EXPERIMENT 01

(a) Study and enlist the basic function used for graphics in c language. Give example for each.

(b) Draw a coordinate axis at the centre of screen.

Answer:-

(a) 1. **CIRCLE():**

Declaration : void circle(int x, int y, int radius);

Theory: The header file graphics.h contains circle() function which draws a circle with center at (x,y) and given radius.

Example: circle(100,200,50);

2. **ARC()**:

Declaration : void arc(int x, int y, int startangle, int endangle, int radius);

Theory: The header file graphics.h contains arc() function which draws an arc with centre at (x, y) and given radius. startangle is the starting point of angle and endangle is the ending point of the angle. The value of the angle can vary from 0 to 360 degree.

Example: arc(60,60,0,180,30);

3. **LINE()**:

Declaration : void line(int x1, int y1, int x2, int y2);

Theory: The header file graphics.h contains line() function which draws a line from one point to another. The x1 and y1 gives the value of pixel from where the line start in x-coordinate and y-coordinate respectively. Similarly , the x2 and y2 gives the va,ue of pixel where the line ends.

Example: line(50,50,150,150);

4. **RECTANGLE()**:

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

Theory:The header file graphics.h conatin rectangle() function which 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.

Example: rectangle(200,200,400,400);

5. **ELLIPSE()**:

Declaration: void ellipse(int x, int y, int start_angle, int end_angle, int x_radius, int y_radius)

Theory: The header file graphics.h contains ellipse() function which draw a ellipse. In this function x, y is the location of the ellipse. x_radius and y_radius decide the radius of form x and y.start_angle is the starting point of angle and end_angle is the ending point of angle. The value of angle can vary from 0 to 360 degree.

Example: ellipse(200,200,0,360,30,40);

6. **<u>OUTTEXTXY()</u>**:

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

Theory:The header file graphics.h contains outtextxy() function which displays the text or string at a specified point (x, y) on the screen.

Example: outtextxy(100,100,"hello");

7. **GETMAXX /GETMAXY:**

Declaration: int getmaxx();

int getmaxy();

Theory:The header file graphics.h contains getmaxx() and getmaxy() function returns the maximum X coordinate and maximum Y coordinate respectively for current graphics mode and driver.

Example: x=getmaxx();

Y=getmaxy();

8. **<u>GETMINX() / GETMINY()</u>**:

Declaration: int getminx();

int getminy();

Theory: The header file graphics.h contains getminx() and getminy() function returns the minimum X coordinate and minimum Y coordinate respectively for current graphics mode and driver.

Example: x=getminx();

Y=getminy();

9. <u>SETCOLOR() / SETBKCOLOR():</u>

Declaration: void setcolor(int color);

void setbkcolor(int color);

Theory:The header file graphics.h contains setcolor() function and setbkcolor() function is used to set the current drawing color to the new color and set the new color to the background respectively.

Example: setcolor(4);

setbkcolor(1);

10. CLEARDEVICE():

Declaration: void cleardevice();

Theory:The header file graphics.h contains cleardevice() function which 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 color.

Example: cleardevice();

11. **CLOSEGRAPH():**

Declaration: void closegraph();

Theory:The header file graphics.h contains closegraph() function which 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.

Example: closegraph();

12. **PUTPIXEL():**

Declaration: void putpixel(int x, int y, int color);

Theory:The header file graphics.h contains putpixel() function which plots a pixel at location (x, y) of specified color.

Example: putpixel(85,85);

13. **FLOODFILL():**

Declaration: void floodfill(int x, int y, int border_color);

Theory: The header file graphics.h contains floodfill() function is used to fill an enclosed area. Current fill pattern and fill color is used to fill the area.

