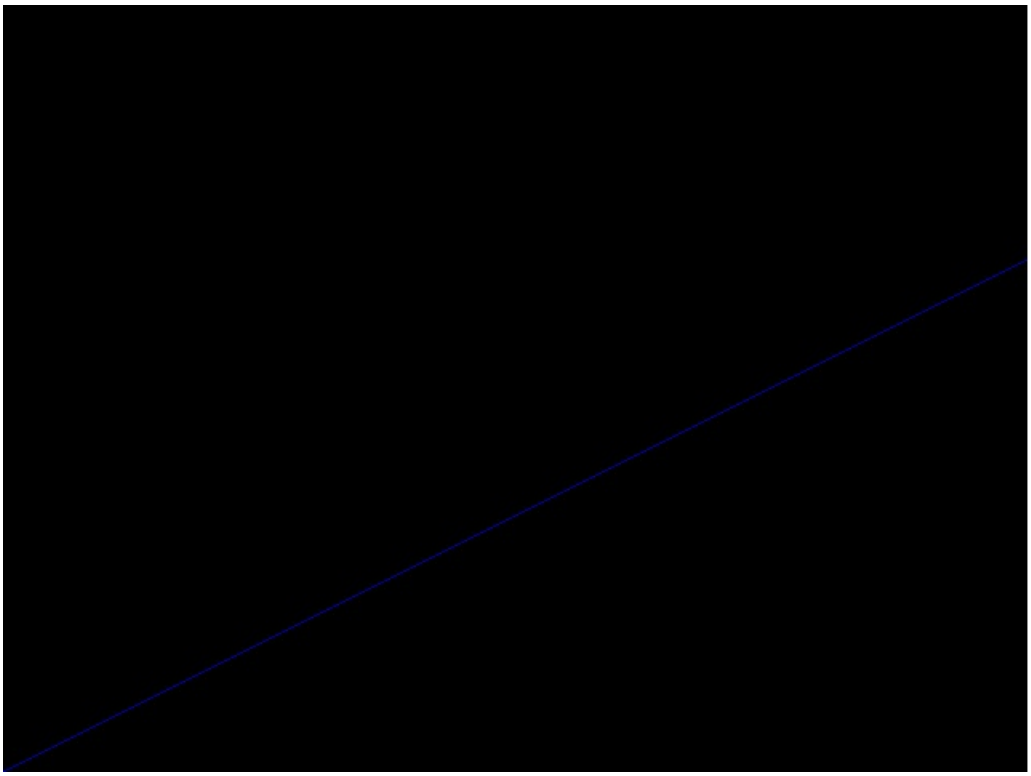
// GAURAV JAIN

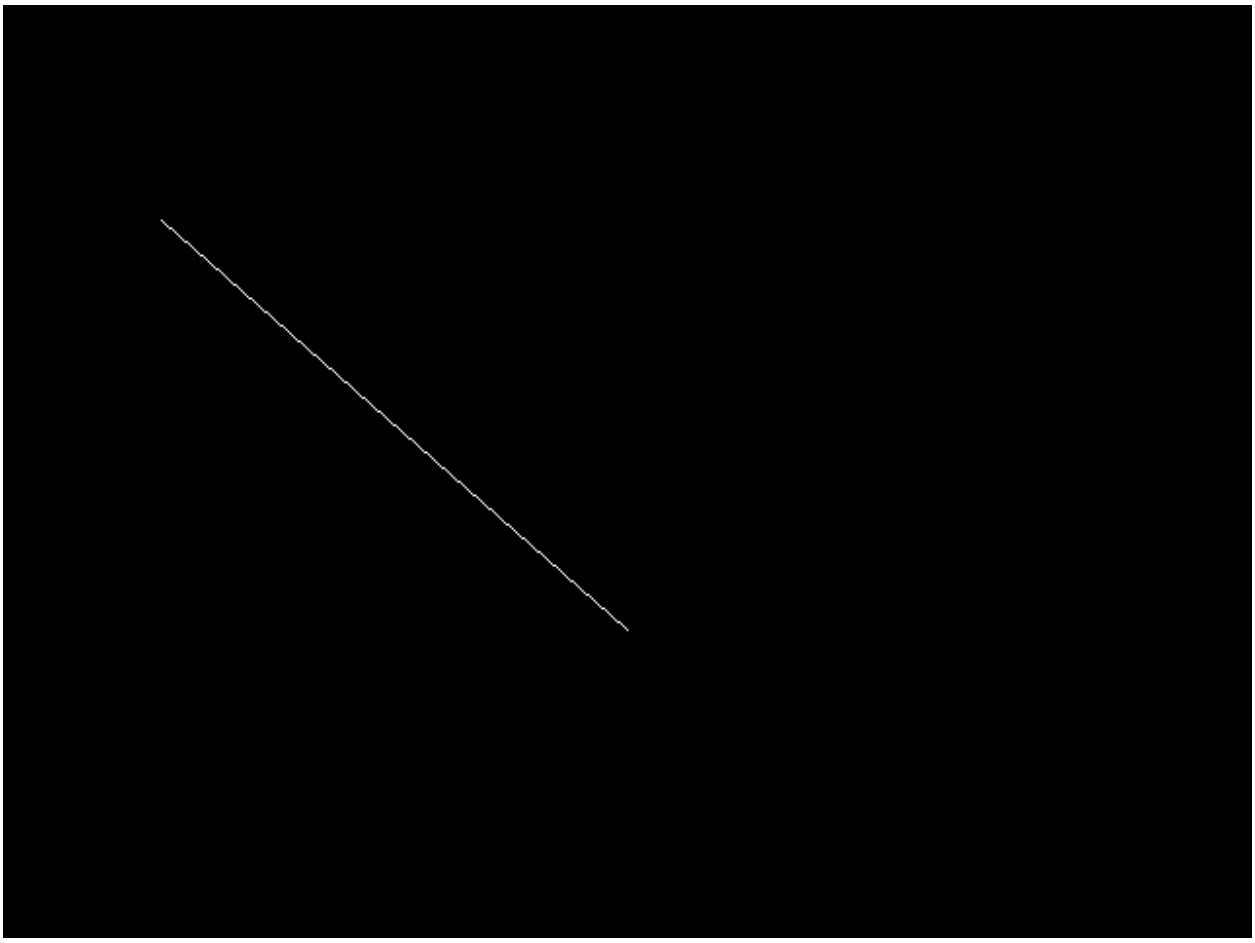
/* Program to print a straight line for a given equation y=x/2+100 */

```
#include<graphics.h>
#include<conio.h>
#include<stdio.h>
#include<dos.h>
main()
{
int x;
float y;
int graphdriver=DETECT,graphmode;
initgraph(&graphdriver,&graphmode,"C:\\bgi");
for(x=0;x<=100;x++)
{
y=.5*x+1;
putpixel(x,getmaxy()-int(y),BLUE);
}
getch();
return 0;
}
```



/*Midpoint Line Algorithm*/

```
#include<stdio.h>
#include<graphics.h>
#include<conio,h>
#include<dos.h>
void main(){
  int a,b,p,q,x,y,x1,x2,y1,y2,dx,dy;
  a=DETECT;
  initgraph(&a,&b,"C:\\TC\\BGI");
  printf("Enter x1 and y1\n");
  scanf("%d %d",&x1,&y1);
  printf("Enter x2 and y2\n");
  scanf("%d %d",&x2,&y2);
  dy=y2-y1;
  dx=x2-x1;
  p=dy-dx/2;
  q=getmaxy();
  y=y1;
  for(x-x1;x<=x2;x++){
          putpixel(x,q-y,RED);
          if(p>0){
                  y++;
                  p+=dy-dx;
                  }
          else
          p+=dy;
  }
  getch();
  closegraph();
}
```



