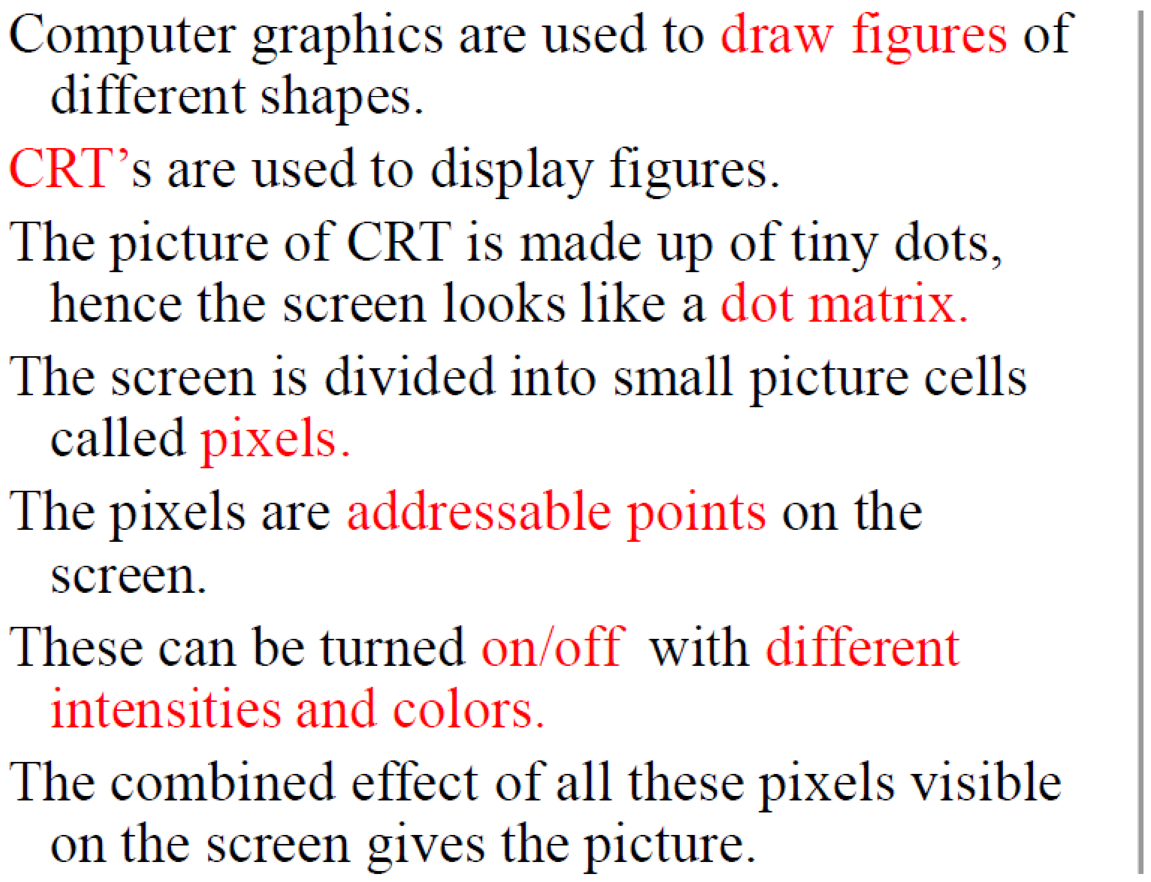
Expt no:1 Basic Graphics Functions

Aim: To get familiar with basic graphics functions

Theory: C graphics using graphics.h functions can be used to draw different shapes, display text in different fonts, change colors and many more. Using functions of graphics.h in Turbo C compiler you can make graphics programs, animations, projects, and games. You can draw circles, lines, rectangles, bars and many other geometrical figures. You can change their colors using the available functions and fill them.



Computer graphics are used to draw figures of different shapes.

CRT's are used to display figures.

The picture of CRT is made up of tiny dots, hence the screen looks like a dot matrix.

The screen is divided into small picture cells called pixels.

The pixels are addressable points on the screen.

These can be turned on/off with different intensities and colors.

The combined effect of all these pixels visible on the screen gives the picture.

The number of pixels available on screen in graphic mode gives its resolution.

If there are large number of pixels, there is high resolution.