Example: floodfill(50,50,1);

14. **SETFILLSTYLE():**

Declaration: void setfillstyle(int pattern, int color)

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

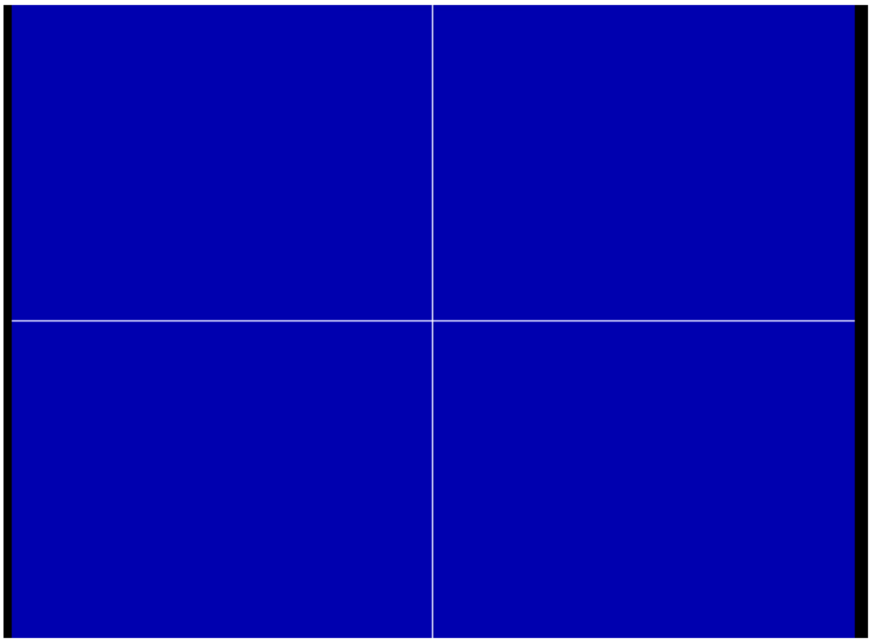
Example: setfillstyle(HATCH_FILL,RED);

(b) CODE:

```
#include<stdio.h>

#include<conio.h>
```

```
  #include<graphics.h>
void main()
{
int gd = DETECT, gm,x,y;
initgraph(&gd, &gm, “c:\\turboc3\\bgi”);
setbkcolor(1);
setcolor(15);
x= getmaxx();
y= getmaxy();
line(0,y/2,x,y/2);
line(x/2,0,x/2,y);
getch();
closegraph();
}
```

OUTPUT:



EXPERIMENT 02

(a) Divide ypur screen into region draw circle , rectangle, ellipse, half ellipse, square and concentric circle in each region with appropriate message.

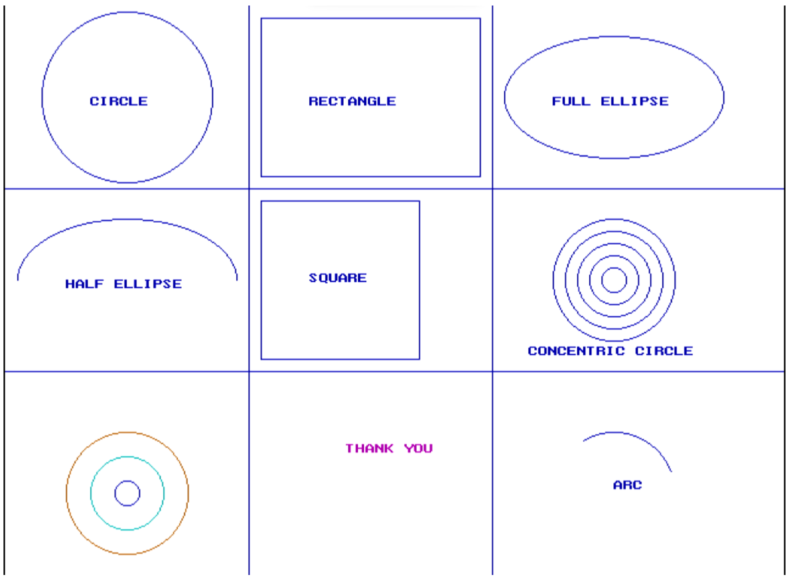
(b) Draw a simple hut on screen.