/*Program to print Circles using Midpoint Algorithm and Polynomial Algorithm*/

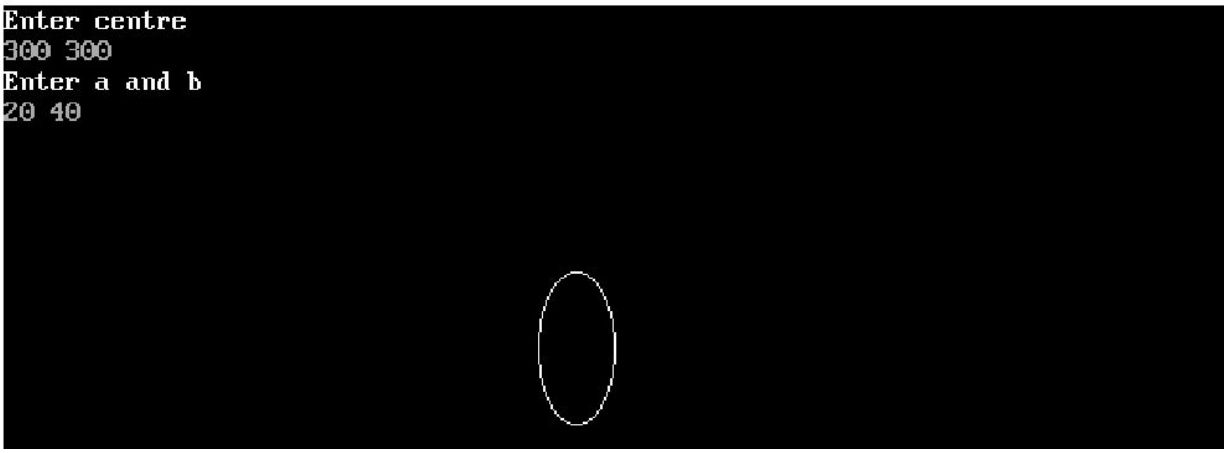
```
#include<graphics.h>
#include<conio.h>
#include<stdlib.h>
#include<stdio.h>
#include<dos.h>
#include<math.h>
void circ(void)
{
float a,b,r,x,y,i,j,d;
printf("enter the centre\n");
scanf("%f%f",&a,&b);
printf("enter the radius\n");
scanf("%f",&r);
x=0;
y=r;
d=(5/4-r);
putpixel(int(a),getmaxy()-int((b+r)),25);
putpixel(int(a-r),getmaxy()-int((b)),25);
putpixel(int(a+r),getmaxy()-int((b)),25);
putpixel(int(a),getmaxy()-int((b-r)),25);
while(y>x)
{
if(d<0)
{
d+=(2*x)+3;
}
else
{
d+=(2*(x-y))+5;
y--;
}
x++;
putpixel(int(a+x),getmaxy()-int((b+y)),25);
putpixel(int(a-x),getmaxy()-int((b+y)),25);
putpixel(int(a+x),getmaxy()-int((b-y)),25);
putpixel(int(a-x),getmaxy()-int((b-y)),25);
putpixel(int(a+y),getmaxy()-int((b+x)),25);
putpixel(int(a-y),getmaxy()-int((b+x)),25);
putpixel(int(a+y),getmaxy()-int((b-x)),25);
putpixel(int(a-y),getmaxy()-int((b-x)),25);
delay(50);
}
a+=150;
for(x=-r;x<=r;x++)
{
y=abs(int(sqrt(r*r-x*x)));
```

```
putpixel(a+x,getmaxy()-(b+y),15);
putpixel(a+x,getmaxy()-(b-y),15);
}
outtextxy(175,300,"Midpoint Algorithm");
outtextxy(350,300,"Bresanhams Algorithm");
}
main()
{
int graphdriver=DETECT,graphmode;
initgraph(&graphdriver,&graphmode,"..\\bgi");
circ();
getch();
return 0;
}
```

/* Drawing an ellipse using midpoint algorithm */

```
#include <conio.h>
#include <math.h>
#include <graphics.h>
#include<stdio.h>
using namespace std;
void plotline(int x,int y,int z,int o)
   {
      int a,b;
      a=getmaxx()/2;
      b=getmaxy()/2;
      line(a+x,b-y,a+z,b-o);
   }

void drawlips(int a,int b,int x,int y){
   putpixel(a+x,getmaxy()-(b+y),RED);
   putpixel(a+x,getmaxy()-(b-y),RED);
   putpixel(a-x,getmaxy()-(b-y),RED);
   putpixel(a-x,getmaxy()-(b+y),RED);
}

int main(){
initwindow(1234,480,"WINDOWS BGI");{

int a,b,x1,y1,c1,c2;
float d,d1;
printf("Enter radiuses of ellipse");
scanf("%d%d",&a,&b);
x1=0;
y1=b;
c1=getmaxx()/2;
c2=getmaxy()/2;

d=(b*b)+((a*a)*0.25)-(a*a*b);
drawlips(c1,c2,x1,y1);
while((b*b*(x1+1))<(a*a*(y1-0.5)))
{
   if(d<0)
   {
      d+=(((2*x1)+3)*(b*b));
      x1++;
   }
   else
   {
      d+=(((((2*x1)+3))*(b*b))+((2-(2*y1)))*(a*a));
      x1++,y1--;
   }
```

