Name :) Amit Dobhal
Course :) BCA VI th
Subject :) Computer Graphics
Roll No :) 1121015

Qu1 :) Algorithm

Step1 :) Start
Step2 :) Declare variable x1, x2, y1, y2, d, i1, i2, dx, dy
Step3 :) Enter value of x1, x2, y1, y2
Where x1, y1 coordinate of starting point
~~Step4~~ and x2, y2 ,, ,, ,, end point
Step4 :) Calculate dx = x2 - x1
Calculate dy = y2 - y1
Calculate i1 = 2* dy
Calculate i2 = 2* (dy - dx)
Calculate d = i1 - dx
Step5 :) Consider (x, y) as starting point and xend as maximum possible value of x.

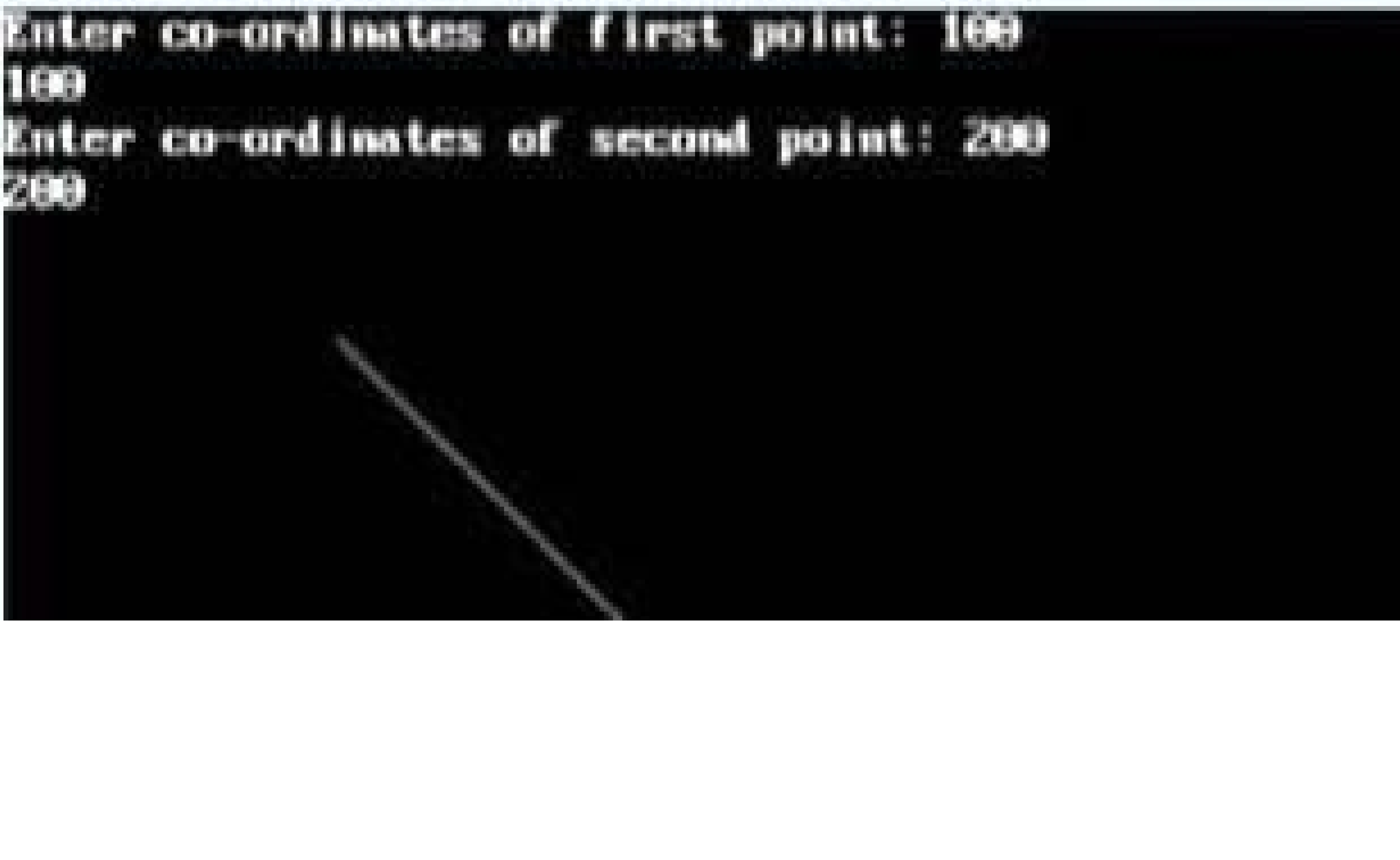
if dx < 0
Then x = x2
y = y2
xend = x1
If dx > 0
Then x = x1
y = y1
xend = x2

Step6 :) Generate point at (x, y) coordinate
Step 7 :) check if whole line is generated
if xy = xend
stop
Step 8 = Calculate co-ordinate of the pixel
if d<0 → Then d = d + i1 if d ≥ 0
Then : d = d + i2
Increment : y = y + 1

Step 9 = Increment x = x+1
Step 10 :- Draw a point of latest (x, y) coordinate
Step 11 :- Go to Step 7
Step 12 :- End of Algorithm.