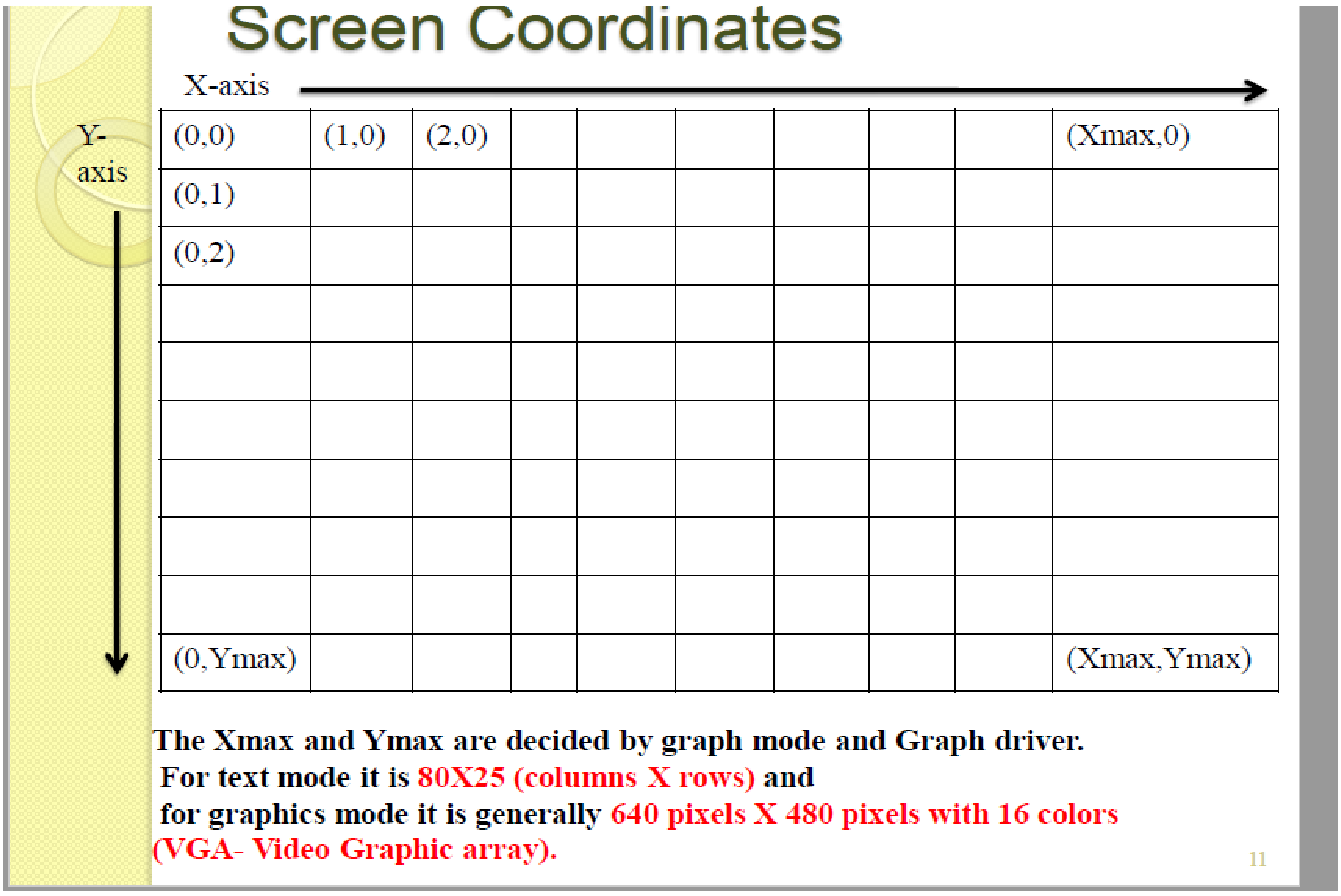
E.g.

Medium resolution = 320X200 pixels

High resolution= 640X200 pixels

Very high resolution = 640X350 pixels

Super high resolution= 1028X768 pixels



## Screen Coordinates

**The Xmax and Ymax are decided by graph mode and Graph driver.**
**For text mode it is 80X25 (columns X rows) and**
**for graphics mode it is generally 640 pixels X 480 pixels with 16 colors (VGA- Video Graphic array).**

## Graphics in C

While writing a C graphics program following points must be known.

1. Header file- A header file graphics.h or graphics.lib must be used in graphics program. The graphics.h contains definitions of constants and prototypes of graphics functions. The file graphics.lib is a library file which contains function definitions.

2. Graphic mode-There are two modes namely text and graphics mode.

In text mode only text display is possible in terms of ASCII, but in graphics mode any text /figures /shapes can be displayed.

selection of mode can be done by standard function initgraph().

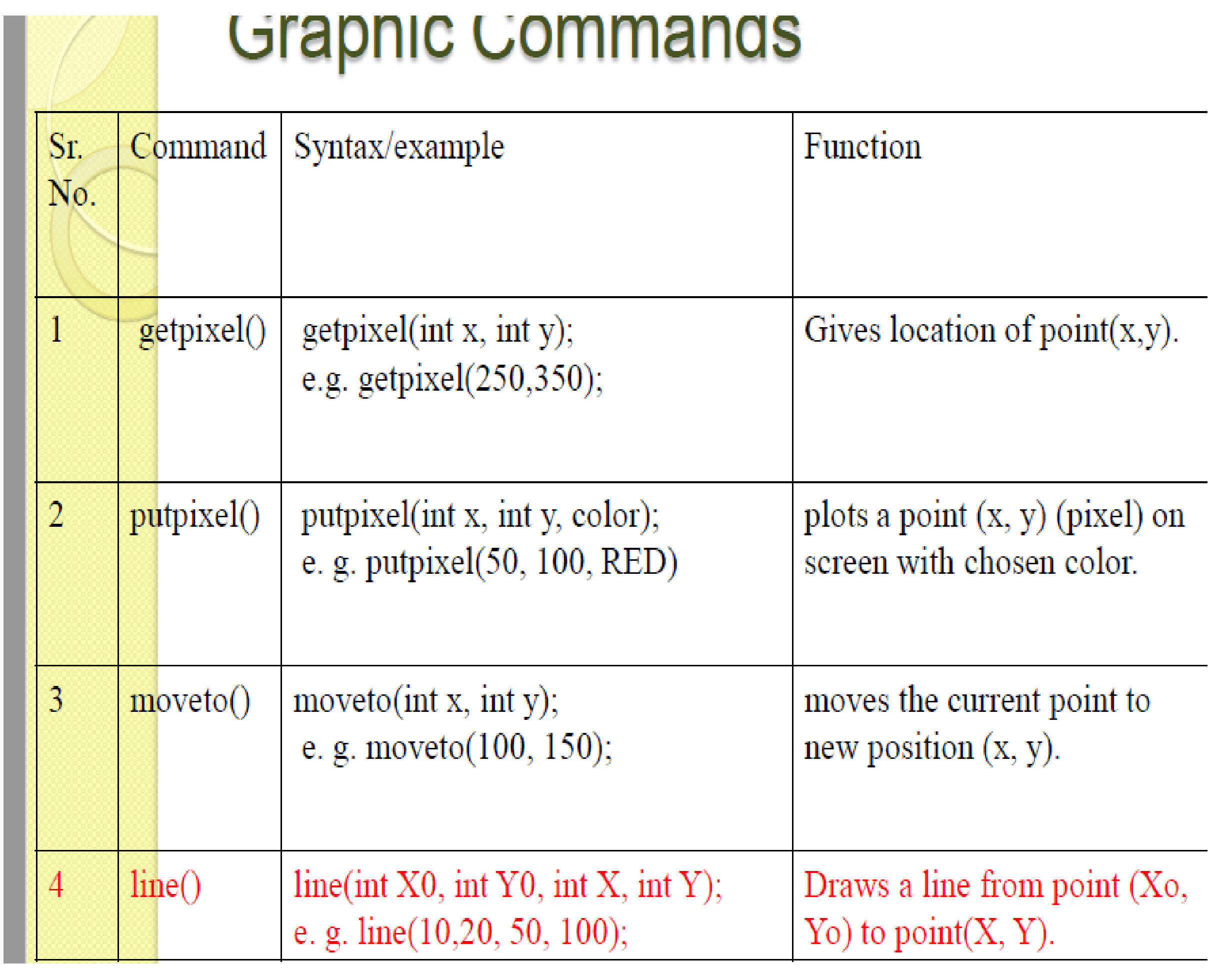
3. Graphic Drivers- Theses are the programs which communicate with specific devices.