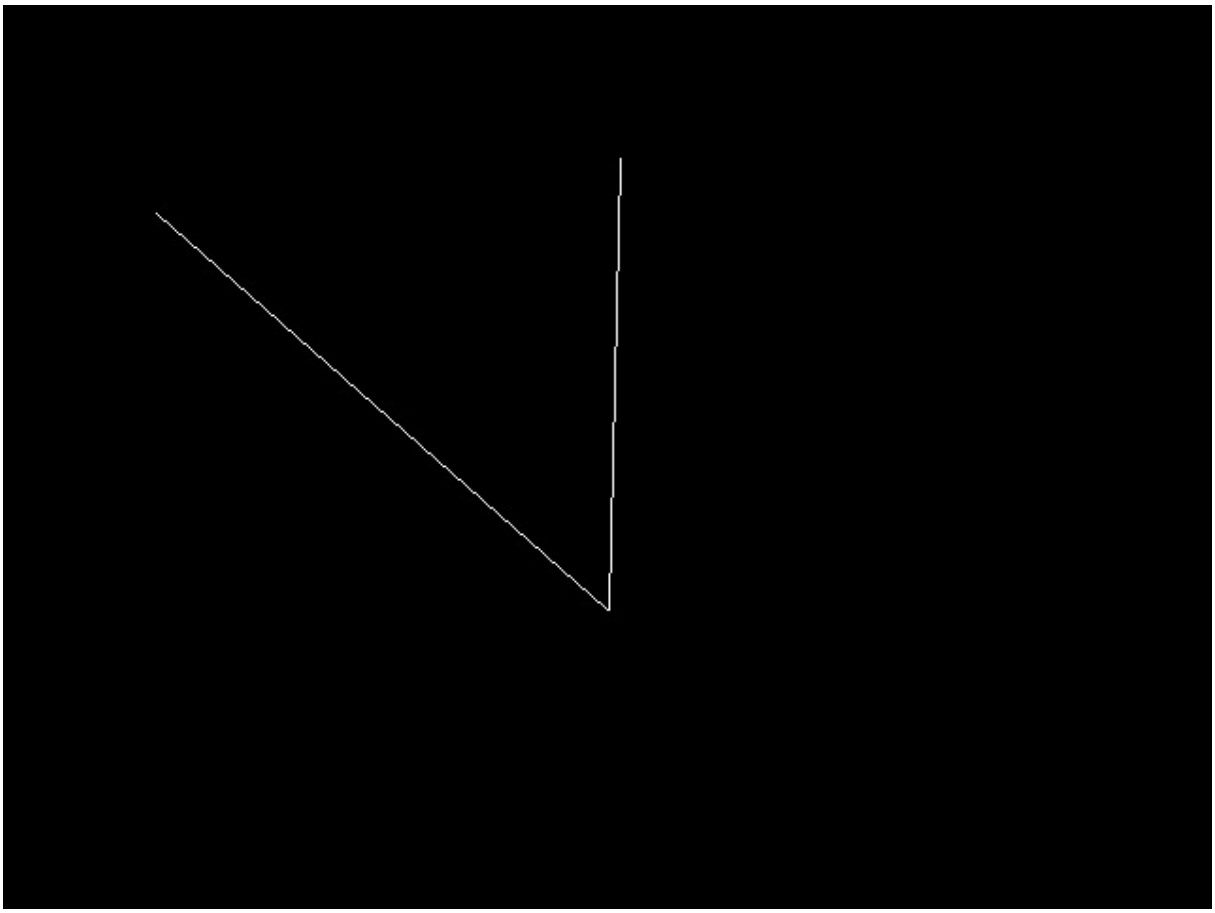
```
   drawlips(c1,c2,x1,y1);
}

d1=((b*b)*(x1+0.5)*(x1+0.5))+(a*a*(y1-1)*(y1-1))-(a*a*b*b);
while(y1>0)
{
   if(d1<0)
   {
      d1+=(((b*b)*((2*x1)+2))+((a*a)*((-2*y1)+3)));
      x1++;y1--;
   }
   else
   {
      d1+=((a*a)*(-2*y1+3));
      y1--;
   }
   drawlips(c1,c2,x1,y1);
}
getch();
closegraph();
}
return 0;
}
```