ANSWER:-

(a) CODE:

```
#include<stdio.h>

#include<conio.h>

#include<graphics.h>

void main()

{

int gd=DETECT ,gm;

initgraph(&gd, &gm, "c:\\turboc3\\bgi");setbkcolor(15);

setcolor(1);

line(200,0,200,600);

line(400,0,400,600);

line(0,150,800,150);

line(0,300,800,300);

circle(100,75,70);

outtextxy(70,75,"CIRCLE");

rectangle(210,10,390,140);

outtextxy(250,75,"RECTANGLE");

ellipse(500,75,0,360,90,50);

outtextxy(450,75,"FULL ELLIPSE");

ellipse(100,225,0,180,90,50);

outtextxy(50,225,"HALF ELLIPSE");

rectangle(210,160,340,290);

outtextxy(250,220,"SQUARE");

circle(500,225,10);
circle (500,225,20);
circle(500,225,30);
circle(500,225,40);
circle(500,225,50);
outtextxy(430,280,"CONCENTRIC CIRCLE");
setcolor(1);
circle(100,400,10);
setcolor(3);
circle(100,400,30);
setcolor(6);
circle(100,400,50);
setcolor(5);
outtextxy(280,360,"THANK YOU");
setcolor(1);
arc(500,400,20,120,50);
outtextxy(500,390,"ARC");
getch();
closegraph();
}
```

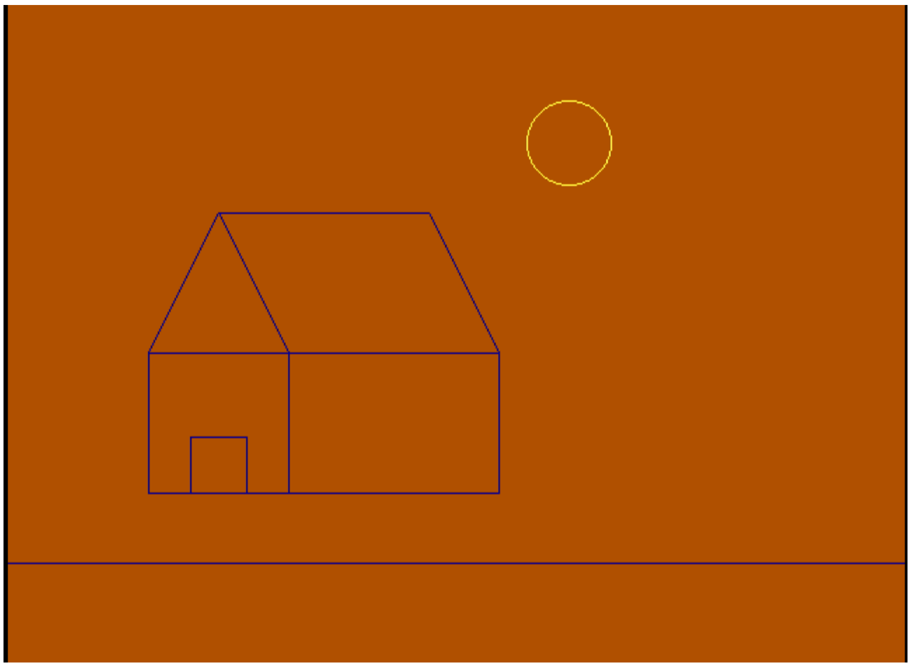
OUTPUT:



(b) CODE:

