Graphics in C

In C programming the concept of graphics can be used to draw different shapes like circles, lines, rectangles, bars,ellipse etc. It is also used to display text in different fonts, change colors and many more.

Using functions of graphics it is possible to make graphics programs, animations, projects and games. Also it is possible to draw geometric figures,change their colors using the available functions and fill them.

"To use graphics functions in the program it is necessary to include graphics.h header file."

Following is a list of some graphics functions:-

initgraph( ) Function In Graphics

- The first step in any graphics program is to initialize the graphics drivers on the computer using **initgraph** method.
- To start the graphics system it is necessary to call initgraph( ).
- initgraph initializes the graphics system by loading a graphics driver from disk, then putting the system into graphics mode.
- initgraph function is available in ***<graphics.h>*** header file.

Syntax:

```
void initgraph(int *graphics-driver, int *graphics-mode, char *path-to-driver);
```

or

```
initgraph(address-of-graphics-driver, address-of-graphics-mode,"path-of-driver");
```

```
ex.   int gd = DETECT,gm;
      initgraph (&gd, &gm, "c:/TC/BGI");
```

gd =graphics driver   **gm**=graphics mode

BGI= **B**orland **G**raphics **I**terface

In above syntax:-
***graphics-driver** :- Integer that specifies the graphics driver to be used.
***graphics-mode:** - Integer that specifies the initial graphics mode
***pathtodriver:**- Specifies the directory path where initgraph looks for graphics drivers (*.BGI).

Colors in C Graphics Programming

- There are 16 colors declared in C Graphics. We use colors to set the current drawing color, change the color of background, change the color of text etc.
- To specify a color, we can either use color constants like setcolor(RED), or their corresponding integer codes like setcolor(4).
- Below is the color code in **increasing order.**

COLOR NAME	INTEGER VALUE		COLOR NAME	INTEGER VALUE
BLACK	**0**		DARKGRAY	**8**
BLUE	**1**		LIGHTBLUE	**9**
GREEN	**2**		LIGHTGREEN	**10**
CYAN	**3**		LIGHTCYAN	**11**
RED	**4**		LIGHTRED	**12**
MAGENTA	**5**		LIGHTMAGENTA	**13**
BROWN	**6**		YELLOW	**14**
LIGHTGRAY	**7**		**WHITE**	**15**

Drawing objects in C

- **Line**
- **Circle**
- **Rectangle**
- **Ellipse**

Line function in c

- Line function in c is used to draw a line from a point(x1,y1) to point(x2,y2)
- i.e. (x1,y1) and (x2,y2) are end points of the line.
- The code given below draws a line.

Syntax :- ***void line(int x1, int y1, int x2, int y2);***

(x1,y1) ——————————— **(x2,y2)**
Starting point of line **End point of line**

/*Program to draw a line*/

```
#include <graphics.h>
#include <conio.h>
void main()
{
  int gd = DETECT, gm;
   initgraph(&gd, &gm, "C:/TC/BGI");
   line(100, 100, 200, 200);
   getch( );
  closegraph( );
 }
```

Where (100,100 start point) (200,200 end point)

Above program draw a line with co-ordinates 100,100 and 200,200

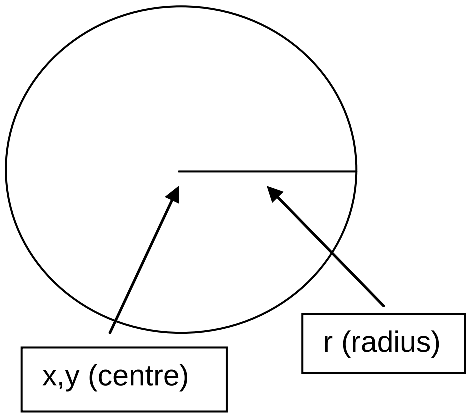
Circle function

- Circle function is used to draw a circle with **center (x,y)** and
- third parameter specifies the **radius** of the circle.