/* To take input from mouse and drawing a line using midpoint algorithm */

```
#include<graphics.h>
#include<stdio.h>
#include<conio.h>
#include<dos.h>
union REGS in,out;

int callmouse()
{
        in.x.ax=1;
        int86(51,&in,&out);
        return 1;
}
void mouseposi (int &xpos,int &ypos,int &click )
{
        in.x.ax=3;
        int86(51,&in,&out);
        click=out.x.bx;
        xpos=out.x.cx;
        ypos=out.x.dx;

}
int mousehide()
{
    in.x.ax=2;
    int86(51,&in,&out);
    return 1;
}
void setposi(int &xpos,int &ypos)
{
   in.x.ax=4;
   in.x.cx=xpos;
   in.x.dx=ypos;
   int86(51,&in,&out);
}
int main()
{
   int x,y,cl,a,b,cl1;
   clrscr();
   int g=DETECT,m;
   initgraph(&g,&m,"C:\\TC\\BGI");

   callmouse();
   do
   {
            mouseposi(x,y,cl);
            if(cl==1)
```

```
            {
                a=x;b=y;
                break;
            }
    }while(1);
    mousehide();
    delay(500);
    callmouse();
    do
    {
            mouseposi(x,y,cl);
            if(cl==1)
                break;
    }while(1);
    mousehide();

    line(a,b,x,y);
    getch();

    closegraph();
    return 0;
}
```

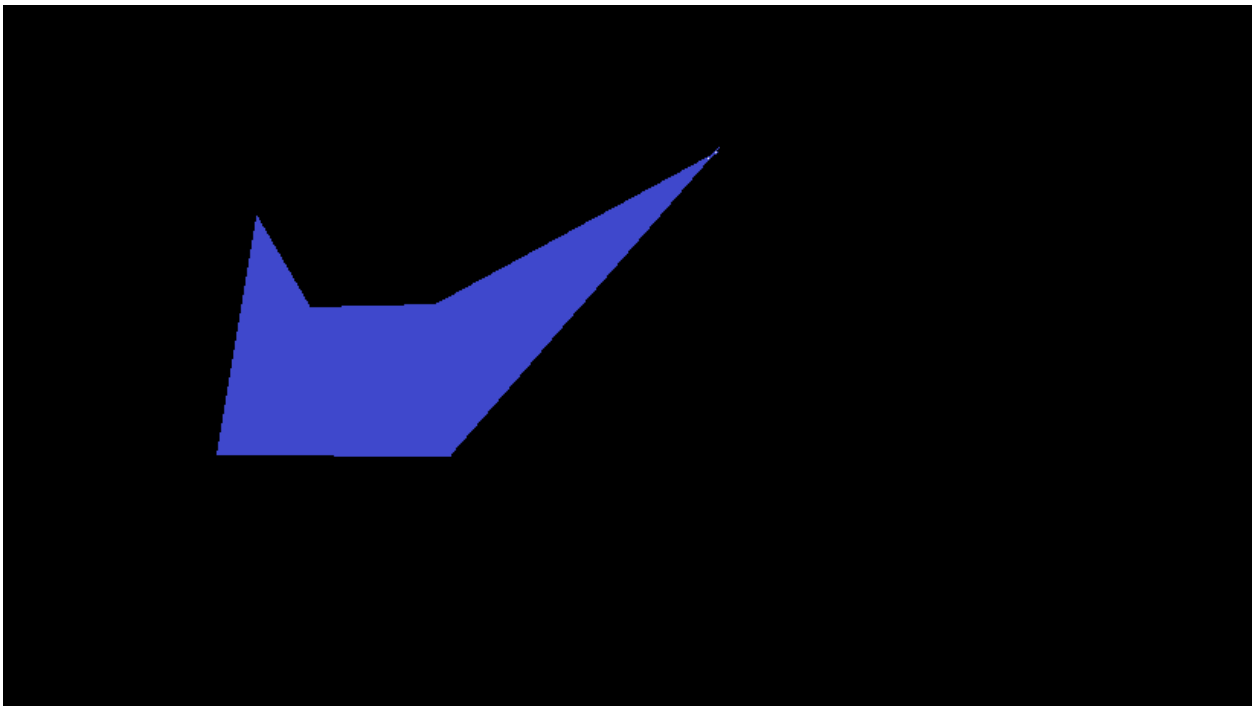


/* Scan Line Polygon Filling algorithm */

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

union REGS in,out;