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
void main()
{
int gd=DETECT,gm;
initgraph(&gd, &gm, "c:\\turboc3\\bgi");
setbkcolor(10);
setcolor(BLUE);
line(150,150,100,250);
line(150,150,200,250);
line(100,250,200,250);
line(100,250,100,350);
line(100,350,200,350);
line(200,350,200,250);
line(150,150,300,150);
line(300,150,350,250);
line(130,350,130,310);
line(130,310,170,310);
line(170,310,170,350);
line(200,250,350,250);
line(350,250,350,350);
line(200,350,350,350);
setcolor(YELLOW);
circle(400,100,30);
setcolor(BLUE);
line(0,400,700,400);
getch();
closegraph();
return 0;
}
```

OUTPUT:



EXPERIMENT 03

Q. Develop the program for DDA line drawing algorithm.

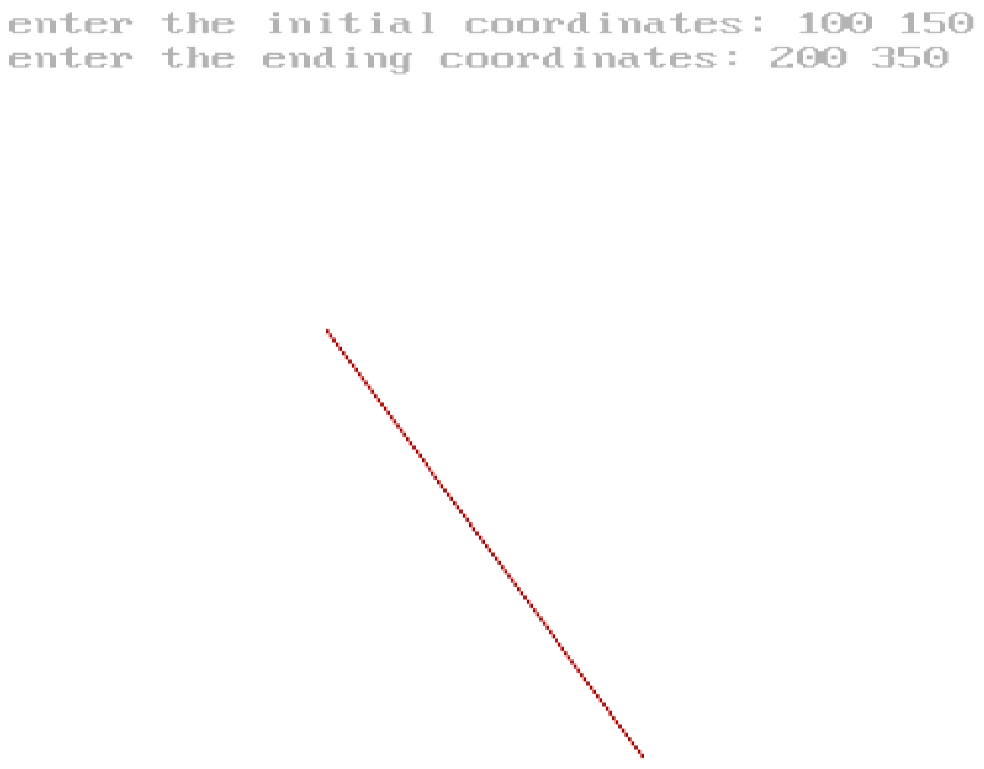
ANSWER:-

CODE:

```
#include<graphics.h>

#include<conio.h>

#include<stdio.h>

void main()

{

   int gd = DETECT ,gm, i;

   float x, y,dx,dy,steps;

   int x0, x1, y0, y1;

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

   setbkcolor(WHITE);

   setcolor(1);

   printf("enter the initial coordinates: ");

   scanf("%d%d", &x0, &y0);

   printf("enter the ending coordinates: ");

   scanf("%d%d", &x1,&y1);

   dx = (float)(x1 - x0);

   dy = (float)(y1 - y0);

   if(dx>=dy)

            {

          steps = dx;

   }

   else{

          steps = dy;

   }

   dx = dx/steps;
```

```
    dy = dy/steps;
    x = x0;
    y = y0;
    //i =1;
    for(i=1; i<=steps;i++)
    //while(i<= steps)
    {
            putpixel(x, y, RED);
            x += dx;
            y += dy;
            delay(30);
    }
    getch();
    closegraph();
}
```

OUTPUT:



```
enter the initial coordinates: 100 150
enter the ending coordinates: 200 350
```

EXPERIMENT 04

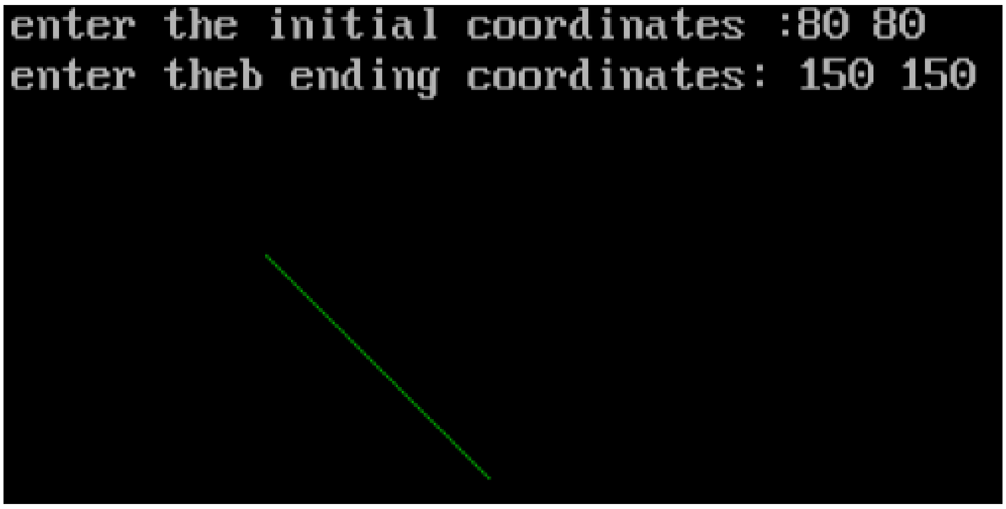
Q. Develop the program for Bresenham line drawing algorithm.

Answer:-

CODE:

