RPS GROUP OF INSTITUTIONS
Balana, Satnali Road,
Mohindergarh, Haryana 123029
Ph.: 91-1285-241431
Mob.: 09466275566, 09416150201
E-mail: info@rpsinstitutions.org
Web : www.rpsinstitutions.org

Approved by **AICTE** (Govt. of India) & Affiliated to **M.D. University**, Rohtak

RPS GROUP OF INSTITUTIONS, BALANA

MOHINDERGARH

LAB MANNAUL

COMPUTER GRAPICS

CSE 5TH SEM

Department of Computer Science and Engineering

2015-16

LAB MANUAL
Computer Graphics Lab

1. Syllabus

- To study basic graphic functions.
- Write a program to draw a line using DDA Algorithm
- Write a program to draw a circle using Bresenham's Algorithm.
- Write a program to draw a line using Bresenham's Algorithm
- Write a proram to draw a circle using mid point circle algorithm.
- Write a program to scale a triangle
- Write a program to translate a triangle
- Write a program to rotate a triangle.
- Write a program to display a hut.
- Write a program to rotate a point around another point.
- Write a program to rotate a circle on another circle.
- Write a program to reflect a triangle.

2. OBJECTIVE OF THE COMPUTER GRAPHICS LAB

- ❖ Computer graphics provide facilities to create 2D and 3D graphs of mathematical, Physical and Economical functions; histograms , bar and pie charts; task scheduling charts; inventory and production charts and the like.

3. Hardware and Software requirement

a) **Software requirement :** Turbo C / C++

b) **Hardware requirement**

- Intel Pentium III800 MHz Processor
- Intel chipset 810 Motherboard
- 14” color Monitor
- Mouse
- Keyboard
- 2GB HDD
- 256 MB RAM

EXPERIMENT NO : 1

- ***AIM:- To study basic graphic functions.***

1) INITGRAPH

• Initializes the graphics system.

Decleration

- Void far initgraph(int far *graphdriver)

Remarks

- To start the graphic system, you must first call initgraph.
- Initgraph initializes the graphic system by loading a graphics driver from disk (or validating a registered driver) then putting the system into graphics mode.
- Initgraph also resets all graphics settings (color, palette, current position, viewport, etc) to their defaults then resets graph.

2) GETPIXEL, PUTPIXEL

- Getpixel gets the color of a specified pixel.
- Putpixel places a pixel at a specified point.

Decleration

- Unsigned far getpixel(int x, int y)
- Void far putpixel(int x, int y, int color)

Remarks

- Getpixel gets the color of the pixel located at (x,y);
- Putpixel plots a point in the color defined at (x, y).

Return value

- Getpixel returns the color of the given pixel.
- Putpixel does not return.

3) CLOSE GRAPH

- Shuts down the graphic system.

Decleration

- Void far closegraph(void);

Remarks

- Close graph deallocates all memory allocated by the graphic system.
- It then restores the screen to the mode it was in before you called initgraph.

Return value

- None.

3. ARC, CIRCLE, PIESLICE

a. arc draws a circular arc.
b. Circle draws a circle.
c. Pieslice draws and fills a circular pieslice

Decleration

- Void far arc(int x, int y, int stangle, int endangle, int radius);
- Void far circle(int x, int y, int radius);
- Void far pieslice(int x, int y, int stangle, int endangle, int radius);

Remarks

- Arc draws a circular arc in the current drawing color .
- Circle draws a circle in the current drawing color
- Pieslice draws a pieslice in the current drawing color, then fills it using the current fill pattern and fill color.

4) ELLIPSE, FILLELIPSE, SECTOR

- Ellipse draws an elliptical arc.
- Fillellipse draws and fills ellipse.
- Sector draws and fills an elliptical pie slice.