int hidemouse()
{
  in.x.ax=2;
  int86(51,&in,&out);
  return 1;
}

int callmouse()
{
 in.x.ax=1;
 int86(51,&in,&out);
 return 1;
}

void mouse(int &x,int &y,int &c)
{

  in.x.ax=3;
  int86(51,&in,&out);
  c=out.x.bx;
  x=out.x.cx;
  y=out.x.dx;
}
void set_min_max()
{
 in.x.ax=8;
 in.x.cx=0;
 in.x.dx=getmaxy();
 int86(51,&in,&out);
 }

main()
{

  int temp,j,a[100][100],x,y,d,c,x1,y1,gd=DETECT,gm,i=0,k,xi[100],n;
  float dy,dx,slope[100];
  initgraph(&gd,&gm,"C://TC//BGI");
  set_min_max();
  do
```

```
 {
   callmouse();

   mouse(x1,y1,c);

   if(c==1)
   {
    if(a[i-1][0]!=x1&&a[i-1][1]!=y1)
    {
          a[i][0]=x1;
          a[i][1]=y1;
          i++;
          printf("%d %d\n",x1,y1);
    }
    hidemouse();
    delay(100);
   }
 }while(!kbhit());
 a[i][0]=a[0][0];
 a[i][1]=a[0][1];
 n=i;
for(i=0;i<n;i++)
{
setcolor(36);
line(a[i][0],a[i][1],a[i+1][0],a[i+1][1]);
}
 for(i=0;i<n;i++)
{
dy=a[i+1][1]-a[i][1];
dx=a[i+1][0]-a[i][0];
if(dy==0) slope[i]=1.0;
if(dx==0) slope[i]=0.0;
if((dy!=0)&&(dx!=0))
{
slope[i]=(float) dx/dy;
}
}
for(y=0;y< 480;y++)
{
k=0;
for(i=0;i<n;i++)
{
if( ((a[i][1]<=y)&&(a[i+1][1]>y))||
((a[i][1]>y)&&(a[i+1][1]<=y)))
{
xi[k]=(int)(a[i][0]+slope[i]*(y-a[i][1]));
k++;
}}
```

```
for(j=0;j<k-1;j++)
for(i=0;i<k-1;i++)
{
if(xi[i]>xi[i+1])
{
temp=xi[i];
xi[i]=xi[i+1];
xi[i+1]=temp;
}
}
setcolor(y%255);
for(i=0;i<k;i+=2)
{
line(xi[i],y,xi[i+1]+1,y);
getch();
}}
 delay(1000);
 closegraph();
 getch();
 return 0;
 }
```



/* Filling circle using Bresenham's Algorithm */

```
#include<stdio.h>
 #include<graphics.h>
 #include<conio.h>
 void cmp(int xc,int yc,int r)
 {int x=0;
  int yr;
  int y=r;
  float d;
  void pp(int,int,int,int);
  d=5/4-r;
  while(x<y)
  {
  if(d<0)
  d=d+(2*x+3);
  else
  {d=d+(2*(x-y)+5);
  y--;
  }
 pp(xc,yc,x,y);
 x++;
 }}
 void pp(int xc,int yc,int x,int y)
 {
 setcolor(WHITE);
 line(xc+x,yc-y,xc-x,yc-y);
 line(xc+x,yc+y,xc-x,yc+y);
 line(xc+y,yc+x,xc-y,yc+x);
 line(xc+y,yc-x,xc-y,yc-x);
 }
 main()
 {
 int xc,yc,r,a,b;
 int gd=DETECT,gm;
 initgraph(&gd,&gm,"C:\\TC\\BGI");
 a=getmaxx();
 b=getmaxy();
 setcolor(RED);
 line(a/2,0,a/2,b);
 line(0,b/2,a,b/2);
 printf("Enter the Center and radius");
 scanf("%d%d%d",&xc,&yc,&r);
 xc=xc+a/2;
 yc=b/2-yc;
 cmp(xc,yc,r);
 getch();
 closegraph();
```

```
return 0;
}
```