```
#include<stdio.h>
 #include<conio.h>
 #include<graphics.h>
 void main()
 {
 int i, gd=DETECT, gm, x1, y1,x2,y2,dx,dy,pk;
 do{
 initgraph(&gd,&gm,"c:\\turboc3\\bgi");
 printf("enter the initial coordinates :");
 scanf ("%d%d:", &x1,&y1);
 printf("enter theb ending coordinates: ");
 scanf("%d%d", &x2, &y2);
cleardevice();
dx=x2-x1;
 dy=y2-y1;
 pk=2*dy-dx;
 do{
 putpixel(x1,y1,GREEN);
 if(pk<0){
 pk+=2*dy;
 }
 else{
 ++y1;
 pk+=2*(dy-dx);
 }
 ++x1;
 }while(x1<=x2);
 printf("press y for try again ... ");
}while(getch()=='y');
 }
```

OUTPUT:



```
enter the initial coordinates :80 80
enter theb ending coordinates: 150 150
```

EXPERIMENT 05

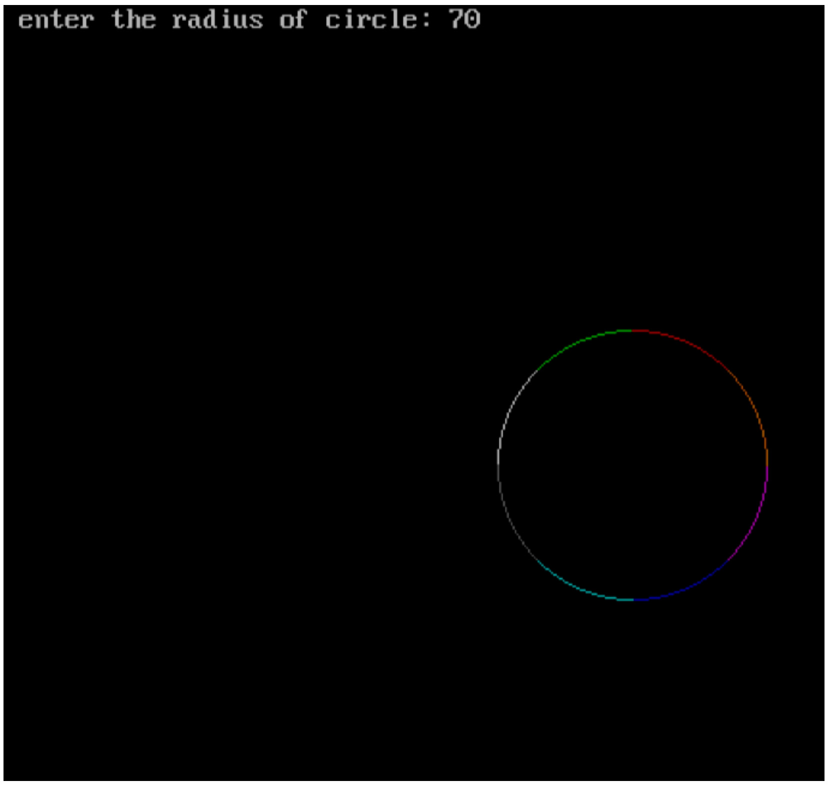
Q. Develop the program for mid-point circle drawing algorithm.

Answer:-

CODE:

```
#include<stdio.h>

#include<conio.h>

#include<graphics.h>

void main()

{

int gd=DETECT, gm=DETECT;

int r,x,y,midx,midy;

float p;

clrscr();