Decleration

- Void far ellipse(int x, int y, int stangle, int endangle, int xradius, int yradius)
- Void far fillellipse(int x, int y, int xradius, int yradius)
- Void farsectoe(int x, int y, int stangle, int endangle, int xradius, int yradius)

Remarks

- Ellipse draws an elliptical arc in the current drawing color.
- Fillellipse draws an elliptical arc in the current drawing color and than fills it with fill color and fill pattern.
- Sector draws an elliptical pie slice in the current drawing color and than fills it using the pattern and color defined by setfillstyle or setfillpattern.

6) FLOODFILL

- Flood-fills a bounded region.

Decleration

Void far floodfill(int x, int y, int border)

Remarks

- Floodfills an enclosed area on bitmap device.
- The area bounded by the color border is flooded with the current fill pattern and fill color(x,y) is a "seed point"

  If the seed is within an enclosed area, the inside will be filled.

- If the seed is outside the enclosed area, the exterior will be filled. Use fillpoly instead of floodfill wherever possible so you can maintain code compatibility with future versions. Floodfill doesnot work with the IBM-8514 driver.

Return value

- If an error occurs while flooding a region, graph result returns '1'.

7) GETCOLOR, SETCOLOR

- Getcolor returns the current drawing color.
- Setcolor returns the current drawing color.

Decleration

- Int far getcolor(void);
- Void far setcolor(int color)

Remarks

- Getcolor returns the current drawing color.
- Setcolor sets the current drawing color to color, which can range from 0 to getmaxcolor.
- To set a drawing color with setcolor , you can pass either the color number or the equivalent color name.

8) LINE,LINEREL,LINETO

- Line draws a line between two specified pints.
- Onerel draws a line relative distance from current position(CP).
- Linrto draws a line from the current position (CP) to(x,y).

Remarks

- Line draws a line from (x1, y1) to (x2, y2) using the current color, line style and thickness. It does not update the current position (CP).
- Linerel draws a line from the CP to a point that is relative distance (dx, dy) from the CP, then advances the CP by (dx, dy).
- Lineto draws a line from the CP to (x, y), then moves the CP to (x,y).

Return value

- None

9) RECTANGLE

- Draws a rectangle in graphics mode.

Decleration

- Void far rectangle(int left, int top, int right, int bottom)

Remarks

- It draws a rectangle in the current line style, thickness and drawing color.
- (left, top) is the upper left corner of the rectangle, and (right, bottom) is its lower right corner.

Return value

- None.

Experiment No.-2

Aim: - Write a program to draw a line using DDA Algorithm

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
void main()
{
int x,y,x1,x2,y1,y2,k,dx,dy,s,xi,yi; int
gdriver=DETECT,gmode;
initgraph(&gdriver,&gmode,"C:\\tc\\bgi:");
printf("enter first point"); scanf("%d%d",&x1,&y1);
printf("enter second point");
scanf("%d%d",&x2,&y2); x=x1;
y=y1;
putpixel(x,y,7); dx=x2-x1;
dy=y2-y1;
if(abs(dx)>abs(dy))
s=abs(dx);
else
s=abs(dy);
xi=dx/s;
yi=dy/s;
x=x1;
y=y1;
putpixel(x,y,7);
for(k=0;k<s;k++)
{
        x=x+xi;
        y=y+yi;
```

```
putpixel(x,y,7);
        }
        getch();
        closegraph();
}
```

Experiment No.-3

Aim:-Write a program to draw a circle using Bresenham's Algorithm.

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
void
circlepoints(int,int);
void main()
{
        int x,y,p,r;
        int             gdriver=DETECT,gmode;
        initgraph(&gdriver,&gmode,"C:\\tc\\bgi:
        "); clrscr();
        printf("enter          the
        radius");
        scanf("%d",&r);
        x=0;y=r;p=1-r;
        while(x<y)
        {
                x++;
                if(p>0)
                {
```

```
                    p=p+2*(x-
                    y)+1; y--;
                }
                else
                    p=p+2*x+1;
                circlepoints(x,y);
            }
            getch();
            closegraph();
}
void circlepoints(int x,int y)
```

ExperimentNo-4

Aim: - Write a program to draw a circle using mid point circle algorithm.

