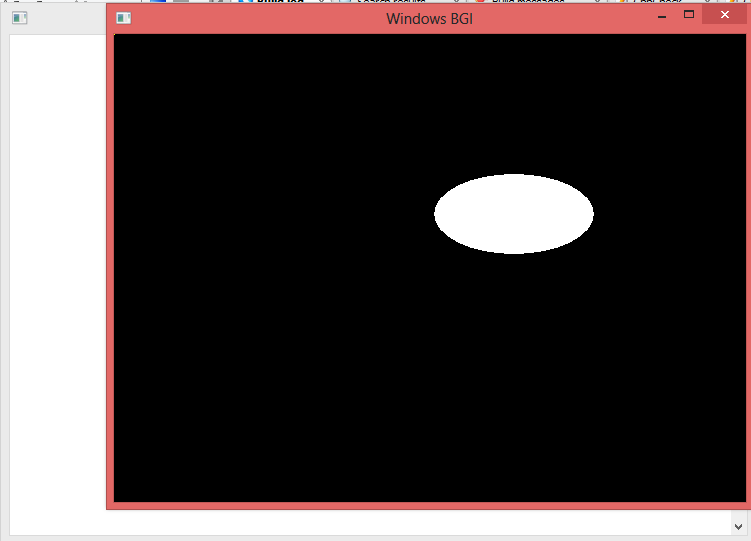
getch();

closegraph();

return 0;

}

**Output:**



**floodfill():**floodfill() fills figures drawn using arc(), circle(), ellipse(), drawpoly(), line(), lineto(), linerel() and rectangle(). The bounded area is filled with the current fill pattern and fill color.

**Code:**

#include<stdio.h>

#include<graphics.h>

#include<dos.h>

void floodFill(int x,int y,int oldcolor,int newcolor)

{

if(getpixel(x,y) == oldcolor)

{

putpixel(x,y,newcolor);

floodFill(x+1,y,oldcolor,newcolor);

floodFill(x,y+1,oldcolor,newcolor);

floodFill(x-1,y,oldcolor,newcolor);

floodFill(x,y-1,oldcolor,newcolor);

}

}

//getpixel(x,y) gives the color of specified pixel

int main()

{

int gm,gd=DETECT,radius;

int x,y;

printf("Enter x and y positions for circle\n");

scanf("%d%d",&x,&y);

printf("Enter radius of circle\n");

scanf("%d",&radius);

initgraph(&gd,&gm,"c:\\turboc3\\bgi");

circle(x,y,radius);

floodFill(x,y,0,15);

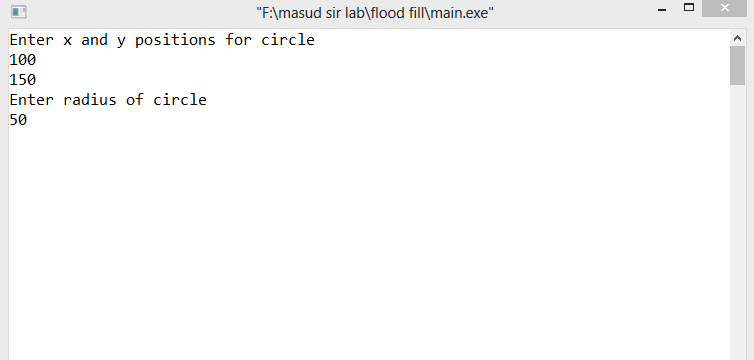
delay(5000);

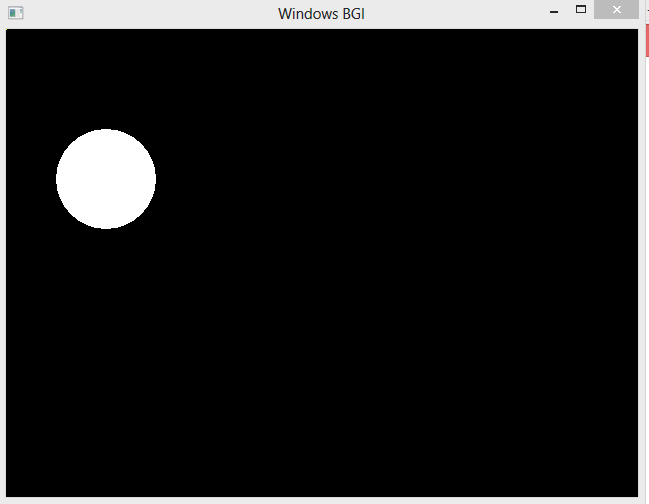
closegraph();

return 0;

}

**Output:**





**boundaryfill():** boundaryfill() function fill the boundary.For filling a region with single boundary color.

**Code:**

// Write a program for Boundary Fill Algorithm.