Program :-

```
#include<stdio.h>
#include<graphics.h>
void drawline(int x0, int y0, int x1, int y1)
{
int dx, dy, p, x, y;
dx = x1-x0;
dy = y1-y0;
x = x0;
y = y0;
p = 2*dy-dx;
while (x<x1)
{
if (p>=0)
{
putpixel (x, y, 7)
y= y+1;
p= p+2* dy -2* dx;
}
else
{
putpixel (x, y, 7);
p= p+2* dy; }
x=x+1;
} }
int main()
{
int gdriver = DETECT, gmode, error, x0, y0, x1, y1;
initgraph (&gdriver, &gmode, "\\tuboc3\\bgi");
Printf ("Enter co-ordinate of Firt point");
Scanf ("%d %d", &x0, &y0);
Print ("Enter Secondpoint of co-ordinate");
Scanf ("%d%d", &x1 &y1);
drawline (x0, y0, x1, y1);
return 0;
}
```



Name :) Amit Dobhal
Course :) BCA VIth
Subject :) Computer Graphics
Roll No. :) 1121015

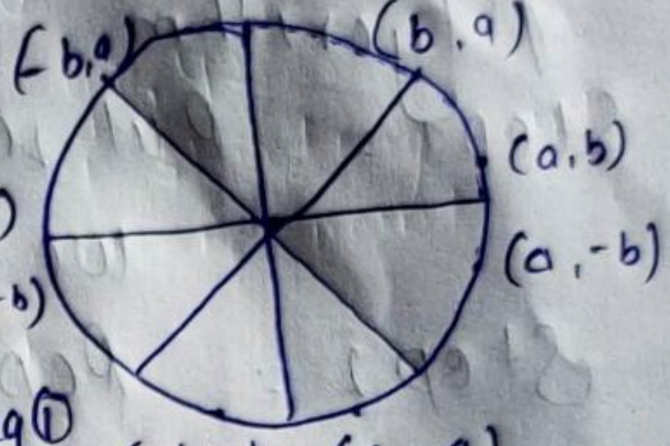
Qus 2 :) Mid point circle Algorithm.

Algorithm

~~Step 1 :) Put x = 0, y = r~~

$f(x,y) = x^2 + y^2 - r^2$

→ [< 0 for (x,y) inside circle
= 0 for (x,y) on circle
> 0 for (x,y) outside of circle] — eq ①



midpoint in $(x_{i+1}, y_i - 1/2)$

$p_i = f(x_{i+1}, y_i - 1/2) = (x_{i+1})^2 + (y_i - 1/2)^2 - r^2$ — eq ②

if p is -ve we choose pixel T

if p is +ve " " " S

The decision parameter for next step is

$p_{i+1} = (x_{i+1}+1)^2 + (y_{i+1} - 1/2)^2 - r^2$ — eq ③

Since $x_{i+1} = x_{i+1}$

$p_{i+1} - p_i = ((x_i + 1) + 1)^2 - (x_i + 1)^2 + (y_{i+1} - 1/2)^2 - (y_i - 1/2)^2$

$= x_i^2 + 4 + 4x - x_i^2 + 1 - 2x + y_{i+1}^2 + 1/4 - y_{i+1} - y_i^2 - 1/4 - y_i$

$= 2(x_i + 1) + 1 + (y_{i+1}^2 - y_i^2) - (y_{i+1} - y_i)$ — eq ④

if pixel T choosen $\Rightarrow p_i < 0$

we have $y_{i+1} = y_i$

if pixel S is choosen $= p_i \geq 0$

we have $y_{i+1} = y_i - 1$

Thus $p_{i+1} = \begin{bmatrix} p_i + 2(x_i+1) + 1, & p_i < 0 \\ p_i + 2(x_i+1) + 1 - 2(y_i - 1) & p_i \geq 0 \end{bmatrix}$ eq ⑤

$$p_{i+1} = \begin{bmatrix} p_i + 2x_i + 3 & \text{if } p_i < 0 \\ p_i + 2(x_i - y_i) + 5, & \text{if } p_i \geq 0 \end{bmatrix} \text{ eq ⑥}$$

Now intial value of pi $(0, r)$ from eq ②

$$p_1 = (0+1)^2 + (r - 1/2)^2 - r^2$$
$$= 1 + 1/4 - r^2 = 5/4 - r \quad [5/4 \simeq 1]$$

So $p_1 = 1 - r$

Algorithm

Step 1 = Put $x = 0$, $y = r$ in eq ②
we have $p = 1 - r$

Step 2 = Repeat Step while $x \leq y$
Plot (x, y)
if $(p < 0)$
Then set $p = p + 2x + 3$
else
$p = p + 2(x - y) + 5$
$y = y - 1$ (end if)
$x = (x + 1$ (end loop)

Step 3 — end

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

class bresen
{
float x,y,a,b,r,p
public:
void get();
void cal();
};
void main()
{
bresen b;
b.get();
b.cal();
getch();
}
void bresen::get()
{
cout<<"Enter center and radious";
cout<<"Enter (a,b)";
cin>>a>>b;
Cout<<"Enter r";
cin>>r;
}
void bresen::cal()
{
int gdriver=DETECT, gmode, errorcode;
int midx, midy, i;
initgraph(&gdriver, &gmode, "");
errorcode=graphresult();
if(errorcode!=grok)
{
```

```
{
Printf("Graphicerror %s", grapherrormsg(errocode);
Printf(" Press any key to halt ");
 getch ();
 exit (1);
}
   x = 0;
   y = r;
Putpixe(a, b+r, RED);
"       (a, b-r, RED);
 Putpixel(a-r, b, RED);
Putpixel (a+r, b, RED);
 P = 5/4 -r;
   while (x<=y)
   {
    if (P<0)
     P+= (u*x)+6;
    else
   { P+ =(2*(x-y)+5);
      y--;
    }
    x++;
  Putpixel (a+x, b+y, RED);
Putpixel (a-x, by , RED);
Put Pixel (a+x, b-y , RED);
Put Pixel (a+b , b-y , RED);
Put Pixel ( a*x , b+y , RED);
Put Pixel ( a+b, b-y, RED);
Put Pixel (a-x, b+y, RED);
Put Pixel (a-x, b-y, RED);
}
}
```