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<math.h>    void
ellips(int x,int y);
void completellipse(int r,int g,int u,int v)
{
        float  s,k,e,f,x;  double
        p1,p2;          s=r;k=g;
        e=(pow((s+.5),2));
        f=(pow((k-1),2));
        p2=((u*e)+(v*f)-(u*v));
        ellips(s,k); while(k>=0)
         {
                if(p2>0) p2=(p2+v-(2*v*s));
                else
```

```
                {
                        p2=(p2+(2*u*(s+1))-(2*v*(k-1))+v);
                        s++;
                }
                k--;
                ellips(s,k);
        }
}
void main()
{
        int gdriver=DETECT,gmode; int a,b,x,y;
        long u,v,p1; initgraph(&gdriver,&gmode,"C:\\tc\\bgi::"); printf("\n enter the length of
        major axis:"); scanf("\t%d",&a);
        printf("\n enter the length of minor axis:"); scanf("\t%d",&b);
        x=0;
        y=b;
        u=pow(b,2);
        v=pow(a,2); p1=(u-(v*b)+(.25*v)); ellips(x,y); while(2*(u*x)<=2*(v*y))
        {
                x++;
                if(p1<0)
                        p1=(p1+(2*u*v)+v);
                else
                {
                        p1=(p1+(2*u*x)-(2*v*y)+u); y--;
                }
                ellips(x,y);
        }
        completellipse(x,y,u,v);
        getch();
        closegraph();
```

```
}
void ellips(int x,int y)
{
        putpixel(x+200,y+200,8); putpixel(-x+200,y+200,8); putpixel(x+200,-y+200,8); putpixel(-
        x+200,-y+200,8);
}
{
        putpixel(x+300,y+300,8); putpixel(x+300,-y+300,8); putpixel(-x+300,y+300,8); putpixel(-
        x+300,-y+300,8);      putpixel(y+300,x+300,8);      putpixel(y+300,-x+300,8);      putpixel(-
        y+300,x+300,8); putpixel(-y+300,-x+300,8);
}
```

Experiment No:-5

Aim:- PROGRAM T O SCALE THE TRIANGLE

```
#include<iostream.h>
#include<conio.h>
#include<graphics.h>O void main()
{
int  gd=DETECT,gm;  initgraph(&gd,
&gm,""); cleardevice();
int x1,y1,x2,y2,x3,y3,x4,y4; float sx,sy;
cout<<"Enter  the  first  coordinates  of  triangle\n";
cin>>x1>>y1;
cout<<"Enter  the  second  coordinates  of  triangle\n";
cin>>x2>>y2;
cout<<"Enter  the  third  coordinates  of  triangle\n";
cin>>x3>>y3;
int                poly[8]={x1,y1,x2,y2,x3,y3,x1,y1};
cleardevice();
drawpoly(4,poly);
```

```
getch();
cout<<"Enter the scaling factors\n";
cin>>sx>>sy;
x4=sx*x1-x1; y4=sy*y1-
y1;
x1=sx*x1-x4; y1=sy*y1-
y4; x2=sx*x2-x4;
y2=sy*y2-y4; x3=sx*x3-x4; y3=sy*y3-y4; poly[0]=x1; poly[1]=y1; poly[2]=x2; poly[3]=y2;
poly[4]=x3; poly[5]=y3; poly[6]=x1; poly[7]=y1; getch(); cleardevice(); drawpoly(4,poly);
getch(); closegraph();
}
```

Experiment No-6

Aim:- PROGRAM TO TRANSLATE A TRIANGLE

