# Graphics

**1. Write the purpose of the following functions with parameters:**

`initgraph( ):`
**Purpose:**
Initializes the graphics system.
**Syntax:**

```
Void initgraph(int *gd, int *gm, char *path);
```

*gd is an integer that specifies the graphics driver to be used. *gm is an integer that specifies the initial graphics mode.

`circle( ):`
**Purpose:**
`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 radius);
```

`line( ):`
**Purpose:**
`line( )` function is used to draws a line between two specified points.
**Syntax:**

```
Void line(int x1, int y1, int x2, int y2);
```

`putpixel( ):`
**Purpose:**
`putpixel( )` function is used to plots a pixel at a specified point in specified color.
**Syntax:**

```
void putpixel(int x, int y, int color);
```

`floodfill( ):`
**Purpose:**
`floodfill( )` function is used to flood fills the bounded region.
**Syntax:**

```
void floodfill(int x, int y, int border);
```

`outtextxy( ):`

**Purpose:**

`outtextxy( )` function is used to displays a text or string at a specified point(x,y) on the screen.

**Syntax:**

```
void outtextxy(int x, int y, char *string);
```

`setfillstyle( ):`

**Purpose:**

`setfillstyle( )` function sets the current fill pattern and fill color.

**Syntax:**

```
void setfillstyle( int pattern, int color);
```

`settextstyle( ):`

**Purpose:**

`settextstyle( )` function sets the current fill pattern and fill color.

**Syntax:**

```
void settextstyle( int font, int direction, int charsize);
```

`closegraph( ):`

**Purpose:**

Shuts down the graphics system and de-allocates all memory allocated by the graphics system.

**Syntax:**

```
void closegraph( );
```

`rectangle( ):`

**Purpose:**

Draws a rectangle.

**Syntax:**

```
void rectangle(int left, int top, int right, int bottom);
```

`bar( ):`

**Purpose:**

Draws a two-dimensional bar.

**Syntax:**

```
void bar (int left, int top, int right, int bottom);
```

`arc( ):`
**Purpose:**
Draws a circular arc.
**Syntax:**
```
void arc(int x, int y, int start_angle, int end_angle, int rad);
```

`cleardevice( ):`
**Purpose:**
Clears the graphics screen and moves the CP to home (0,0).
**Syntax:**
```
void cleardevice( );
```

`detectgraph( ):`
**Purpose:**
Determines the graphics driver and graphics mode to use by checking hardware.
**Syntax:**
```
void detectgraph(int *gd, int *gm);
```

`getcolor( ):`
**Purpose:**
Returns the current drawing color.
**Syntax:**
```
int getcolor( );
```

`getmaxx( ):`
**Purpose:**
Returns maximum x screen co-ordinate in the current mode.
**Syntax:**
```
int getmaxx( );
```

`getmaxy( ):`
**Purpose:**
Returns maximum y screen co-ordinate in the current mode.
**Syntax:**
```
int getmaxy( );
```

`getpixel( ):`
**Purpose:**
Gets the color of a specified pixel.
**Syntax:**

```
unsigned getpixel(int x, int y);
```

`settextjustify( ):`
**Purpose:**
Sets text justification for graphics function.
**Syntax:**

```
void settextjustify(int horiz, int vert);
```

## Initiate graphics in C:

To initialize graphics mode we use `initgraph` function in our program. `initgraph` function is present in "`graphics.h`" header file, so your every graphics program should include "`graphics.h`" header file.

```
#include<graphics.h>
#include<conio.h>

void main(){
int gd = DETECT, gm;

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

getch();

closegraph();
}
```

This program initializes graphics mode and then closes it after a key is pressed. To begin with we have declared two variables of `int` type `gd` and `gm` for graphics driver and graphics mode respectively. `DETECT` is a macro defined in "`graphics.h`" header file, then we have passed three arguments to `initgraph` function to initialize graphics mode, first is the address of `gd`, second is the address of `gm` and third is the path where your BGI files are present. `getch` helps us to wait until a key is pressed, `closegraph` function closes the graphics mode.