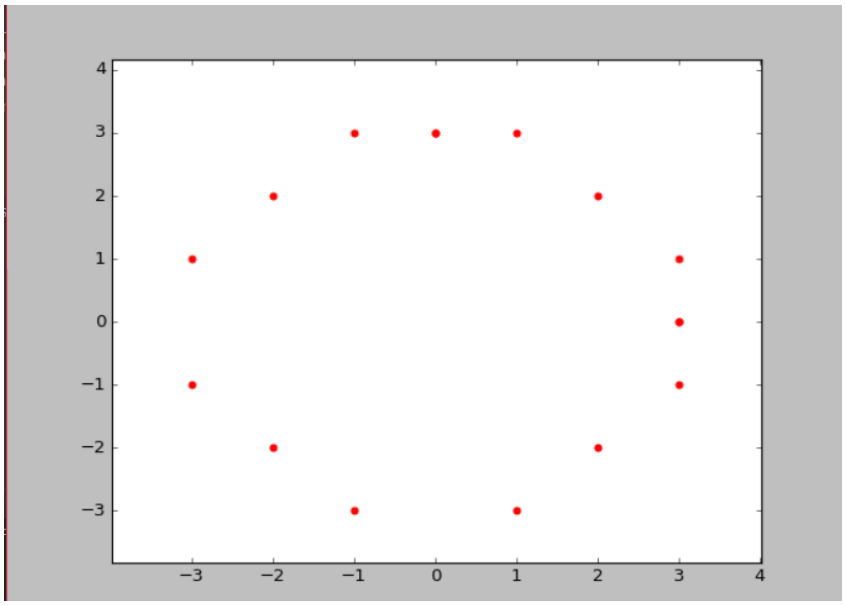
```
// C program for implementing
// Mid-Point Circle Drawing Algorithm
#include<stdio.h>

// Implementing Mid-Point Circle Drawing Algorithm
void midPointCircleDraw(int x_centre, int y_centre, int r)
{
    int x = r, y = 0;

    // Printing the initial point on the axes
    // after translation
    printf("(%d, %d) ", x + x_centre, y + y_centre);

    // When radius is zero only a single
    // point will be printed
    if (r > 0)
    {
        printf("(%d, %d) ", x + x_centre, -y + y_centre);
        printf("(%d, %d) ", y + y_centre, x + x_centre);
        printf("(%d, %d)\n", -y + y_centre, x + x_centre);
    }

    // Initialising the value of P
    int P = 1 - r;
    while (x > y)
    {
        y++;

        // Mid-point is inside or on the perimeter
        if (P <= 0)
            P = P + 2*y + 1;

        // Mid-point is outside the perimeter
        else
        {
            x--;
            P = P + 2*y - 2*x + 1;
        }

        // All the perimeter points have already been printed
        if (x < y)
            break;
```

```
        // Printing the generated point and its reflection
        // in the other octants after translation
        printf("(%d, %d) ", x + x_centre, y + y_centre);
        printf("(%d, %d) ", -x + x_centre, y + y_centre);
        printf("(%d, %d) ", x + x_centre, -y + y_centre);
        printf("(%d, %d)\n", -x + x_centre, -y + y_centre);

        // If the generated point is on the line x = y then
        // the perimeter points have already been printed
        if (x != y)
        {
            printf("(%d, %d) ", y + y_centre, x + x_centre);
            printf("(%d, %d) ", -y + y_centre, x + x_centre);
            printf("(%d, %d) ", y + y_centre, -x + x_centre);
            printf("(%d, %d)\n", -y + y_centre, -x + x_centre);
        }
    }
}

// Driver code
int main()
{
    // To draw a circle of radius 3 centred at (0, 0)
    midPointCircleDraw(0, 0, 3);
    return 0;
}
/*output*/
```



```
// C-program for circle drawing
// using Bresenham’s Algorithm
// in computer-graphics
#include <stdio.h>
#include <dos.h>
#include <graphics.h>
#include<conio.h>

// Function to put pixels
// at subsequence points
void drawCircle(int xc, int yc, int x, int y)
{
    putpixel(xc+x, yc+y, RED);
    putpixel(xc-x, yc+y, RED);
    putpixel(xc+x, yc-y, RED);
    putpixel(xc-x, yc-y, RED);
    putpixel(xc+y, yc+x, RED);
    putpixel(xc-y, yc+x, RED);
    putpixel(xc+y, yc-x, RED);
    putpixel(xc-y, yc-x, RED);
}

// Function for circle-generation
// using Bresenham's algorithm
void circleBres(int xc, int yc, int r)
{
    int x = 0, y = r;
    int d = 3 - 2 * r;
    while (y >= x)
    {
        // for each pixel we will
        // draw all eight pixels
        drawCircle(xc, yc, x, y);
        x++;

        // check for decision parameter
        // and correspondingly
        // update d, x, y
        if (d > 0)
        {
            y--;
            d = d + 4 * (x - y) + 10;
        }
```

```
            else
                d = d + 4 * x + 6;
            drawCircle(xc, yc, x, y);
            delay(50);
        }
}


// driver function
int main()
{   clrscr() ;
    int xc = 50, yc = 50, r2 = 30;
    int gd = DETECT, gm;
    initgraph(&gd, &gm, "C:\TURBOC3\BGI");  // initialize graph
    circleBres(xc, yc, r2);    // function call
    getch();
    return 0;
}
/*output*/
```