```
#include<iostream.h>
#include<conio.h>
#include<graphics.h>
#include<process.h>
#include<math.h>

void main()
{
clrscr();
int graphdriver=DETECT,graphmode;
initgraph(&graphdriver,&graphmode,"...\\bgi");

int x,y,x1,y1,x2,y2,x3,y3;
cout<<"\n Enter Ist coordinates of triangle:"; cin>>x1>>y1;

cout<<"\n Enter 2nd coordinates of triangle:";
```

```
cin>>x2>>y2;

cout<<"\n          Enter 3rd coordinates of triangle:";
cin>>x3>>y3;

cleardevice();
line(x1,y1,x2,y2);
line(x2,y2,x3,y3);
line(x1,y1,x3,y3);
getch();
cleardevice();
cout<<"\n Enter translatio factors :\n"; cin>>x>>y;

x1-=x;
y1-=y;
x2-=x;
y2-=y;
x3-=x;
y3-=y;

cleardevice();
line(x1,y1,x2,y2);
line(x2,y2,x3,y3);
line(x1,y1,x3,y3);
getch();
closegraph();}
```

Experiment No:-7

Aim:- WAP TO ROTATE A TRIANGLE ABOUT ORIGIN.

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

```
#include<process.h>
#include<math.h>        void
main()
{
clrscr();
int                             graphdriver=DETECT,graphmode;
initgraph(&graphdriver,&graphmode,"...\\bgi");

int       x,y,x1,a[3][3];       double
b[3][3],c[3][3];
cout<<"\n     Enter     Ist     coordinates     of     triangle:";
cin>>a[0][0]>>a[1][0];

cout<<"\n          Enter 2nd coordinates of triangle:";
cin>>a[0][1]>>a[1][1];

cout<<"\n          Enter 3rd coordinates of triangle:";
cin>>a[0][2]>>a[1][2];

line(a[0][0],a[1][0],a[0][1],a[1][1]);
line(a[0][1],a[1][1],a[0][2],a[1][2]);
line(a[0][0],a[1][0],a[0][2],a[1][2]);
getch();
cleardevice();
cout<<"\n Enter angle of rotation:\n"; cin>>x;
b[0][0]=b[1][1]=cos((x*3.14)/180);  b[0][1]=-sin((x*3.14)/180);  b[1][0]=sin((x*3.14)/180);
b[2][2]=1; b[2][0]=b[2][1]=b[0][2]=b[1][2]= 0; for(int i=0;i<3;i++)
{
for(int j=0;j<3;j++) { c[i][j]=0;
for (int k=0; k<3;k++)
{
```

```
c[i][j]+=a[i][k]*b[k][j];
}
x1=(c[i][j]+0.5);
a[i][j]=x1;
}
  }
cout<<"\n Triangle after rotation is:\n" ;

line(a[0][0],a[1][0],a[0][1],a[1][1]);
line(a[0][1],a[1][1],a[0][2],a[1][2]);
line(a[0][0],a[1][0],a[0][2],a[1][2]);

getch();
closegraph();
}
```

Experiment No:-8

Aim:- PROGRAM TO DRAW A HUT USING SIMPLE GRAPHIC.

```
#include<conio.h>
#include<iostream.h>
#include<graphics.h>
#include<math.h>
#include<dos.h>
#include<process.h>
void main()
{
int             graphdriver=DETECT,graphmode;
initgraph(&graphdriver,&graphmode,"...\\bgi");
line(100,100,150,50);
line(150,50,200,100);
```

```
line(100,100,200,100);
line(150,50,350,50);
line(200,100,350,100);
line(350,50,350,100);
circle(150,75,10);
rectangle(100,100,200,300);
rectangle(200,100,350,300);
rectangle(250,175,300,225);
line(250,175,300,225);
line(300,175,250,225);
line(125,300,125,225);
line(175,300,175,225);
arc(150,225,0,180,25);
getch();
closegraph();

