Name: Shaikh Salif Aminuddin
CLASS: S2 **ROLL NO:** 2201094

EXPERIMENT NO: 3

AIM: Implement Midpoint Circle algorithm

THEORY: The mid-point circle drawing algorithm is an algorithm used to determine the points needed for rasterizing a circle. We use the mid-point algorithm to calculate all the perimeter points of the circle in the first octant and then print them along with their mirror points in the other octants. This will work because a circle is symmetric about its centre.

The advantages of Midpoint circle drawing Algorithm:

1. It is an efficient algorithm
2. It uses simple equation on which the algorithm is based.
3. It is easy to understand and implement.
4. The algorithm is used to dig out scan conversion algorithm for drawing geomatric curves on raster display

The disadvantages of Midpoint circle drawing Algorithm:

1. Inefficient in generating smooth circle as distance between generated pixels are not same
2. It is time consuming
3. Not suitable for high graphic images

Algorithm:

Step1: Put x =0, y =r in equation 2
We have p=1-r

Step2: Repeat steps 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)

Step3: End

Problem:

Given the centre point coordinates (0, 0) and radius as 10, generate all the points to form a circle.

Centre Coordinates of Circle (X_0, Y_0) = (0, 0)
Radius of Circle = 10

Step-01:

Assign the starting point coordinates (X_0, Y_0) as-

- $X_0 = 0$
- $Y_0 = R = 10$

Step-02:

Calculate the value of initial decision parameter P_0 as-

$P_0 = 1 – R$

$P_0 = 1 – 10$

$P_0 = -9$

Step-03:

As $P_{initial} < 0$, so case-01 is satisfied.

Thus,

- $X_{k+1} = X_k + 1 = 0 + 1 = 1$
- $Y_{k+1} = Y_k = 10$
- $P_{k+1} = P_k + 2 \times X_{k+1} + 1 = -9 + (2 \times 1) + 1 = -6$

Step-04:

Step-03 is executed similarly until $X_{k+1} >= Y_{k+1}$ as follows-

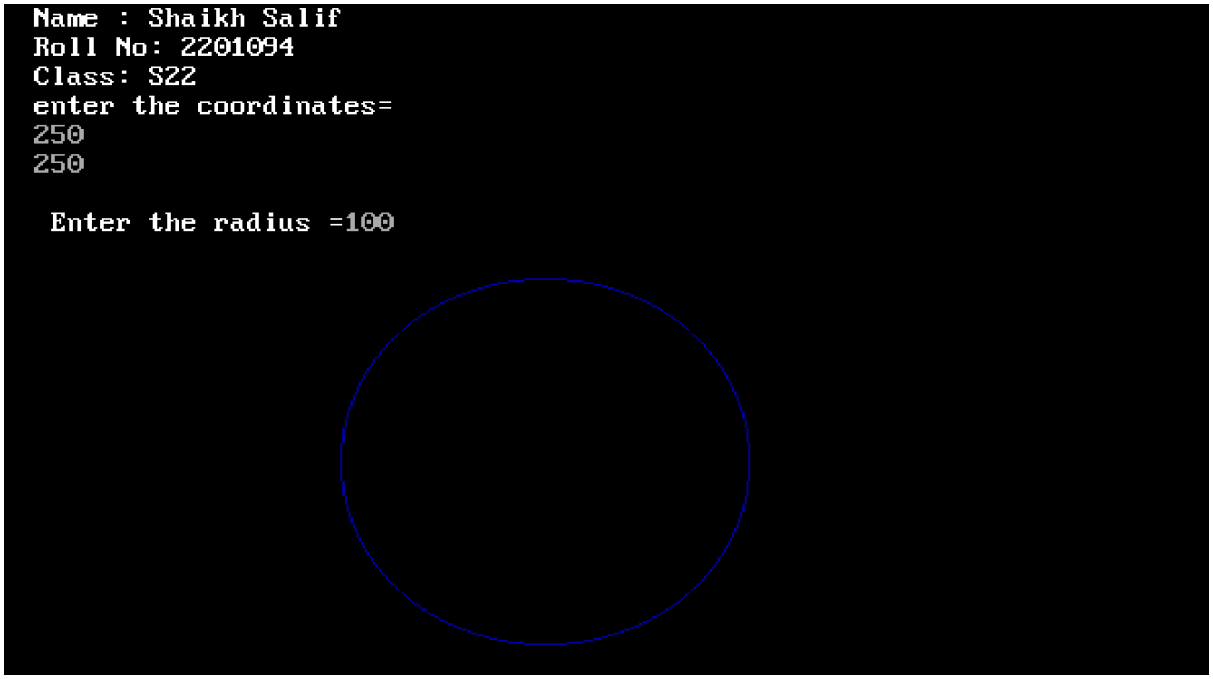
P_k	P_{k+1}	(X_{k+1}, Y_{k+1})
		(0, 10)
-9	-6	(1, 10)
-6	-1	(2, 10)
-1	6	(3, 10)
6	-3	(4, 9)
-3	8	(5, 9)
8	5	(6, 8)
Algorithm Terminates **These are all points for Octant-1.**		

CODE:

```
#include<conio.h>
#include<graphics.h>
void main()
{
int x,y,x_mid,y_mid,r,d;
int g_mode,g_driver=DETECT;
clrscr();
initgraph(&g_driver,&g_mode,"C:\\TURBOC3\\BGI");
printf("Name : Shaikh Salif\nRoll no: 2201094\nClass: S22");
printf("\nenter the coordinates=\n");
scanf("%d %d",&x_mid,&y_mid);
printf("\n now enter the radius =");
scanf("%d",&r);
x=0;
y=r;
d=1-r;
do
{
putpixel(x_mid+x,y_mid+y,1);
putpixel(x_mid+y,y_mid+x,1);
```

```
putpixel(x_mid-y,y_mid+x,1);
putpixel(x_mid-x,y_mid+y,1);
putpixel(x_mid-x,y_mid-y,1);
putpixel(x_mid-y,y_mid-x,1);
putpixel(x_mid+y,y_mid-x,1);
putpixel(x_mid+x,y_mid-y,1);
if(d<0)  {
d+=(2*x)+1;
}
else{
y=y-1;
d+=(2*x)-(2*y)+1;
}
x=x+1;
}while(y>x);
getch();
}
```

OUTPUT:



CONCLUSION: We have now learned the Midpoint circle algorithm in C. It works on some easy formulas mentioned in the theory part. Henceforth Implementation is successfully done using the program.