#include<bits/stdc++.h>

#include<graphics.h>

using namespace std;

int BoundaryFill(int x,int y)

{

int CheckColor;

CheckColor=getpixel(x,y);

if((CheckColor!=WHITE)&&(CheckColor!=YELLOW))

{

putpixel(x,y,WHITE);

BoundaryFill(x+1,y);

BoundaryFill(x,y+1);

BoundaryFill(x-1,y);

BoundaryFill(x,y-1);

}

return 0;

}

int main()

{

int x,y,gd=DETECT,gm;

initgraph(&gd,&gm,"");

rectangle(50,50,200,200);

x=110,y=111;

BoundaryFill(x,y);

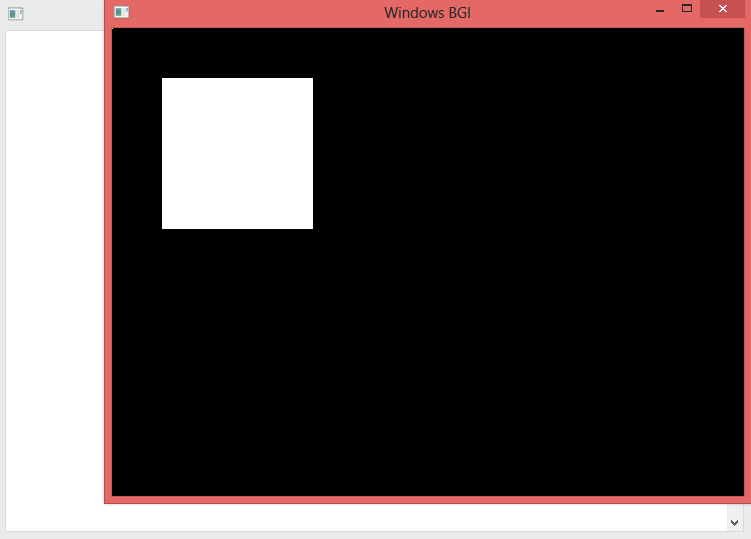
getch();

closegraph();

return 0;

}

**Output:**



**setcolor():**setcolor() sets the current drawing color to 'color', which represents an index into the palette. The range of values 'color' can have is determined by the size of the palette.

**Code:**

#include<graphics.h>

#include<conio.h>

main()

{

int gd = DETECT, gm;

initgraph(&gd,&gm,"C:\\TC\\BGI");

circle(100,100,50); /\* drawn in white color \*/

setcolor(YELLOW);

circle(200,200,50); /\* drawn in red color \*/

getch();

closegraph();

return 0;

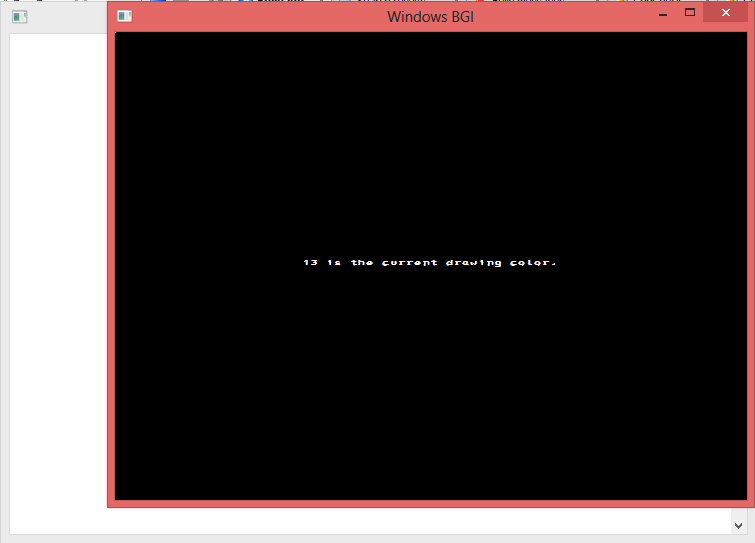
}

**Output:**



**getcolor():**getcolor() returns the current drawing color. The value returned is actually an index into the palette which contains the exact color information.

