# Geetanjali Institute of Technical Studies

**(Approved by AICTE, New Delhi and Affiliated to Rajasthan Technical University Kota (Raj.))**

**DABOK, UDAIPUR, RAJASTHAN 313022**

## DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

## B. Tech - V SEMESTER


**ACADEMIC YEAR – 2024-25**

# COMPUTER GRAPHICS & MULTIMEDIA TECHNIQUES

(5CS4-21)

**Submitted to:** Ms.Somya Agrawal

**Submitted by:**

**Roll No.:**

# VISION & MISSION OF INSTITUTE

### INSTITUTE VISION

To achieve excellence in technical and management education through quality teaching and innovation.

### INSTITUTE MISSION

**M1:** To provide an excellent learning environment to produce socially responsible and productive technical professionals.

**M2:** To set up the state-of-the-art facilities for quality education and innovation.

**M3:** To impart knowledge & Skills leading to shaping a budding manager as a quality executive.

**M4:** To encourage for life-long learning and team-based problem solving through learning environment.

# VISION & MISSION OF DEPARTMENT

### VISION

To nurture the students to become employable graduates who can provide solutions to the societal issues through ICT.

### MISSION

M1:To focus on practical approach towards learning and exposing the students on the latest ICT technologies.

M2:To foster logical thinking among the students to solve real-time problems using innovative approaches.

M3:To provide state-of-the-art resources that contributes to inculcate ethical and life-long learning environment.

## COURSE OUTCOMES (COs)

| | |
|---|---|
| **CO1** | Students will be able to understand and explain the mathematical and practical principles of Computer Graphics Eg: To draw basic objects like lines, Circles and Ellipse. |
| **CO2** | Students will be able to apply concept of geometric, mathematical and algorithmic concepts necessary for programming computer graphics |
| **CO3** | Students will be able to apply the comprehension of windows, clipping and view-ports object representation in relation to images displayed on screen. |
| **CO4** | Students will be able to use matrix algebra in computer graphics and implement fundamental algorithms and transformations involved in viewing models. |

## TEXT / REFERENCE BOOKS

- Computer Graphics, C version by Donald Hearn and M. Paulline Baker.
- Computer Graphics: Principles and Practices by Jhon Hughes, Andries van Dam and Morgan.
- Fundamental of Computer Graphics by Steve Marshner and Peter Shirley.
- Computer Graphics by Xiang, McGraw Hill Publisher.
- Computer Graphics by Apurva A. Desai.

## VIDEO REFERENCE

- https://nptel.ac.in/courses/202/105/106205166/

| Sr. No. | LIST OF EXPERIMENT | Date | Signature | Remarks |
|---|---|---|---|---|
| 1. | Study and Understanding of basic graphics function. | | | |
| 2. | Write a program to implement line, circle and ellipse attributes. | | | |
| 3. | Write program to draw any figure using circle and lines | | | |
| 4. | Write a program to draw a hut following figure using circle and line and rectangle. | | | |
| 5. | Write a program to plot a point (pixel) on the screen. | | | |
| 6. | Write a program to draw straight line using direct method. | | | |
| 7. | Write a program to draw straight line using simple DDA method. | | | |
| 8. | Write a program to draw straight line using incremental DDA algorithm. | | | |
| 9. | Write a program to draw straight line using Bresenham's line drawing algorithm. | | | |
| 10. | Write a program to draw straight line using midpoint line drawing algorithm. | | | |
| 11. | Write a program to draw a circle using midpoint circle drawing algorithm. | | | |
| 12. | Write a program to draw an ellipse using midpoint ellipse drawing algorithm. | | | |

## EXPERIMENT NO. 1

**AIM:** Study and Understanding of basic graphics function.

**PROGRAM:**

### 1) Initgraph ()

initgraph() function initializes the graphics mode and clears the screen.

Declaration:

void far initgraph(int far *driver, int far *mode, char far *path)

### 2) Detectgraph ()

Detectgraph function determines the graphics hardware in the system, if the function finds a graphics adapter then it returns the highest graphics mode that the adapter supports.

Declaration:

void far detectgraph(int far *driver, int far *mode)

Integer that specifies the graphics driver to be used. You can give graphdriver a value using a constant of the graphics_drivers enumeration type.

### 3) Closegraph ()

closegraph() function switches back the screen from graphcs mode to text mode. It clears the screen also. A graphics program should have a closegraph function at the end of graphics. Otherwise DOS screen will not go to text mode after running the program.

### 4) Getpixel ()

getpixel function returns the color of pixel present at location(x, y).

Declaration:-

intgetpixel(int x, int y);

### 5) Putpixel ()

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

Declaration:-

void putpixel(int x, int y, intcolor);

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.

## 6) line()

line function 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.

Declaration :-

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

## 7) lineto()

lineto function draws a line from current position(CP) to the point(x,y), you can get current position using getx and gety function.

## 8) circle()

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

Declaration :-

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

## 9) ellipse()

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

Declaration :-

void ellipse(int x, int y, intstangle, intendangle, intxradius, intyradius);

## 10) drawpoly()

draw poly function is used to draw polygons i.e. triangle, rectangle, pentagon, hexagon etc. Declaration :-

void drawpoly( intnum, int *polypoints );

num indicates (n+1) number of points where n is the number of vertices in a polygon, polypoints points to a sequence of (n*2) integers . Each pair of integers gives x and y coordinates of a point on the polygon. We specify (n+1) points as first point coordinates should be equal to (n+1)th to draw a complete figure.

To understand more clearly we will draw a triangle using drawpoly, consider for example the array :-

int points[] = { 320, 150, 420, 300, 250, 300, 320, 150};
points array contains coordinates of triangle which are (320, 150), (420,300) and (250, 300). Note that last point(320, 150) in array is same as first.

## 11) outtext ()

outtext function displays text at current position.

Declaration :-



void outtext(char *string);

## 12) outtextxy ()

out textxy function display text or string at a specified point(x,y) on the screen. Declaration :-

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

x, y are coordinates of the point and third argument contains the address of string to be displayed.

## 13) rectangle()

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.

Declaration :-

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

## 14) floodfill()

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.

Declaration :-

voidfloodfill(int x, int y, int border);

## 15)fillpoly()

fillpoly function draws and fills a polygon. It require same arguments as drawpoly. Declaration :-

void drawpoly( int num, int *polypoints );

## 16) fillellipse()

fillellipse function draws and fills a polygon.

Declaration:-

void fillellipse(int x, int y, int xradius, int yradius);

x and y are coordinates of center of the ellipse, xradius and yradius are x and y radius of ellipse respectively.

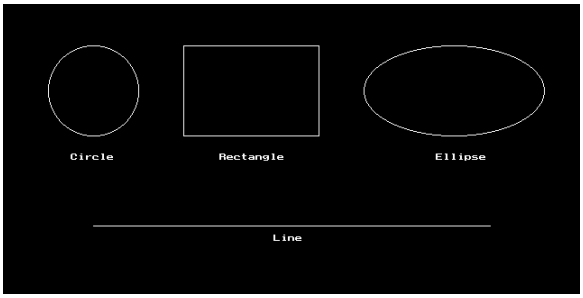
## EXPERIMENT NO. 2

**AIM:** Write a program to implement line, circle and ellipse attributes.

## PROGRAM:

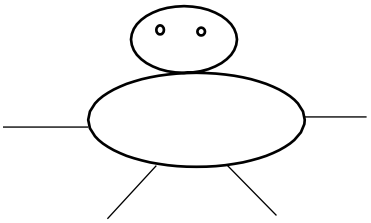
```
#include <graphics.h>
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
void main()
{
        int gd=DETECT, gm;
        initgraph(&gd, &gm, " C:\\TC\\bgi");
        circle(100,100,50);
        outtextxy(75,170, "Circle");
        rectangle(200,50,350,150);
        outtextxy(240, 170, "Rectangle");
        ellipse(500, 100,0,360, 100,50);
        outtextxy(480, 170, Ellipse");
        line(100,250,540,250);
        outtextxy(300,260,"Line");
        getch();
        closegraph();
}
```

**Output:**



## EXPERIMENT NO. 3

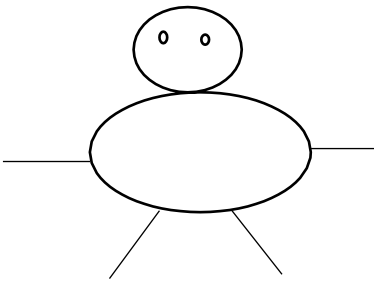
**AIM:** Write a program to draw following figure using circle and lines.



## PROGRAM:

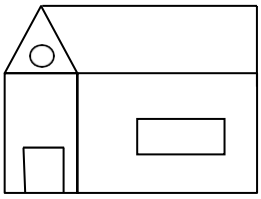
```
#include<graphics.h>
#include<conio.h>
#include<stdio.h> void
main()
{
        int gd=DETECT,gm;
        initgraph(&gd,&gm,"C:\\TC\\bgi ");
        circle (200,100,50);
        circle(180, 80, 5);
        circle( 200,80,5);
        ellipse( 200,220,0,360,100,70);
        line(30,220,100,220);
        line(300,220,370,220);
        line(170, 290,140,380);
        line(220,290,250,380);
        getch();
        closegraph();
}
```

**Output:**



## EXPERIMENT NO. 4

**AIM:** Write a program to draw a hut following figure using circle and line and rectangle.



## PROGRAM:

```
#include<graphics.h>
#include<conio.h>
#include<stdio.h>
void main()
{
int gd=DETECT,gm;
initgraph(&gd,&gm,"C:\\TC\\bgi ");
rectangle(100,100,200,200);
line(100,100,150,50);
line(150,50,200,100);
line(150,50,400,50);
line(200,100,430,100);
line(400,50,430,100);
line(400,100,400,200);
line(200,200,400,200);
rectangle(135,140,165,200);
circle(150,75,15);
rectangle(250,140,330,170);
getch();
closegraph();
}
```

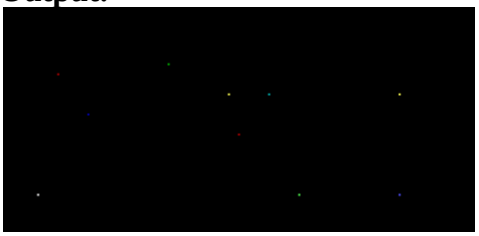
## EXPERIMENT NO. 5

**AIM:** Write a program to plot a point (pixel) on the screen.

## PROGRAM:

```
#include <graphics.h>
#include <stdio.h> void
main()
{
    int gd = DETECT, gm, color;
    initgraph(&gd, &gm, " C:\\TC\\bgi");
    putpixel(85, 35, GREEN);
     putpixel(30, 40, RED);
     putpixel(115, 50, YELLOW);
     putpixel(135, 50, CYAN);
     putpixel(45, 60, BLUE);
     putpixel(20, 100, WHITE);
     putpixel(200, 100, LIGHTBLUE);
     putpixel(150, 100, LIGHTGREEN);
     putpixel(200, 50, YELLOW);
     putpixel(120, 70, RED);
     color= getpixel(120,70);
     printf("the value of color of pixel is %d", color);
     getch();
     closegraph();
    }
```

**Output:**

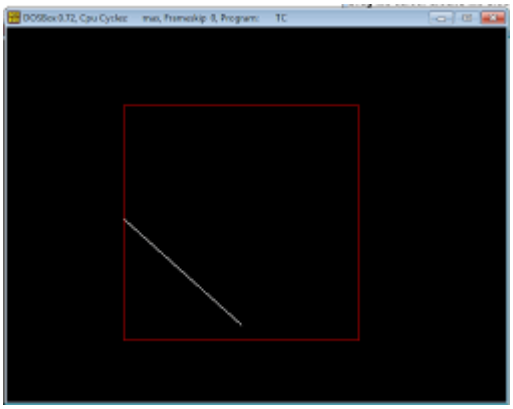


## EXPERIMENT NO. 6

**AIM:** Write a program to draw straight line using direct method.

```
include<graphics.h>
#include<stdio.h>
#include<conio.h>
void main()
{
int gd = DETECT, gm; int x,x1,y,y1,dx,dy,m,c;
       initgraph(&gd, &gm, "C:\\TC\\bgi");
       printf("\n\n Enter X:");
       scanf("%d",&x);
       printf("\n\n Enter Y:");
       scanf("%d",&y);
       printf("\n\n Enter X1:");
       scanf("%d",&x1);
       printf("\n\n Enter Y1:");
       scanf("%d",&y1);
       dy=y1-y;
       dx=x1-x;
       m=dy/dx;
       c=y-(m*x);
       if(m<=1)
 {for(;x<=x1;x++)
   {y=(m*x)+c; putpixel(x,y,1);     } }
          else
          {for(;y<=y1;y++)
                   {x=(y-c)/m;
                    putpixel(x,y,1); } }
  getch();
closegraph();
      }
```

**Output:**



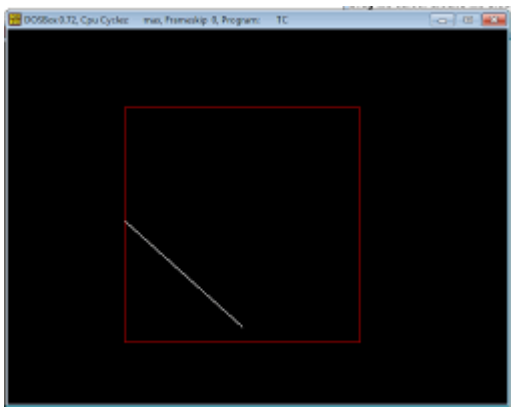
## EXPERIMENT NO. 7

**AIM:** Write a program to draw straight line using simple DDA method.

**PROGRAM:**

```
include<graphics.h>
#include<stdio.h>
#include<conio.h>
void main()
{
      intgd = DETECT, gm; int
      x,x1,y,y1,dx,dy,m,c;
      initgraph(&gd, &gm, "C:\\TC\\bgi");
      printf("\n\n Enter X:");
      scanf("%d",&x);
      printf("\n\n Enter Y:");
      scanf("%d",&y);
      printf("\n\n Enter X1:");
      scanf("%d",&x1);
      printf("\n\n Enter Y1:");
      scanf("%d",&y1);
dy=y1-y; dx=x1-x; m=dy/dx;
      if(m<=1)
         {
 for(;x<=x1;x++)
   {
    y=y+m; putpixel(x,y,1);
         }
         }
         else
         {
          for(;y<=y1;y++)
                {  x=x+(1/m); putpixel(x,y,1); }
          }
  getch();
closegraph(); }
```

**Output:**



## EXPERIMENT NO. 8

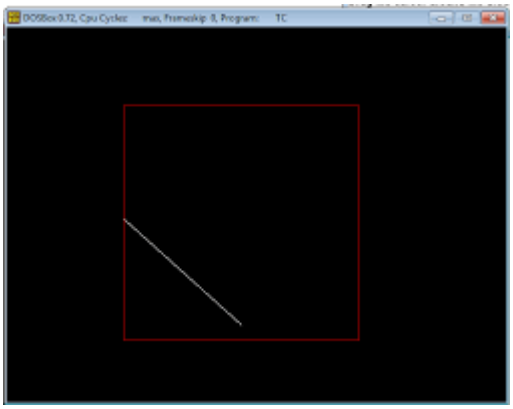
**AIM:** Write a program to draw straight line using incremental DDA algorithm.

**PROGRAM:**

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<math.h>
void main()
 {
 int x1,y1,x2,y2,dx,dy,length,i; float
 x,y,xinc,yinc;
 int gd=DETECT,gm;
 //clrscr();
 initgraph(&gd,&gm," C:\\TC\\bgi ");
 printf("Enter the starting coordinates");
 scanf("%d%d",&x1,&y1); printf("Enter the
 ending coordinates");
 scanf("%d%d",&x2,&y2);
 dx=x2-x1;
 dy=y2-y1;
 if(abs(dx)>abs(dy))
length=abs(dx); else
length=abs(dy);
xinc=dx/(float)length;
yinc=dy/(float)length; x=x1
y=y1 putpixel(x,y,10);
```

```
 for(i=0;i<length;i++)
{
putpixel(x,y,10);
 x=x+xinc; y=y+yinc;
 //delay(10);
 }
  getch();
  closegraph();
 }
```

**Outtput:**



## EXPERIMENT NO. 9

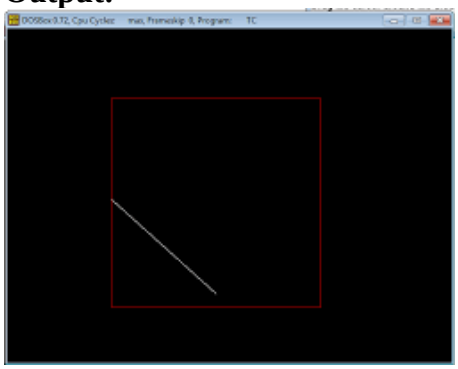
**AIM:** Write a program to draw straight line using Bresenham's line drawing algorithm.

**PROGRAM:**

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

#include<graphics.h>
#include<math.h>
void main()
  {
  int x1,y1,x2,y2,dx,dy,length,i; float
  x,y,xinc,yinc;
  int gd=DETECT,gm;
  //clrscr();
  initgraph(&gd,&gm," C:\\TC\\bgi ");
  printf("Enter the starting coordinates");
  scanf("%d%d",&x1,&y1); printf("Enter the
  ending coordinates");
  scanf("%d%d",&x2,&y2);
  dx=x2-x1;
  dy=y2-y1;
  if(abs(dx)>abs(dy))
length=abs(dx); else
length=abs(dy);
xinc=dx/(float)length;
yinc=dy/(float)length; x=x1
y=y1 putpixel(x,y,10);
for(i=0;i<length;i++)
{ putpixel(x,y,10);
x=x+xinc; y=y+yinc;
//delay(10); }
  getch();
  closegraph();
}
```

**Output:**



## EXPERIMENT NO. 10

**AIM:** Write a program to draw straight line using midpoint line drawing algorithm.

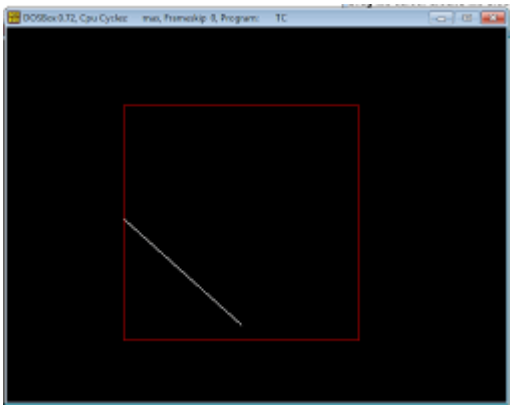
## PROGRAM:

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

 void main()
{
int x0,y0,x1,y1,x,y,dx,dy,d,de,dne; int
gdriver=DETECT,gmode,gerror;
initgraph(&gdriver,&gmode,"..\\BGI");
     printf("\n Enter the x and y value for initial point:\n");
      scanf("%d%d",&x0,&y);
printf("\n Enter the x and y value for final point:\n");
scanf("%d%d",&x1,&y1);
printf("\n The Line is shown below: \n");
     x1-x0;
      m=(dy/dx);
      de=dy;
      dne=dy-dx;
putpixel(x,y,RED);
if(m<1)
{
d= dy-(dx/2);
while(x<x1)
{
        x=x+1;
```

```
        if(d<=0)
            d=d+de;
        else
        {
            y=y+1;
            d=d+dne;
        }
putpixel( x,y,RED);
}
}
else if(m>1)
{
d= dx-(dy/2); while(y<y1)
{
        y=y+1;
        if(d<=0)
            d=d+dx;
        else
        {
            x=x+1;
            d=d+dx-dy;
        }
putpixel( x,y,RED);
}
}
    getch();
    }
```

**Output:**



## EXPERIMENT NO. 11

**AIM:** Write a program to draw a circle using midpoint circle drawing algorithm.

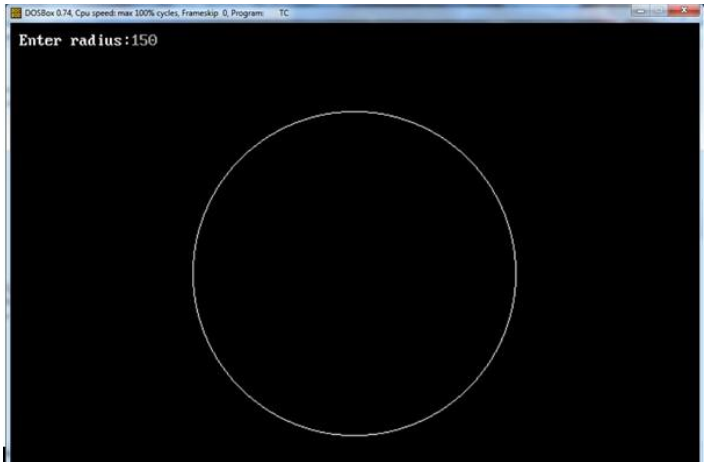
## PROGRAM:

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
void pixel(int xc,int yc,int x,int y);
void main()
{
        int gd=DETECT,gm,xc,yc,r,x,y,d;
        clrscr(); initgraph(&gd,&gm,"C:\\TC\\bgi
        "); printf("Enter the value of Xc\t");
        scanf("%d",&xc);
        printf("Enter the value of Yc \t");
        scanf("%d",&yc);
        printf("Enter the Radius of circle\t");
        scanf("%d",&r);
        x=0;
        y=r;
        d=1-r;
        pixel(xc,yc,x,y);
        while(x<y)
        {
                if(d<0)
                {
                  x=x+1;
                d=d+(2*x)+3; }
                else
                {
              x=x+1; y=y-1;
```

```
                d=d+(2*x)-(2*y)+5;
                }
                pixel(xc,yc,x,y);
        }
        getch();
        closegraph();

                }
void pixel(int xc,int yc,int x,int y)
{
        putpixel(xc+x,yc+y,7);
        putpixel(xc+y,yc+x,7);
        putpixel(xc-y,yc+x,7);
        putpixel(xc-x,yc+y,7);
        putpixel(xc-x,yc-y,7);
        putpixel(xc-y,yc-x,7);
        putpixel(xc+y,yc-x,7);
        putpixel(xc+x,yc-y,7);
}
```

**Output:**



## EXPERIMENT NO. 12

**AIM:** Write a program to draw an ellipse using mid-point ellipse drawing algorithm.

## PROGRAM:

```
#include<conio.h>
#include<stdio.h>
#include<graphics.h>
void main()
{
 int gd=DETECT,gm;
 float x,y,xc,yc,rx,ry,pk,pk1;
 clrscr();
 initgraph(&gd,&gm,"..\\bgi");
 printf("Mid point ellipse drawing algorithm\n");
 printf("Enter Center for ellipse\n x : ");
 scanf("%f",&xc);
 printf("y : ");
 scanf("%f",&yc);
 printf("Enter x-radius and y-radius\n x-radius : ");
scanf("%f",&rx);
 printf("y-radius : ");
 scanf("%f",&ry);
 x=0;
 y=ry;
 pk=(ry*ry)-(rx*rx*ry)+((rx*rx)/4);
 while((2*x*ry*ry)<(2*y*rx*rx))
 {
  if(pk<=0)
  {
   x=x+1;
   pk1=pk+(2*ry*ry*x)+(ry*ry);
  }
  else
  {
  x=x+1;
  y=y-1;
  pk1=pk+(2*ry*ry*x)-(2*rx*rx*y)+(ry*ry);
  }
  pk=pk1;
  putpixel(xc+x,yc+y,2);
  putpixel(xc-x,yc+y,2);
```

```
 putpixel(xc+x,yc-y,2);
  putpixel(xc-x,yc-y,2);

  }
  pk=((x+0.5)*(x+0.5)*ry*ry)+((y-1)*(y-1)*rx*rx)-(rx*rx*ry*ry);
  while(y>0)
  {
  if(pk>0)
  {
  y=y-1;
  pk1=pk-(2*rx*rx*y)+(rx*rx);
  }
  else
  {
  x=x+1;
  y=y-1;
  pk1=pk+(2*ry*ry*x)-(2*rx*rx*y)+(rx*rx);
  }
  pk=pk1;
  putpixel(xc+x,yc+y,2);
  putpixel(xc-x,yc+y,2);
  putpixel(xc+x,yc-y,2);
 putpixel(xc-x,yc-y,2);

  }
  line(xc+rx,yc,xc-rx,yc);
  line(xc,yc+ry,xc,yc-ry);
  outtextxy(xc+(1.2*rx),yc-(1.2*ry),"(x,y)");
  outtextxy(xc-(1.2*rx),yc+(1.2*ry),"(-x,-y)");
  outtextxy(xc+(1.2*rx),yc+(1.2*ry),"(x,-y)");
  outtextxy(xc-(1.2*rx),yc-(1.2*ry),"(-x,y)");
  getch();
 }
```

**Output:**