These are applicable only in graphics mode. Also their selection depends upon graphic adapters used. The graphic drivers are stored in the files with extension *.bgi and can be detected with DETECT macro.

4. DETECT macro- It is a program which selects appropriate graphic driver. This value is stored in variable gd for further use.
5. Exiting graphic mode- The closegraph(); function is used to exit from graphic mode.
6. Restoring the text mode- The restorecrtmode(); function is used to restore the screen mode.

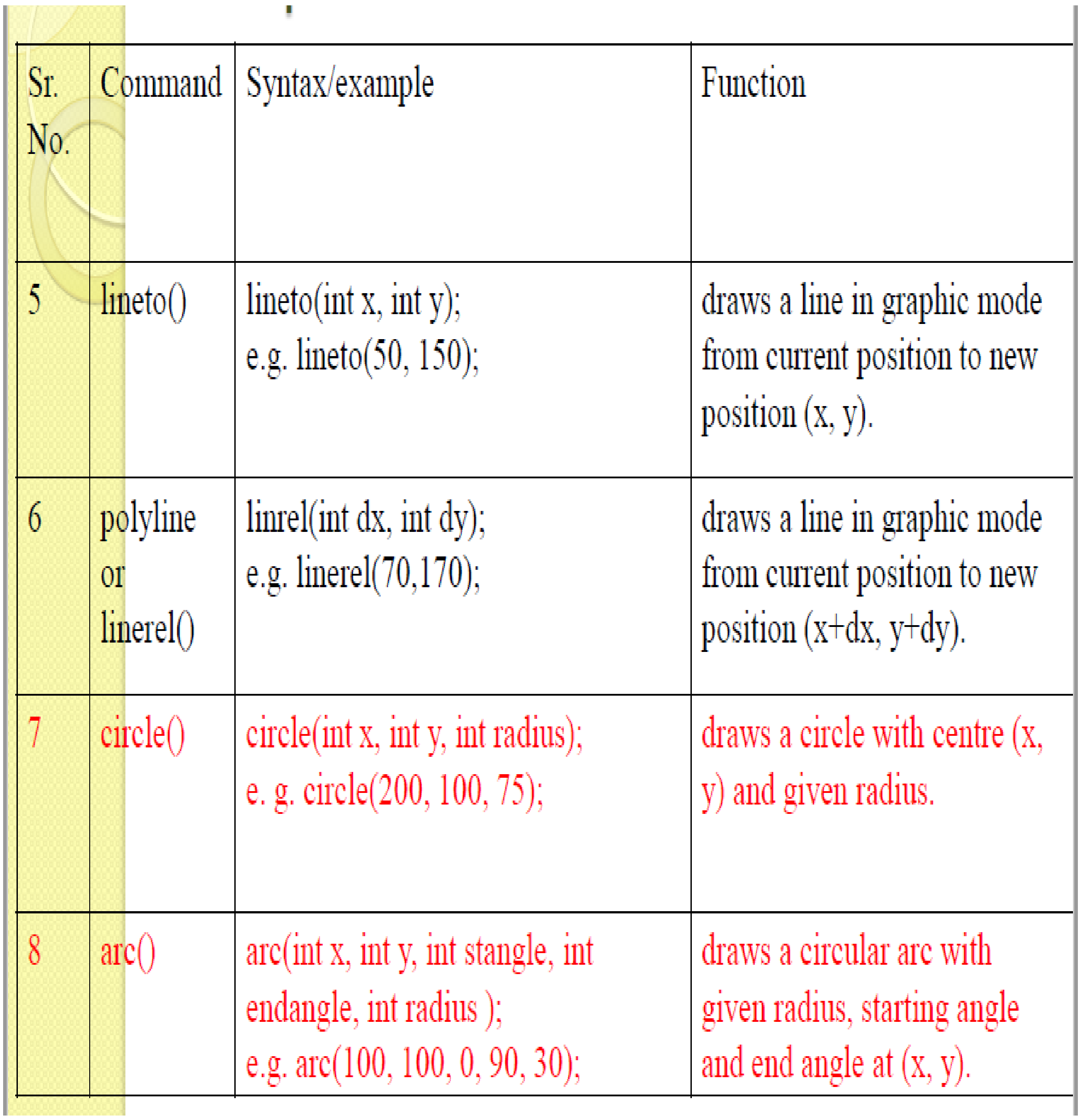
Procedure:

Following is a list of functions of graphics.h header file.