**Code:**



**setbkcolor():**setbkcolor() sets the current background color which can be

the values defined in <graphics.h>

**Code:**

#include<graphics.h>

#include<conio.h>

main()

{

int gd = DETECT, gm;

initgraph(&gd, &gm, "C:\\TC\\BGI");

outtext("Press any key to change the background color to GREEN.");

getch();

setbkcolor(YELLOW);

getch();

closegraph();

return 0;

}

Output:

**getbkcolor**() : getbkcolor() returns the current background color which can be one of

the following values defined in <graphics.h> .

**Code:**

#include<graphics.h>

#include<conio.h>

#include<stdio.h>

main()

{

int gd = DETECT, gm, bkcolor;

char a[100];

initgraph(&gd,&gm,"C:\\TC\\BGI");

bkcolor = getbkcolor();

sprintf(a,"Current background color = %d", bkcolor);

outtextxy( 10, 10, a);

getch();

closegraph();

return 0;

}

**Output:**

****

**getmaxcolor():**getmaxcolor() returns the highest valid pixel value for the current graphics driver and mode. A pixel value is an index into a color table called a 'palette'.

**Code:**

#include<graphics.h>

#include<conio.h>

#include<stdio.h>

main()

{

int gd = DETECT, gm, max\_colors;

char a[100];

initgraph(&gd,&gm,"C:\\TC\\BGI");

max\_colors = getmaxcolor();

sprintf(a,"Maximum number of colors for current graphics mode and driver = %d",max\_colors+1);

outtextxy(0, 40, a);

getch();

closegraph();

return 0;

}

**Output:**

****

**getdrivername() :** getdrivername() function returns a pointer to a string holding the name of the current graphics driver.

**Code:**

#include<graphics.h>

#include<conio.h>

main()

{

int gd = DETECT, gm;

char \*drivername;

initgraph(&gd, &gm, "C:\\TC\\BGI");

drivername = getdrivername();

outtextxy(200, 200, drivername);

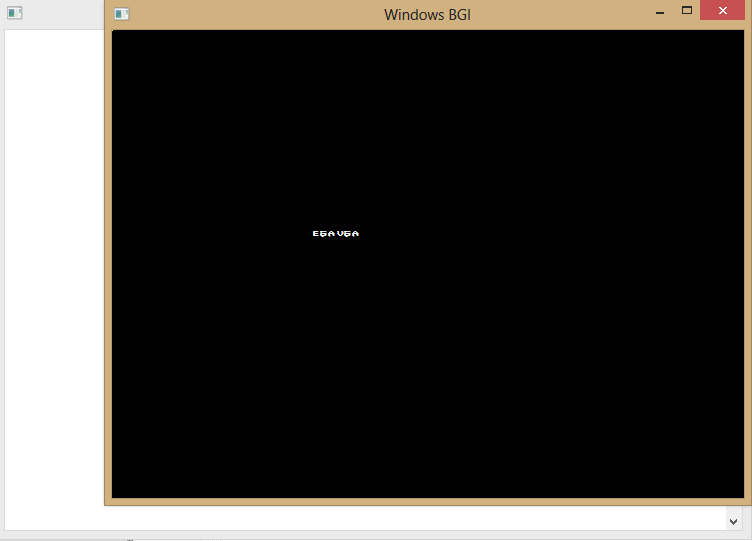
getch();

closegraph();

return 0;

}

**Output:**

****

**getmodename() :**the getmodename() function returns a pointer to a string which contains the name of the indicated graphics mode, as specified by mode\_num..

**Code:**

**Output:**

**getpixel():**getpixel() gets the color of the pixel specified by coordinates ('x','y').

Returns: The color of the specified pixel.