Syntax :- ***void circle(int x, int y, int r);***

Where **x ,y** :- coordinates of **centre.**

r :- **radius** of circle



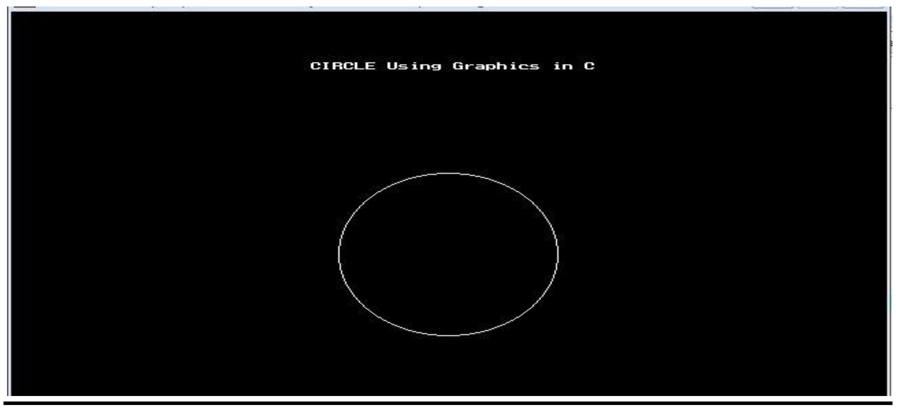
/* Program to draw a circle */

```
 #include<graphics.h>
#include<conio.h>
 void main()
{
   int gd = DETECT, gm;
    initgraph(&gd, &gm, "C:\\TC\\BGI");
    circle(100, 100, 50);
    getch();
   closegraph();
}
```

- The function circle(100,100,50),
- Where (100,100)= co-ordinates of centre
- (50)= radius of circle
- In this program we first initialize graphics mode, by passing graphics driver(DETECT), default graphics mode and specifies the directory path where initgraph looks for graphics drivers (*.BGI).
- It is the first step to do during graphics programming.
- Setting graphics driver as DETECT, means instructing the compiler to auto detect graphics driver.

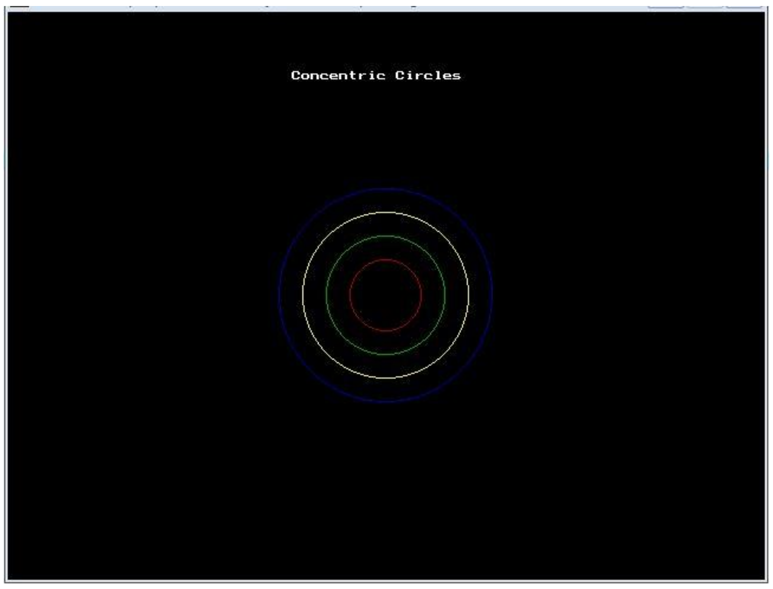
```
/*C program to draw circle using c graphics */
#include<stdio.h>
#include<graphics.h>
#include<conio.h>
void main( )
{
  int gd = DETECT,gm;
  int x ,y ,radius=80;
  initgraph(&gd, &gm, "C:\\TC\\BGI");
  x = getmaxx()/2;
  y = getmaxy()/2;
   outtext("CIRCLE Using Graphics in C");
  circle(x, y, radius);
   getch();
  closegraph();
 }
```

- in this program we used
- ***getmaxx*** It returns the maximum X coordinate in current graphics mode and driver.
- ***getmaxy*** It returns the maximum Y coordinate in current graphics mode and driver.
- ***outtextxy*** It displays a string on the screen.
- ***circle*** It draws a circle with radius r and centre at (x, y).
- ***closegraph*** It unloads the graphics drivers and sets the screen back to text mode.
- Here we are using getmaxx and getmaxy function to find the center coordinate of the screen.



```
CIRCLE Using Graphics in C
```

```
/*C program to draw concentric circle using c graphics */
#include<stdio.h>
#include<graphics.h>
#include<conio.h>
void main( )
{
   int gd = DETECT,gm;
   initgraph(&gd, &gm, "C:\\TC\\BGI");
    outtext("Concentric Circles");
   circle(100,100,20);
   circle(100,100,40);
   circle(100,100,60);
   circle(100,100,80);
    getch();
   closegraph();
 }
```



Concentric Circles

rectangle function in c

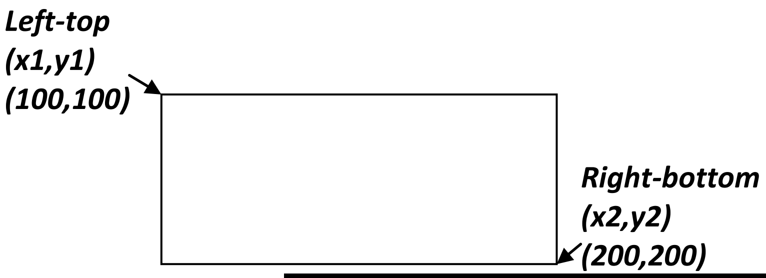
- rectangle function is used to draw a rectangle.
- Coordinates of left top and right bottom corner are required to draw the rectangle. left specifies the X-coordinate of top left corner, top specifies the Y-coordinate of top left corner, right specifies the X-coordinate of right bottom corner, bottom specifies the Y-coordinate of right bottom corner.

Syntax: ***void rectangle(int left, int top, int right, int bottom);***

or

void rectangle(int x1, int y1, int x2, int y2);

```
/* Program to draw a rectangle*/
#include<graphics.h>
#include<conio.h>
void main()
{
   int gd = DETECT, gm;
   initgraph(&gd, &gm, "C:\\TC\\BGI");
   rectangle(100,100,200,200);
   getch();
  closegraph();
 }
```



bar function in c

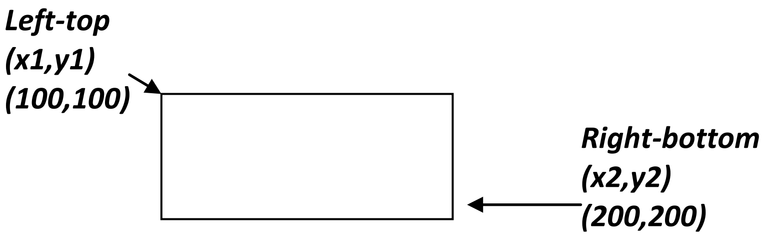
- bar function is used to draw a bar.
- Coordinates of left top and right bottom corner are required to draw the bar. left specifies the X-coordinate of top left corner, top specifies the Y-coordinate of top left corner, right specifies the X-coordinate of right bottom corner, bottom specifies the Y-coordinate of right bottom corner.
- ***"bar is a filled rectangle"***

Syntax: ***void rectangle(int left, int top, int right, int bottom);***

or

void rectangle(int x1, int y1, int x2, int y2);

```
/* Program to draw a bar*/
#include<graphics.h>
#include<conio.h>
void main()
{
   int gd = DETECT, gm;
   initgraph(&gd, &gm, "C:\\TC\\BGI");
   bar(100,100,200,200);
   getch();
  closegraph();
 }
```



ellipse function in c

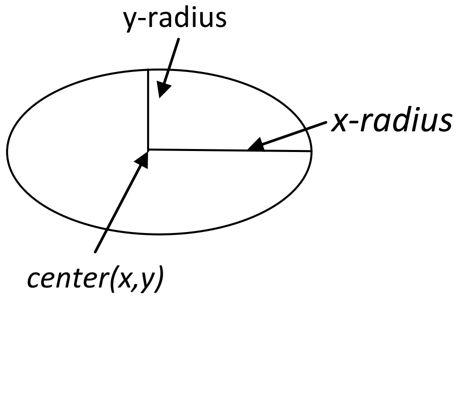
- Ellipse is used to draw an ellipse
- ***(x,y)*** are coordinates of ***center of the ellipse***,
- ***stangle*** is the ***starting angle***,
- ***endangle*** is the ***ending angle,*** and
- ***fifth and sixth parameters*** specifies the ***X-radius and Y-radius*** of the ellipse.
- To draw a complete ellipse strangles and end angle should be 0 and 360 respectively.

Syntax :-
void ellipse(int x,int y,int stangle,int endangle,int x-radius, int y- radius);

/*progfam to draw ellipse */

```
 #include<graphics.h>
#include<conio.h>
void main()
{
   int gd = DETECT, gm;
   initgraph(&gd, &gm, "C:\\TC\\BGI");
   ellipse(100, 100, 0, 360, 50, 25);
   getch();
  closegraph();
}
```

Where ***ellipse(100, 100, 0, 360, 50, 25);***
100,100 = x,y of centerof ellipse
0=sarting angle
360=end angle
50= x-radius
25= y-radius



putpixel function in c

- putpixel function plots a pixel at location (x, y) of specified color.

Syntax :- ***void putpixel(int x, int y, int color);***

For example if we want to draw a GREEN color pixel at (35, 45) then we will write putpixel(35, 35, GREEN);
in our c program, putpixel function can be used to draw circles, lines and ellipses using various algorithms.

```
#include<graphics.h>
#include<conio.h>
 void main()
{
   int gd = DETECT, gm;
    initgraph(&gd, &gm, "C:\\TC\\BGI");
    putpixel(25, 25, RED);
       getch();
   closegraph()
}
```

closegraph function in c

- At the end of our graphics program, we have to unloads the graphics drivers and sets the screen back to text mode by calling closegraph function.
- closegraph function closes the graphics mode, deallocates all memory allocated by graphics system and restores the screen to the mode it was in before you called initgraph

Syntax: **void closegraph();**

```
#include<graphics.h>
#include<conio.h>
 void main()
{
   int gd = DETECT, gm;
    initgraph(&gd, &gm, "C:\\TC\\BGI");
    outtext("Press any key to close the graphics mode...");
   getch();
   closegraph();
  }
```

outtext function

outtext function displays text at current position.

Syntax :- **void outtext(char *string);**

```
/* Program for outtext */
#include<graphics.h>
#include<conio.h>
void main()
{
   int gd = DETECT, gm;
    initgraph(&gd, &gm, "C:\\TC\\BGI");
    outtext("To display text at a particular position on the screen use
outtextxy");
 getch();
   closegraph();
 }
```

/*Sample program covering all graphics*

```
#include<graphics.h>
#include<conio.h>
 void main()
{
   int gd = DETECT,gm;
   int left=100,top=100,right=200,bottom=200,x= 300,y=150,radius=50;
    initgraph(&gd, &gm, "C:\\TC\\BGI");
    rectangle(left, top, right, bottom);
   circle(x, y, radius);
   bar(left + 300, top, right + 300, bottom);
   line(left - 10, top + 150, left + 410, top + 150);
   ellipse(x, y + 200, 0, 360, 100, 50);
   outtextxy(left + 100, top + 325, "My First C Graphics Program");
    getch();
   closegraph();
}
```

setcolor function

setcolor function is used to change the current drawing color.e.g. setcolor(RED) or setcolor(4) changes the current drawing color to RED.

Syntax :- **void setcolor(int color);**

- In Turbo Graphics each color is assigned a number. Total 16 colors are available.

For Example :- BLACK is assigned 0, WHITE is assigned 15 etc.

Remember that default drawing color is WHITE.

/*Program for setcolor*/

```
#include<graphics.h>
#include<conio.h>
 void main()
{
   int gd = DETECT, gm;
   initgraph(&gd,&gm,"C:\\TC\\BGI");
    circle(100,100,50);         /* drawn in white color */
   setcolor(RED);
   circle(200,200,50);         /* drawn in red color   */
    getch();
   closegraph();
 }
```

Setbkcolor function in c

- setbkcolor function changes current background color
- ex. setbkcolor(YELLLOW) changes the current background color to YELLOW.
- The default drawing color is WHITE and background color is BLACK.

Syntax **:-** ***void setbkcolor(int color);***

/*Program for background color*/

```
 #include<graphics.h>
#include<conio.h>
void main()
{
  int gd = DETECT, gm;
  initgraph(&gd, &gm, "C:\\TC\\BGI");
   outtext("Press any key to change the background color to RED.");
   setbkcolor(GREEN);
   getch();
  closegraph();
  }
```

floodfill function

- floodfill function is used to fill an enclosed area.
- Current fill pattern and fill color is used to fill the area.(x, y) is any point on the screen if (x,y) lies inside the area then inside will be filled otherwise outside will be filled,border specifies the color of boundary of area. To change fill pattern and fill color use setfillstyle.

Syntax: ***void floodfill(int x, int y, int border);***

/*Code given below draws a circle and then fills it.*/

```
#include <graphics.h>
#include <conio.h>
 void main()
{
   int gd = DETECT, gm;
    initgraph(&gd, &gm, "C:\\TC\\BGI");
    setcolor(RED);
   circle(100,100,50);
   floodfill(100,100,RED);
    getch();
   closegraph();
 }
```

- In the above program a circle is drawn in RED color.
- Point (100,100) lies inside the circle as it is the center of circle, third argument to floodfill is RED which is color of boundary of circle.
- So the output of above program will be a circle filled with WHITE color as it is the default fill color.

setfillstyle function

- setfillstyle function sets the current fill pattern and fill color.

syntax :- ***void setfillstyle(int pattern, int color);***

Different fill styles:

```
enum fill_styles
{
  EMPTY_FILL,
  SOLID_FILL,
  LINE_FILL,
  LTSLASH_FILL,
  SLASH_FILL,
  BKSLASH_FILL,
  LTBKSLASH_FILL,
  HATCH_FILL,
  XHATCH_FILL,
  INTERLEAVE_FILL,
  WIDE_DOT_FILL,
  CLOSE_DOT_FILL,
  USER_FILL
};
```

/*C programming source code for setfillstyle*/

```
#include<graphics.h>
#include<conio.h>
void  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();
}
```