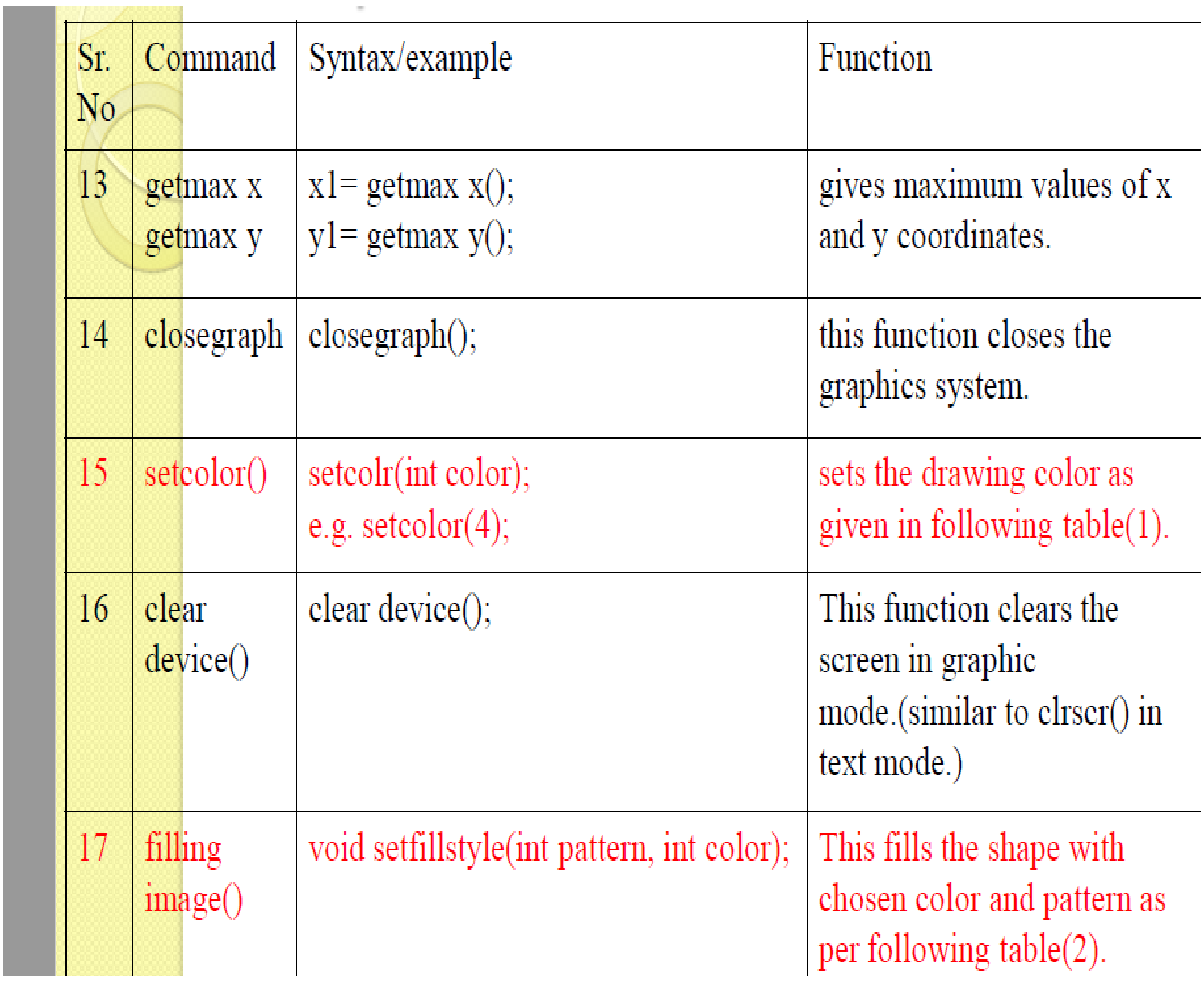
## Graphic Commands

| Sr. No. | Command | Syntax/example | Function |
|---|---|---|---|
| 1 | getpixel() | getpixel(int x, int y);<br>e.g. getpixel(250,350); | Gives location of point(x,y). |
| 2 | putpixel() | putpixel(int x, int y, color);<br>e. g. putpixel(50, 100, RED) | plots a point (x, y) (pixel) on screen with chosen color. |
| 3 | moveto() | moveto(int x, int y);<br>e. g. moveto(100, 150); | moves the current point to new position (x, y). |
| 4 | line() | line(int X0, int Y0, int X, int Y);<br>e. g. line(10,20, 50, 100); | Draws a line from point (Xo, Yo) to point(X, Y). |