**Code:**

#include<graphics.h>

#include<conio.h>

main()

{

int gd = DETECT, gm;

initgraph(&gd, &gm, "C:\\TC\\BGI");

putpixel(25, 25, WHITE);

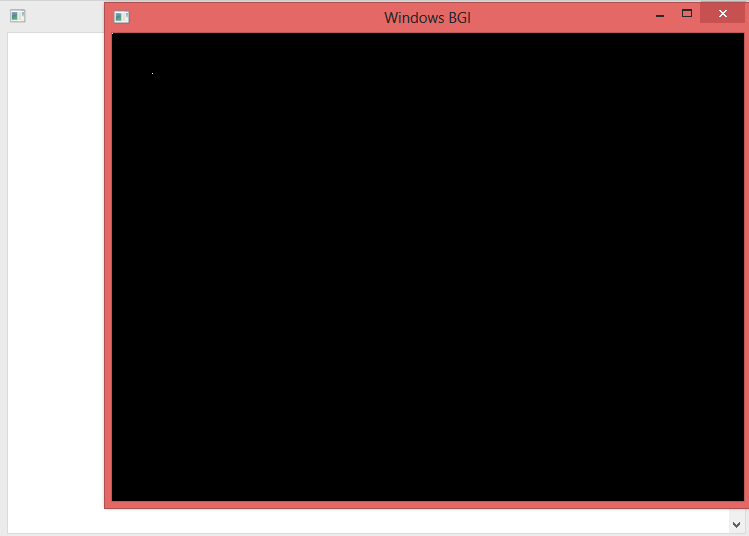
getch();

closegraph();

return 0;

}

**Output:**

****

**putpixel():**putpixel() draws a pixel at the specified coordinates ('x','y'). The pixel is displayed in the color 'pcolor'.

**Code:**

#include<graphics.h>

#include<conio.h>

main()

{

int gd = DETECT, gm;

initgraph(&gd, &gm, "C:\\TC\\BGI");

putpixel(25, 25, WHITE);

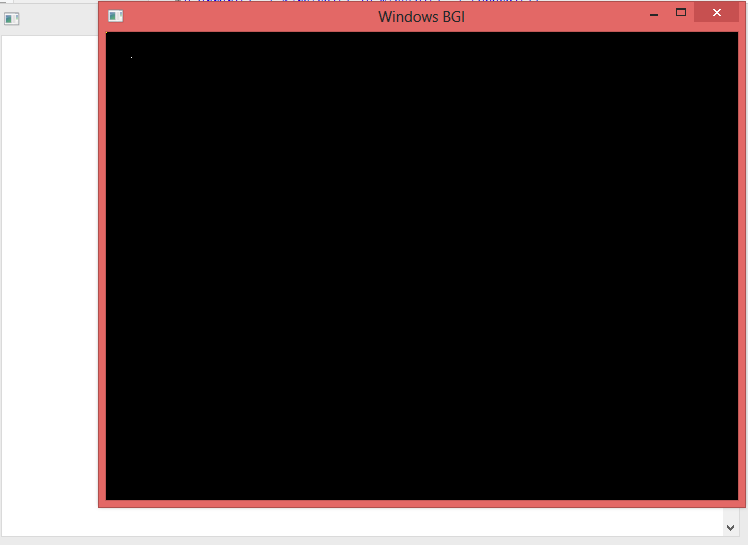
getch();

closegraph();

return 0;

}

**Output:**

****

**outtextxy()** :outtextxy() outputs graphics text at the specified position ('x','y') relative to the current viewport.The text is output using the current text font, text direction, character size and text justification settings.