}
```

PROGRAM TO REFLECT A TRIANGLE

```
#include<iostream.h>
#include<conio.h>
#include<graphics.h>
#include<process.h>
#include<math.h>
void main()
{
clrscr();
int             graphdriver=DETECT,graphmode;
initgraph(&graphdriver,&graphmode,"...\\bgi");

int       x,y,x1,a[3][3];
```

```
double b[3][3],c[3][3];
cout<<"\n Enter Ist coordinates of triangle:";
cin>>a[0][0]>>a[1][0];

cout<<"\n Enter 2nd coordinates of triangle:";
cin>>a[0][1]>>a[1][1];

cout<<"\n Enter 3rd coordinates of triangle:";
cin>>a[0][2]>>a[1][2];

cout<<"\n Enter 1. for reflection in x-axis:\n";
cout<<"\n Enter 2. for reflection in y-axis:\n";
cout<<"\n Enter 3. for reflection in both the axis:\n";
cin>>x;
cleardevice();
line(320,0,320,479);
line(0,240,639,240);
```

```
line(a[0][0],a[1][0],a[0][1],a[1][1]);
line(a[0][1],a[1][1],a[0][2],a[1][2]);
line(a[0][0],a[1][0],a[0][2],a[1][2]);
switch(x)
{
case   1:b[0][0]=640-a[0][0];
       b[0][1]=640-a[0][1];
       b[0][2]=640-a[0][2];
       b[1][0]=a[1][0];
       b[1][1]=a[1][1];
       b[1][2]=a[1][2];
            line(320,0,320,479);
            line(0,240,639,240);
line(b[0][0],b[1][0],b[0][1],b[1][1]);
line(b[0][1],b[1][1],b[0][2],b[1][2]);
line(b[0][0],b[1][0],b[0][2],b[1][2]);
            getch();
            break;
case   2:b[1][0]=480-a[1][0];
       b[1][1]=480-a[1][1];
       b[1][2]=480-a[1][2];
       b[0][0]=a[0][0];
       b[0][1]=a[0][1];
       b[0][2]=a[0][2];
            line(320,0,320,479);
            line(0,240,639,240);
line(b[0][0],b[1][0],b[0][1],b[1][1]);
line(b[0][1],b[1][1],b[0][2],b[1][2]);
line(b[0][0],b[1][0],b[0][2],b[1][2]);
            getch();
            break;
```

```
case  3:  b[0][0]=640-a[0][0];
      b[0][1]=640-a[0][1];
      b[0][2]=640-a[0][2];
      b[1][0]=a[1][0];
      b[1][1]=a[1][1];
      b[1][2]=a[1][2];
          line(320,0,320,479);
          line(0,240,639,240);
line(b[0][0],b[1][0],b[0][1],b[1][1]);
line(b[0][1],b[1][1],b[0][2],b[1][2]);
          line(b[0][0],b[1][0],b[0][2],b[1][2]);
          b[1][0]=480-a[1][0];
      b[1][1]=480-a[1][1];
      b[1][2]=480-a[1][2];
      b[0][0]=a[0][0];
      b[0][1]=a[0][1];
      b[0][2]=a[0][2];
          line(320,0,320,479);
          line(0,240,639,240);
line(b[0][0],b[1][0],b[0][1],b[1][1]);
line(b[0][1],b[1][1],b[0][2],b[1][2]);
line(b[0][0],b[1][0],b[0][2],b[1][2]);
          getch();
          break;
}
getch();
closegraph();

}
```

Expe

riment No:- 9

Aim:- PROGRAM TO REFLECT A TRIANGLE

```
#include<iostream.h>
#include<conio.h>
#include<graphics.h>
#include<process.h>
clrscr();
int
graphdri
ver=DE
TECT,gr
aphmode
;
initgraph
(&graph
driver,&
graphmo
de,"...\\
]
cout<<"\n
Enter Ist
coordinates
of
triangle:";
cin>>a[0][0
]>>a[1][0];

cout<<"\n Enter 2nd coordinates of
```

```
triangle:";
cin>>a[0][1]>>a[1][1];

cout<<"\n        Enter 3rd coordinates of
triangle:";
cin>>a[0][2]>>a[1][2];

cout<<"\n
Enter 1. for
reflection in
x-axis:\n";
cout<<"\n
Enter 2. for
reflection in
y-axis:\n";
cout<<"\n
Enter 3. for
reflection in
both the
axis:\n";
cin>>x;
cleardevice();
line(320,0,320,479);
line(0,240,639,240);
line(a[0][0],a[1][0],a[0][1],a[1][1]);
line(a[0][1],a[1][1],a[0][2],a[1][2]);
line(a[0][0],a[1][0],a[0][2],a[1][2]);
switch(x)
{
case  1:b[0][0]=640-a[0][0];
      b[0][1]=640-a[0][1];
```

```
        b[0][2]=640-a[0][2];
        b[1][0]=a[1][0];
        b[1][1]=a[1][1];
        b[1][2]=a[1][2];
            line(320,0,320,479);
            line(0,240,639,240);
    line(b[0][0],b[1][0],b[0][1],b[1][1]);
    line(b[0][1],b[1][1],b[0][2],b[1][2]);
    line(b[0][0],b[1][0],b[0][2],b[1][2]);
            getch();
            break;
    case   2:b[1][0]=480-a[1][0];
        b[1][1]=480-a[1][1];
        b[1][2]=480-a[1][2];
        b[0][0]=a[0][0];
        b[0][1]=a[0][1];
        b[0][2]=a[0][2];
            line(320,0,320,479);
            line(0,240,639,240);
    line(b[0][0],b[1][0],b[0][1],b[1][1]);
    line(b[0][1],b[1][1],b[0][2],b[1][2]);
    line(b[0][0],b[1][0],b[0][2],b[1][2]);
            getch();
            break;
    case  3:  b[0][0]=640-a[0][0];
        b[0][1]=640-a[0][1];
        b[0][2]=640-a[0][2];
        b[1][0]=a[1][0];
        b[1][1]=a[1][1];
        b[1][2]=a[1][2];
            line(320,0,320,479);
            line(0,240,639,240);
```

```
line(b[0][0],b[1][0],b[0][1],b[1][1]);
line(b[0][1],b[1][1],b[0][2],b[1][2]);
        line(b[0][0],b[1][0],b[0][2],b[1][2]);
        b[1][0]=480-a[1][0];
    b[1][1]=480-a[1][1];
    b[1][2]=480-a[1][2];
    b[0][0]=a[0][0];
    b[0][1]=a[0][1];
    b[0][2]=a[0][2];
        line(320,0,320,479);
        line(0,240,639,240);
line(b[0][0],b[1][0],b[0][1],b[1][1]);
line(b[0][1],b[1][1],b[0][2],b[1][2]);
line(b[0][0],b[1][0],b[0][2],b[1][2]);
        getch();
        break;
}
getch();
closegraph();}
```

Experiment No:-10

Aim:- WAP to rotate a point around a origin.

```
#include<iostream.h>
#include<conio.h>
#include<graphics.h>
#include<math.h>
#include<dos.h> void main()
```

```
{


clrscr();

int gm,gd=DETECT;
initgraph(&gd,&gm,""); int
h,k,x1,y1,x2,y2,x3,y3; float t;

cout<<" OUTPUT"<<endl;

cout<<"Enter the coordinates of
point"<<endl; cin>>x2>>y2;

putpixel(x2,y2,2);

cout<<"Enter the angle for
rotation"<<endl; cin>>t;

cleardevice(); x1=int(x2*cos(t*3.14/180))-
(y2*sin(t*3.14/180));
y1=int(x2*sin(t*3.14/180))+(y2*cos(t*3.1
4/180)); cout<<"Point after rotation is:";
putpixel(x1,y1,2);

getch();

closegraph();

}
```