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

printf("enter the radius of circle: ");

scanf("%d", &r);

x=0;

y=r;

p=1.25-r;

midx=getmaxx()/2;

midy=getmaxy()/2;

do{

putpixel(midx+x,midy+y,1);

putpixel(midx-x,midy-y,2);

putpixel(midx-x, midy+y, 3);

putpixel(midx+x, midy-y, 4);

putpixel(midx+y, midy+x, 5);

putpixel(midx+y, midy-x, 6);

putpixel(midx-y, midy-x, 7);
putpixel(midx-y, midy+x, 8);
delay(20);
if(p<0){
x++;
p=p+2*x+1;
}
else{
x++;
y--;
p=p+2*(x-y)+1;
}
}while(x<=y);
getch();
closegraph();
}
```

OUTPUT:



```
enter the radius of circle: 70
```

EXPERIMENT 06

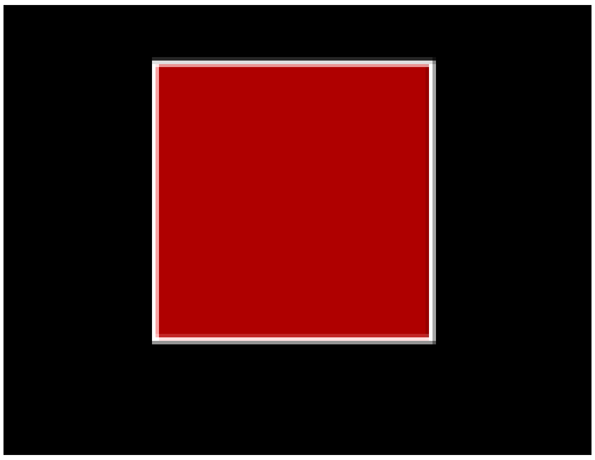
Q. Write a program to fill a rectangle using boundary fill algorithm.

Answer:-

CODE:
```
#include<stdio.h>

#include<conio.h>

#include <graphics.h>

void boundaryFill8(int x, int y, int fill_color,int boundary_color)

{

   if(getpixel(x, y) != boundary_color &&

      getpixel(x, y) != fill_color)

   {

         putpixel(x, y, fill_color);

         boundaryFill8(x + 1, y, fill_color, boundary_color);

         boundaryFill8(x, y + 1, fill_color, boundary_color);

         boundaryFill8(x - 1, y, fill_color, boundary_color);

         boundaryFill8(x, y - 1, fill_color, boundary_color);

         boundaryFill8(x - 1, y - 1, fill_color, boundary_color);

         boundaryFill8(x - 1, y + 1, fill_color, boundary_color);

         boundaryFill8(x + 1, y - 1, fill_color, boundary_color);

         boundaryFill8(x + 1, y + 1, fill_color, boundary_color);

   }

}


void main()

{

   int gd = DETECT, gm;

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

   rectangle(50, 50, 100, 100);