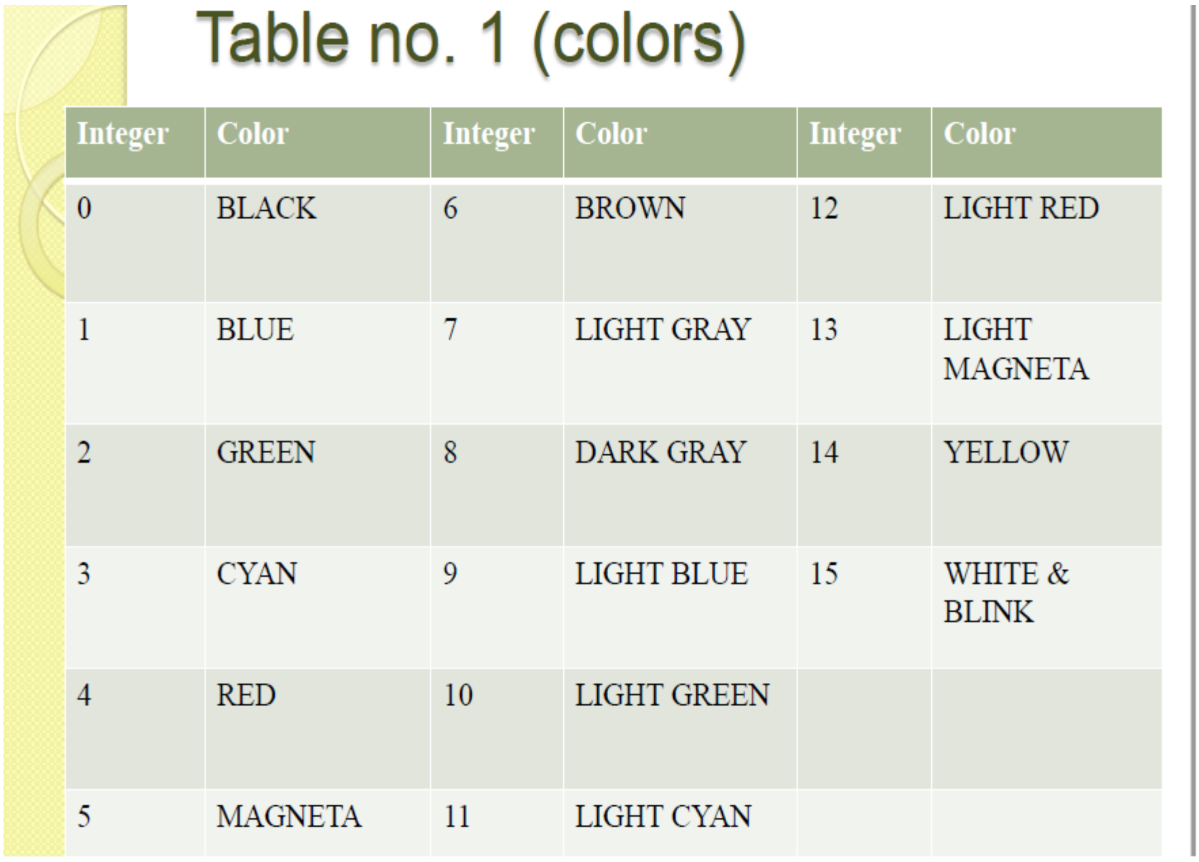


| Sr. No. | Command | Syntax/example | Function |
|---|---|---|---|
| 5 | lineto() | lineto(int x, int y);<br>e.g. lineto(50, 150); | draws a line in graphic mode from current position to new position (x, y). |
| 6 | polyline or linerel() | linrel(int dx, int dy);<br>e.g. linerel(70,170); | draws a line in graphic mode from current position to new position (x+dx, y+dy). |
| 7 | circle() | circle(int x, int y, int radius);<br>e. g. circle(200, 100, 75); | draws a circle with centre (x, y) and given radius. |
| 8 | arc() | arc(int x, int y, int stangle, int endangle, int radius );<br>e.g. arc(100, 100, 0, 90, 30); | draws a circular arc with given radius, starting angle and end angle at (x, y). |

| Sr. No. | Command | Syntax/example | Function |
|---|---|---|---|
| 9 | ellipse() | ellipse(int x, int y, int stangle, int endangle, int Xradius, int Yradius);<br>e.g. ellipse(150, 100, 0, 360, 75, 50); | draws an ellipse at centre (x, y) with given starting and angles and semi major /minor radius on x and y axis. |
| 10 | rectangle () | rectangle(int left, int top, int right, int bottom);<br>e.g. rectangle(100, 50, 150, 250); | draws a rectangle. Left and top gives first left top corner of rectangle and Right & bottom gives second corner point of rectangle. |
| 11 | bar() | bar(int x, int y, int x1, int y1);<br>bar(100, 50, 150, 250); | draws a filled bar with corner points (x, y) and (x1, y1) |
| 12 | drawpoly () | drawpoly(int points, int polypoints );<br>e.g.<br>int triangle[]={320, 100, 100, 250, 520, 250, 320, 100};<br>drawpoly(4, triangle); | draws outline of a polygon. the number of points are joined using coordinates. |

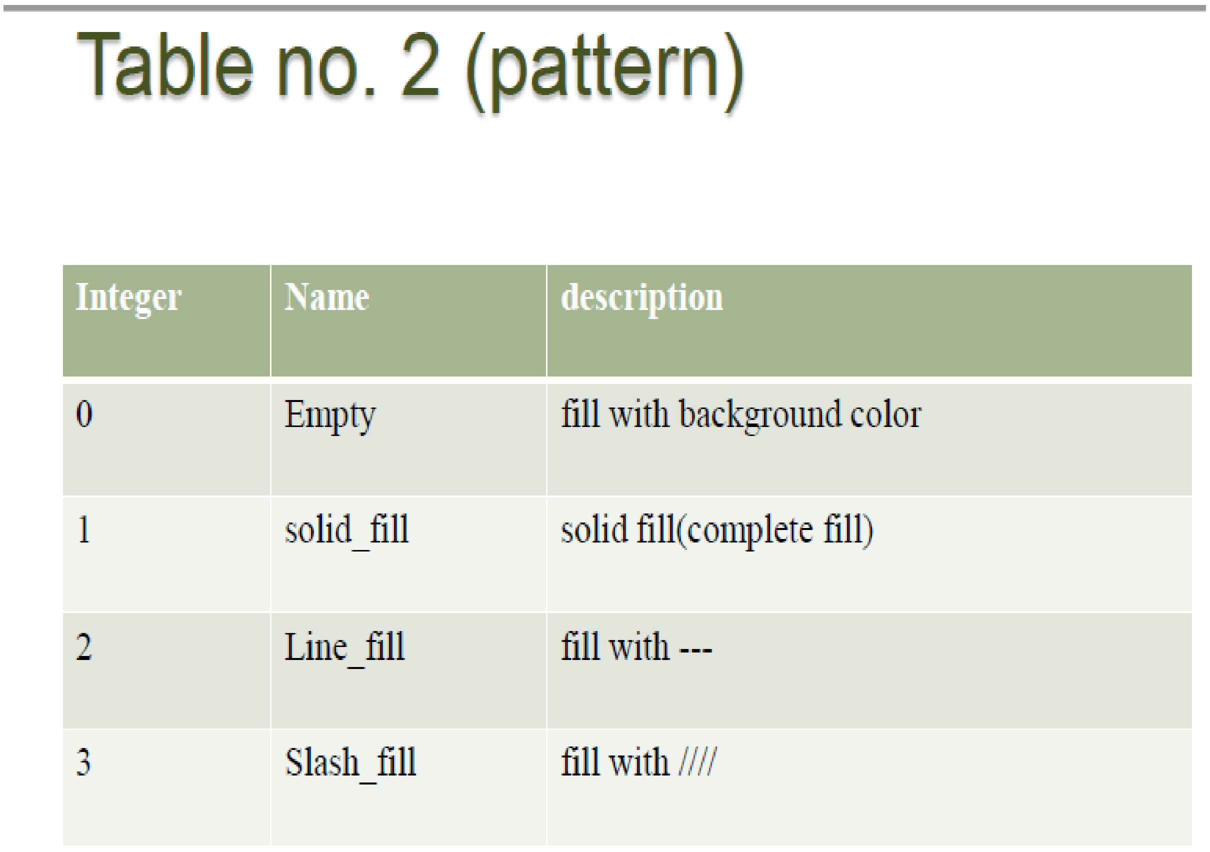


| Sr. No | Command | Syntax/example | Function |
|---|---|---|---|
| 13 | getmax x getmax y | x1= getmax x(); y1= getmax y(); | gives maximum values of x and y coordinates. |
| 14 | closegraph | closegraph(); | this function closes the graphics system. |
| 15 | setcolor() | setcolr(int color); e.g. setcolor(4); | sets the drawing color as given in following table(1). |
| 16 | clear device() | clear device(); | This function clears the screen in graphic mode.(similar to clrscr() in text mode.) |
| 17 | filling image() | void setfillstyle(int pattern, int color); | This fills the shape with chosen color and pattern as per following table(2). |



## Table no. 1 (colors)

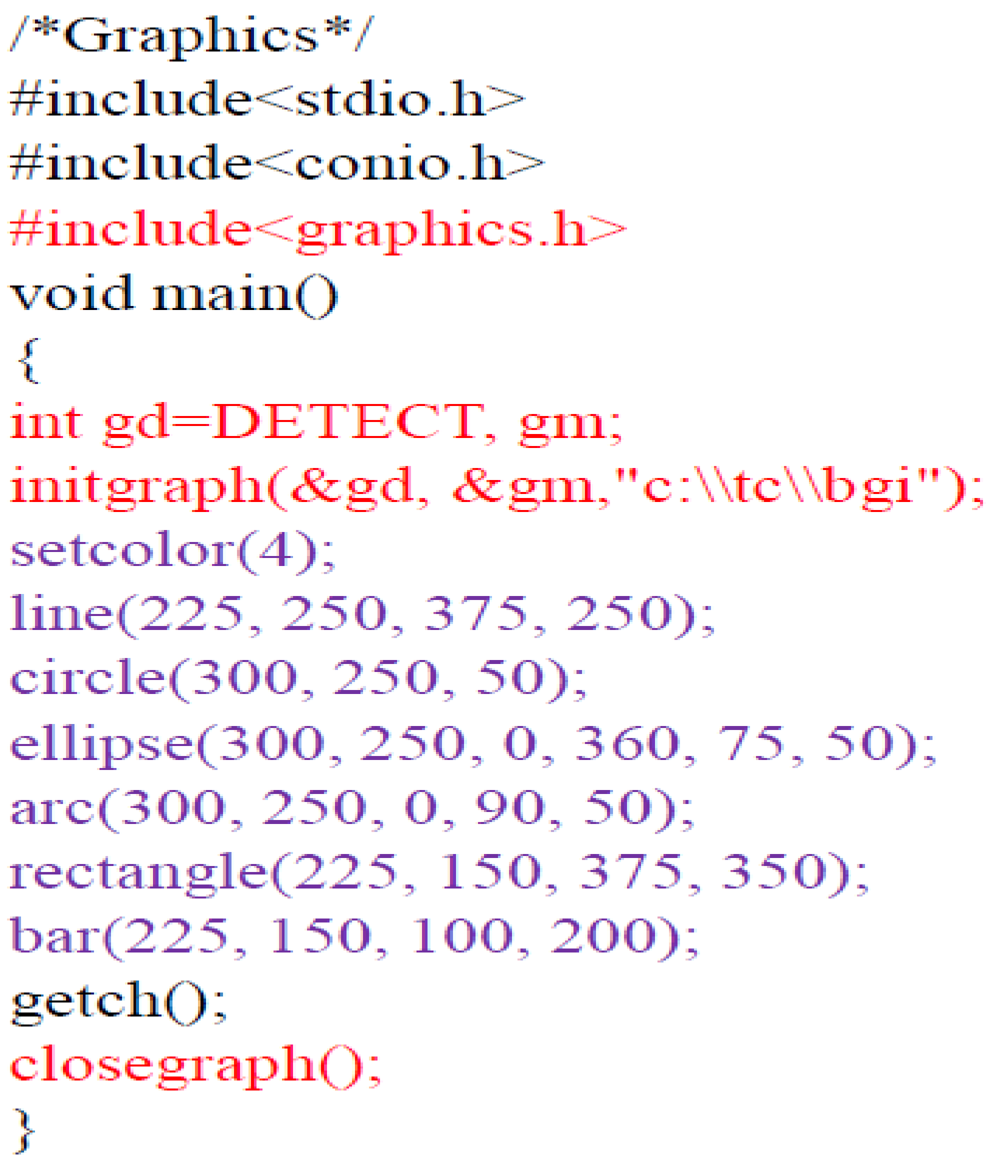
| Integer | Color | Integer | Color | Integer | Color |
|---|---|---|---|---|---|
| 0 | BLACK | 6 | BROWN | 12 | LIGHT RED |
| 1 | BLUE | 7 | LIGHT GRAY | 13 | LIGHT MAGNETA |
| 2 | GREEN | 8 | DARK GRAY | 14 | YELLOW |
| 3 | CYAN | 9 | LIGHT BLUE | 15 | WHITE & BLINK |
| 4 | RED | 10 | LIGHT GREEN | | |
| 5 | MAGNETA | 11 | LIGHT CYAN | | |



## Table no. 2 (pattern)

| Integer | Name | description |
|---|---|---|
| 0 | Empty | fill with background color |
| 1 | solid_fill | solid fill(complete fill) |
| 2 | Line_fill | fill with --- |
| 3 | Slash_fill | fill with //// |

## Ur First Program



```
/*Graphics*/
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
void main()
{
int gd=DETECT, gm;
initgraph(&gd, &gm,"c:\\tc\\bgi");
setcolor(4);
line(225, 250, 375, 250);
circle(300, 250, 50);
ellipse(300, 250, 0, 360, 75, 50);
arc(300, 250, 0, 90, 50);
rectangle(225, 150, 375, 350);
bar(225, 150, 100, 200);
getch();
closegraph();
}
```

## Graphic Initialization

The initgraph function is used to initialization of graphics.

Syntax

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

It has three arguments

gd- means graphic driver (gd=DETECT)

gm- means graphic mode (gm)

"    "- includes the driver path

(C:\\TC\\BGI) or null.

gd/graphdriver- is an integer that specifies the graphics driver to be used. One can give a constant value to the driver or can detect its value using DETECT macro.

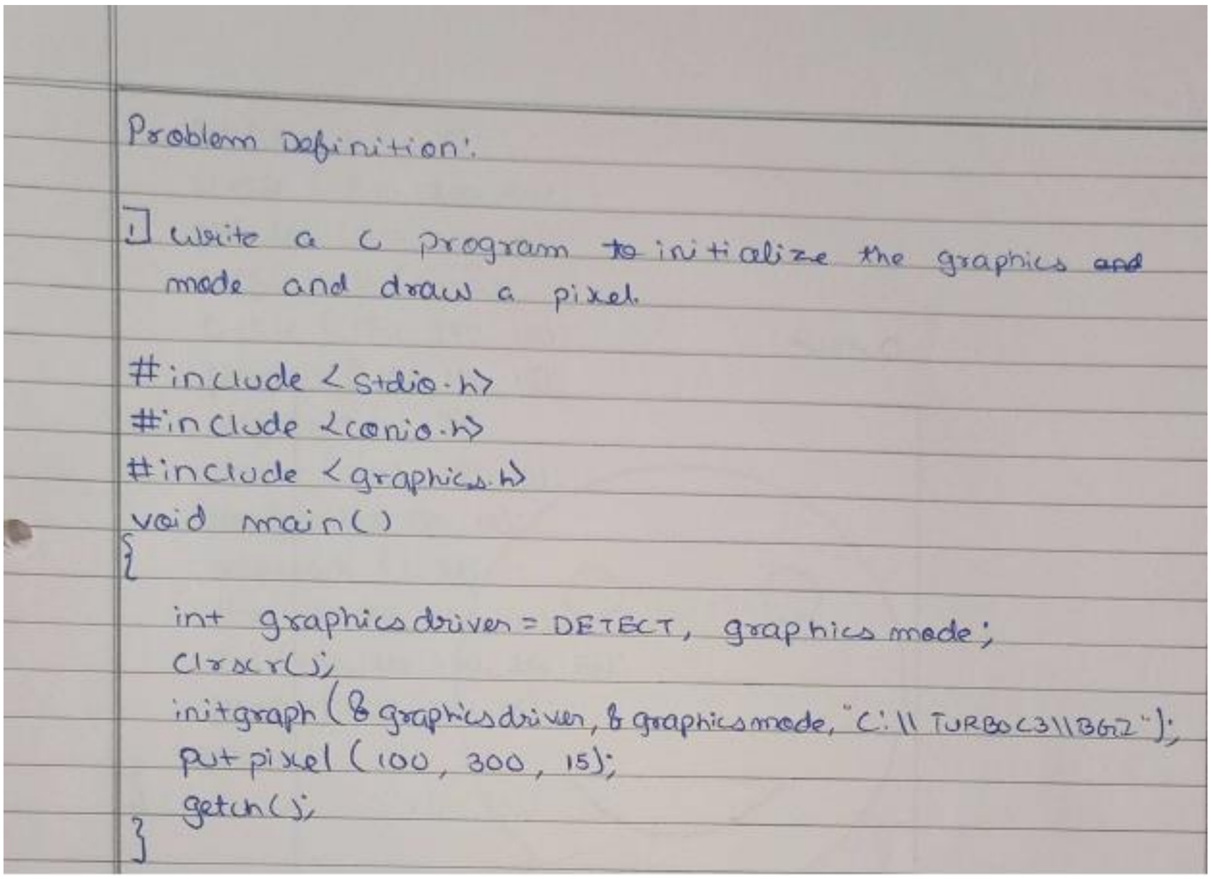
gm/graphmode- is an integer that specifies initial graphics mode. If gd=DETECT then grahics mode is set to highest resolution available for detected driver.

pathtodriver- it specifies the directory path where initgraph looks for graphic driver. If it's not there then it looks into current directory. If path driver is null, then driver files must be in current directory.

detectgraph() function- It detects which graphic adapter is fitted and its highest resolution mode.

---

**EXERCISE 1:**



Problem Definition:

1] Write a C program to initialize the graphics ~~and~~ mode and draw a pixel.

```
#include <stdio.h>
#include <conio.h>
#include <graphics.h>
void main()
{
  int graphicsdriver = DETECT, graphicsmode;
  clrscr();
  initgraph(&graphicsdriver, &graphicsmode, "C:\\TURBOC3\\BGI");
  putpixel(100, 300, 15);
  getch();
}
```

## EXERCISE 2:

setfillstyle() and floodfill() in C

The header file graphics.h contains setfillstyle() function which sets the current fill pattern and fill color. floodfill() function is used to fill an enclosed area. Current fill pattern and fill color is used to fill the area.

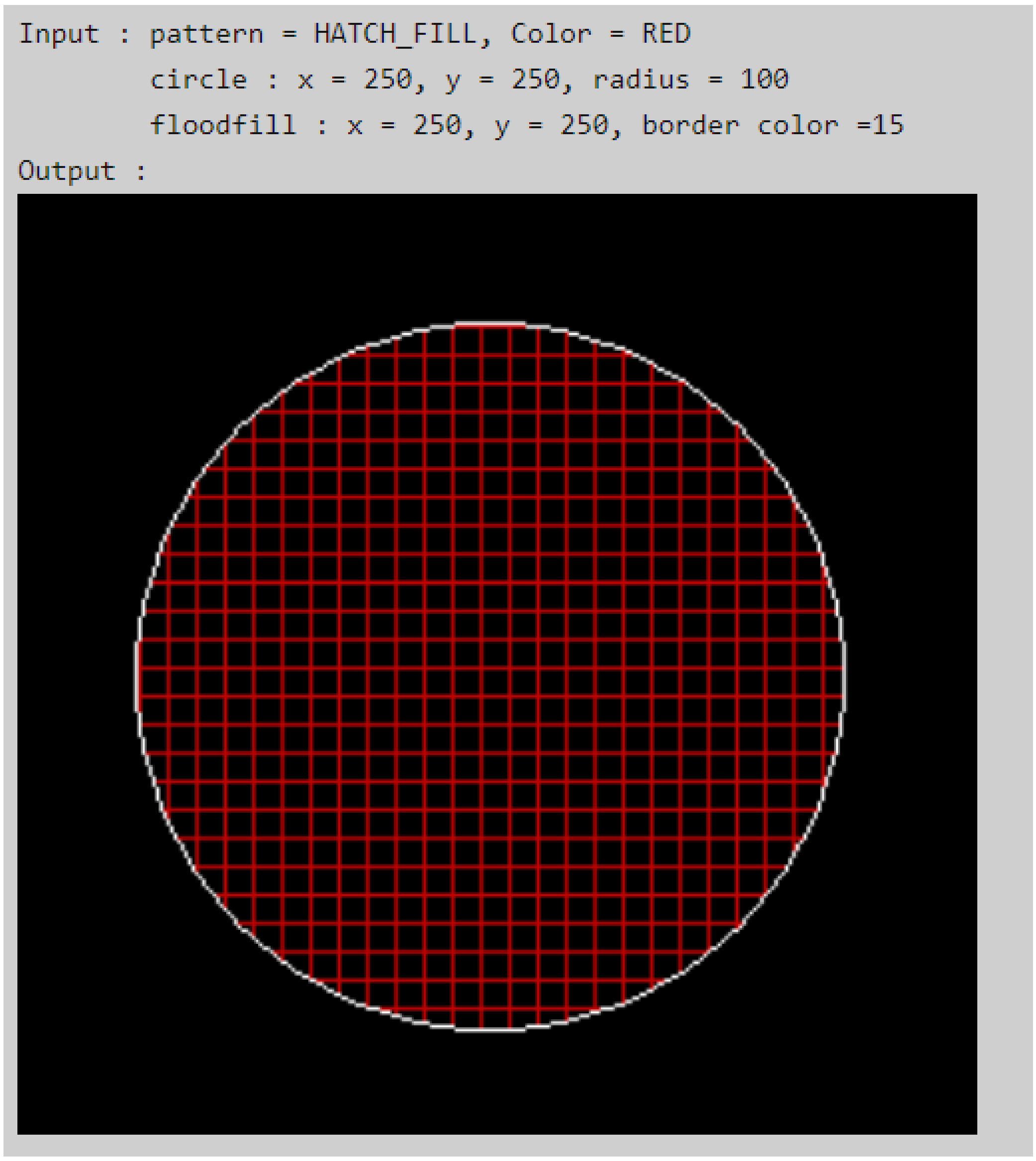
Syntax :

void setfillstyle(int pattern, int color)

void floodfill(int x, int y, int border_color)

Below is the table showing INT VALUES corresponding to Patterns :

| PATTERN | INT VALUES |
|---|---|
| EMPTY_FILL | 0 |
| SOLID_FILL | 1 |
| LINE_FILL | 2 |
| LTSLASH_FILL | 3 |
| SLASH_FILL | 4 |
| BKSLASH_FILL | 5 |
| LTBKSLASH_FILL | 6 |
| HATCH_FILL | 7 |
| XHATCH_FILL | 8 |
| INTERLEAVE_FILL | 9 |
| WIDE_DOT_FILL | 10 |
| CLOSE_DOT_FILL | 11 |
| USER_FILL | 12 |



```
Input : pattern = HATCH_FILL, Color = RED
        circle : x = 250, y = 250, radius = 100
        floodfill : x = 250, y = 250, border color =15
Output :
```

#include <graphics.h>

void main()

{

// gm is Graphics mode which is a computer display mode that generates image using pixels. DETECT is a macro defined in "graphics.h" header file

int gd = DETECT, gm;

```
    initgraph(&gd, &gm, " ");
      // center and radius of circle
    int x_circle = 250;
    int y_circle = 250;
    int radius=100;
       // setting border color
    int border_color = WHITE;
          // set color and pattern
    setfillstyle(HATCH_FILL,RED);
      circle(x_circle,y_circle,radius);
       floodfill(x_circle,y_circle,border_color);
      getch();
        // closegraph function closes the  graphics mode and
deallocates  all memory allocated by   graphics system
closegraph();
}
```

## **CONCLUSION :: ????**