**Code:**

#include<graphics.h>

#include<conio.h>

main()

{

int gd = DETECT, gm;

initgraph(&gd,&gm,"C:\\TC\\BGI");

outtextxy(250, 210, "Outtextxy function");

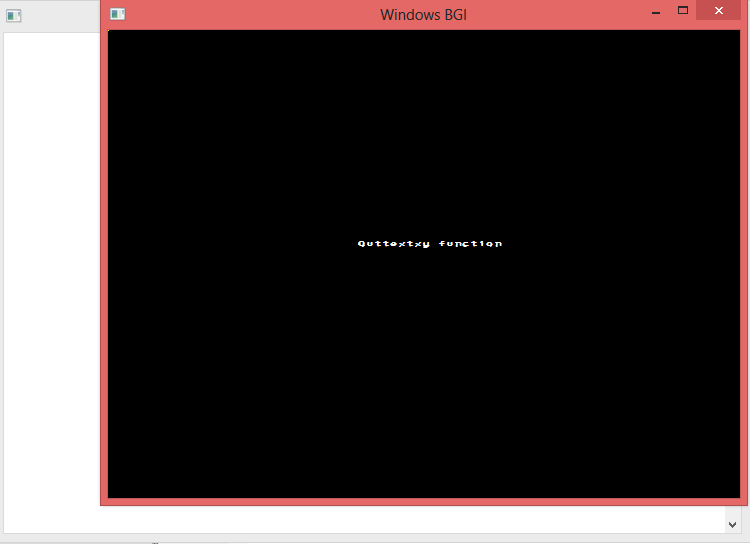
getch();

closegraph();

return 0;

}

**Output:**

****

**setfillstyle() :**setfillstyle() sets the current fill pattern and fill color used by bar(), bar3d(), fillpoly(), floodfill() and pieslice().

**Code**:

#include<graphics.h>

#include<conio.h>

main()

{

int gd = DETECT, gm;

initgraph(&gd, &gm, "C:\\TC\\BGI");

setfillstyle(XHATCH\_FILL, RED);

circle(100, 100, 50);

floodfill(100, 100, WHITE);

getch();

closegraph();

return 0;

}

**Output**:

**settextstyle():**settextstyle() sets the current text font, direction and character size. All calls to outtext() and outtextxy() are affected by the new settings.

**Code:**

#include <graphics.h>

#include <conio.h>

main()

{

int gd = DETECT, gm, x = 25, y = 25, font = 0;

initgraph(&gd,&gm,"C:\\TC\\BGI");

for (font = 0; font <= 10; font++)

{

settextstyle(font, HORIZ\_DIR, 1);

outtextxy(x, y, "Text With Different Fonts");

y = y + 25;

}

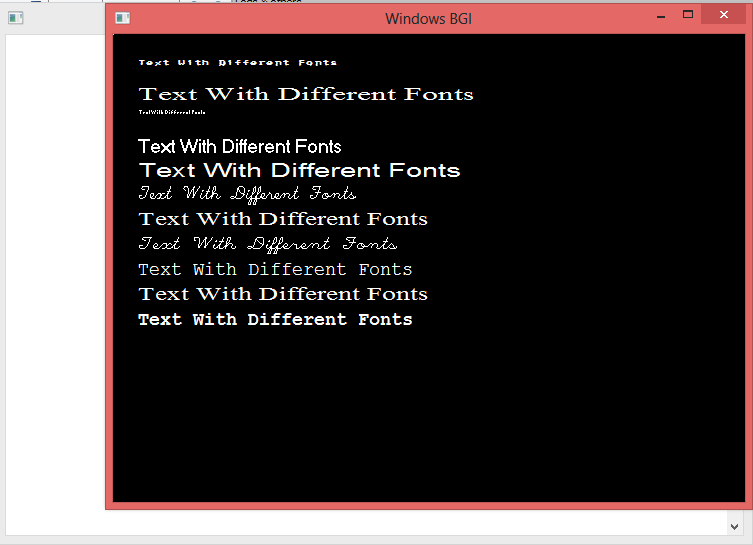
getch();

closegraph();

return 0;

}

**Output:**



**setviewport():**setviewport() defines a rectangular viewport, a "virtual screen", on the screen. The viewport's position is defined in terms of absolute screen coordinates, ('left','top'), ('right','bottom').

**Code:**

#include<graphics.h>

#include<conio.h>

main()

{

int gd = DETECT, gm, midx, midy;

initgraph(&gd, &gm, "F:\\masud sir lab\\set view port");

midx = getmaxx()/2;

midy = getmaxy()/2;

setviewport(midx - 50, midy - 50, midx + 50, midy + 50, 1);

circle(50, 50, 55);

getch();

closegraph();

return 0;

}

**Output:**