   boundaryFill8(55, 55, 4, 15);
```

```
    delay(30);
    getch();
    closegraph();
}
```

OUTPUT:



EXPERIMENT 07

Q. Write a program to implement 2D translation and scaling.

Answer:-

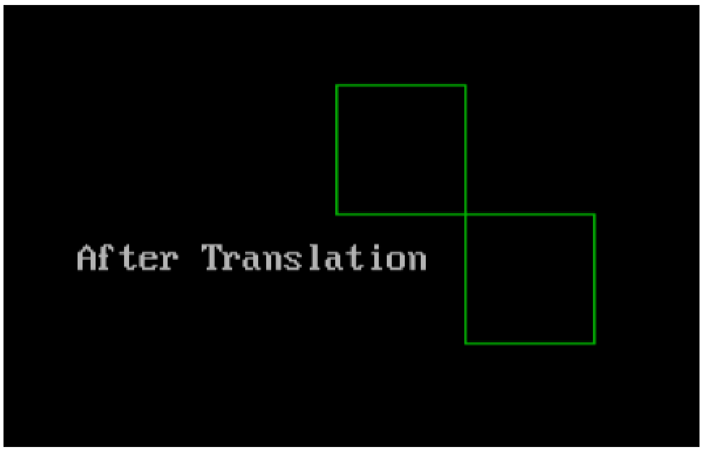
CODE:

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

void translate();
void scale();

void main()
{
int ch;
int gd=DETECT,gm;
initgraph(&gd,&gm,"c:\\turboc3\\bgi");

setcolor(6);
outtextxy (100,88,"Object.");
rectangle(100,150,150,100);

printf("---MENU---");
printf("\n 1)Translate\n 2)Scale\n ");
printf("\nEnter your choice: ");
scanf("%d",&ch);
cleardevice();

switch(ch)
```

```
{
case 1: translate();
break;
case 2: scale();
break;
default: printf("you have enterd wrong choice");
break;
}
getch();
closegraph();
}

void translate()
{
int tx,ty;
setcolor(2);
outtextxy(240,10,"TRANSLATION");
outtextxy(238,20," ----------");
printf("\nEnter tx: ");
scanf("%d",&tx);
printf("\nEnter ty: ");
scanf("%d",&ty);
cleardevice();
rectangle(100,150,150,100);
printf("\nAfter Translation");
rectangle(100+tx,150+ty,150+tx,100+ty);
}

void scale()
```

```
{
int sx,sy;
setcolor(2);
outtextxy(240,10,"SCALING");
outtextxy(238,20," ------ ");
printf("\nEnter sx: ");
scanf("%d",&sx);
printf("\nEnter sy: ");
scanf("%d",&sy);
cleardevice();
rectangle(100,150,150,100);
printf("\nAfter Scaling");
rectangle(100*sx,150*sy,150*sx,100*sy);
}
```

OUTPUT:



```
After Translation
```

<u>EXPERIMENT 08</u>

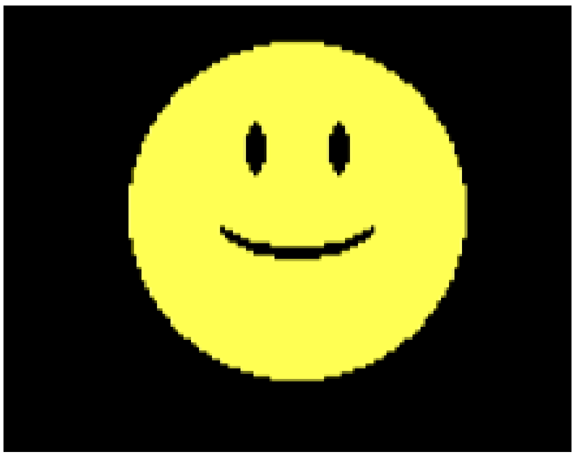
Q. Perform smiling face animation using graphics function.

Answer:-

CODE:

```
#include<stdio.h>

#include<conio.h>

#include<graphics.h>

void main()

{


int gd= DETECT, gm;

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


setbkcolor(3);

setcolor(15);

setcolor(YELLOW);



        circle(300, 100, 40);

        setfillstyle(SOLID_FILL, YELLOW);

        floodfill(300, 100, YELLOW);



        setcolor(BLACK);

        setfillstyle(SOLID_FILL, BLACK);



        fillellipse(310, 85, 2, 6);

        fillellipse(290, 85, 2, 6);
        ellipse(300, 100, 205, 335, 20, 9);
        ellipse(300, 100, 205, 335, 20, 10);
        ellipse(300, 100, 205, 335, 20, 11);
getch();
closegraph();
}
```

OUTPUT:



EXPERIMENT 09

Q. Draw a moving car on screen.

Answer:-

CODE:

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<dos.h>
void main()
{
int gd=DETECT,gm, i;
initgraph(&gd,&gm,"c:\\turboc3\\bgi");
setbkcolor(1);
setcolor(15);

for( i=0;i<600;i++){
line(0,470,getmaxx(),470);
circle(100+i,440,30);
circle(200+i,440,30);
line(0+i,440,70+i,440);
line(130+i,440,170+i,440);
line(230+i,440,290+i,440);
line(290+i,440,290+i,390);
line(0+i, 440, 0+i, 390);
line(0+i, 390, 70+i,390);
line(220+i,390,290+i, 390);
line(70+i,390, 70+i, 340);
line(220+i, 390, 220+i, 340);
line(70+i,340, 220+i,340);
delay(10);
cleardevice();
}
getch();
closegraph();
}
```

OUTPUT: