<-->

Sample Source Code :-

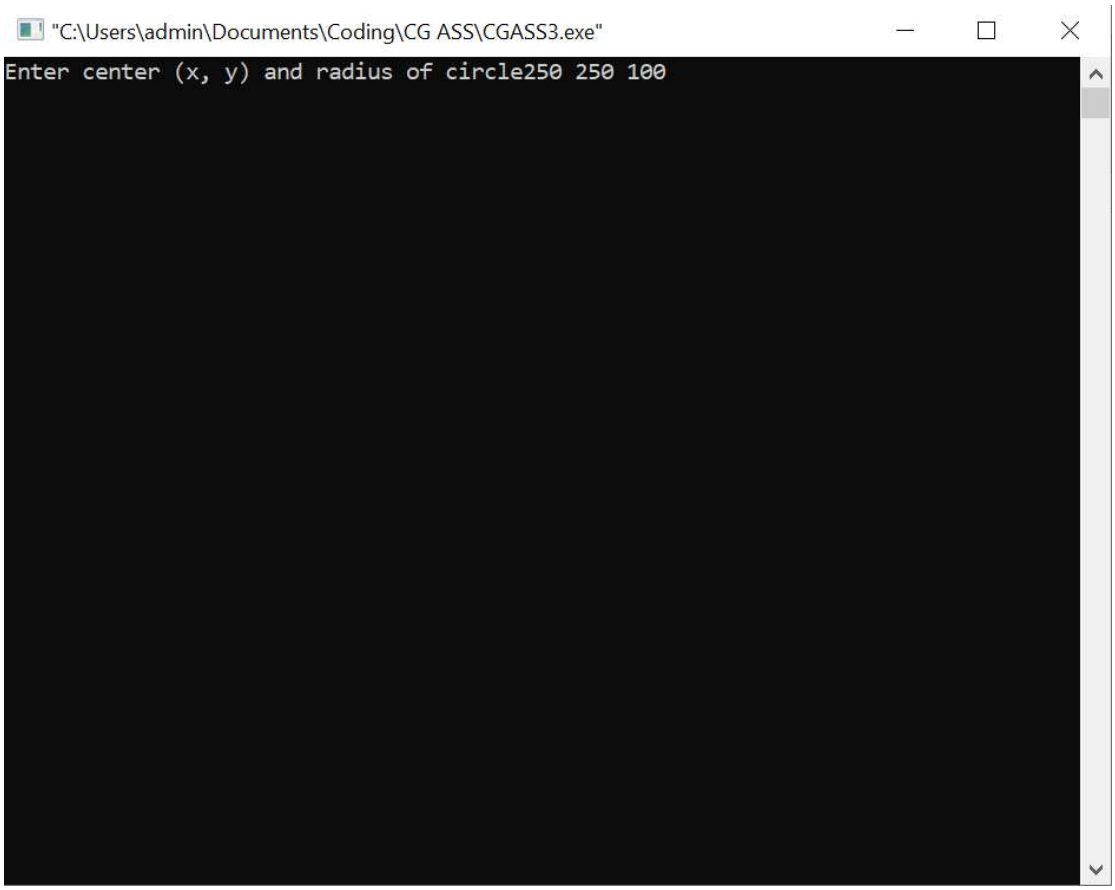
```
#include<iostream>
#include<graphics.h>
#include<math.h>
#include<conio.h>

using namespace std;

class Pattern
{
   int xc, yc, r;
   public:
      void getdata()
      {
         cout << "Enter center (x, y) and radius of circle";
         cin >> xc >> yc >> r;
      }
      void draw_pattern()
      {
         bresCircle(xc, yc, r);
         ddaLine (xc - sqrt(3)*r, yc + r, xc +sqrt(3)*r, yc + r);
         ddaLine (xc - sqrt(3)*r, yc + r, xc, yc - 2*r);
         ddaLine (xc + sqrt(3)*r, yc + r, xc, yc - 2*r);
         outtextxy (xc - sqrt(3)*r, yc + r, "(x1, y1)");
         outtextxy (xc + sqrt(3)*r, yc + r, "(x2, y2)");
         outtextxy (xc, yc - 2*r, "(x3, y3)");
         bresCircle (xc, yc, 2*r);
      }
      void ddaLine (int x1, int y1, int x2, int y2);
      void bresCircle (int xc, int yc, int r);

      int round (float p)
      {
         int q;
         q = p;
         if ((p - q) > 0.5)
            return q++;
         else
            return q;
      }
};

void Pattern :: ddaLine (int x1, int y1, int x2, int y2)
{
   int steps, i, dx, dy;
   float x, y, xinc, yinc;
   dx = x2 - x1;
   dy = y2 - y1;
```

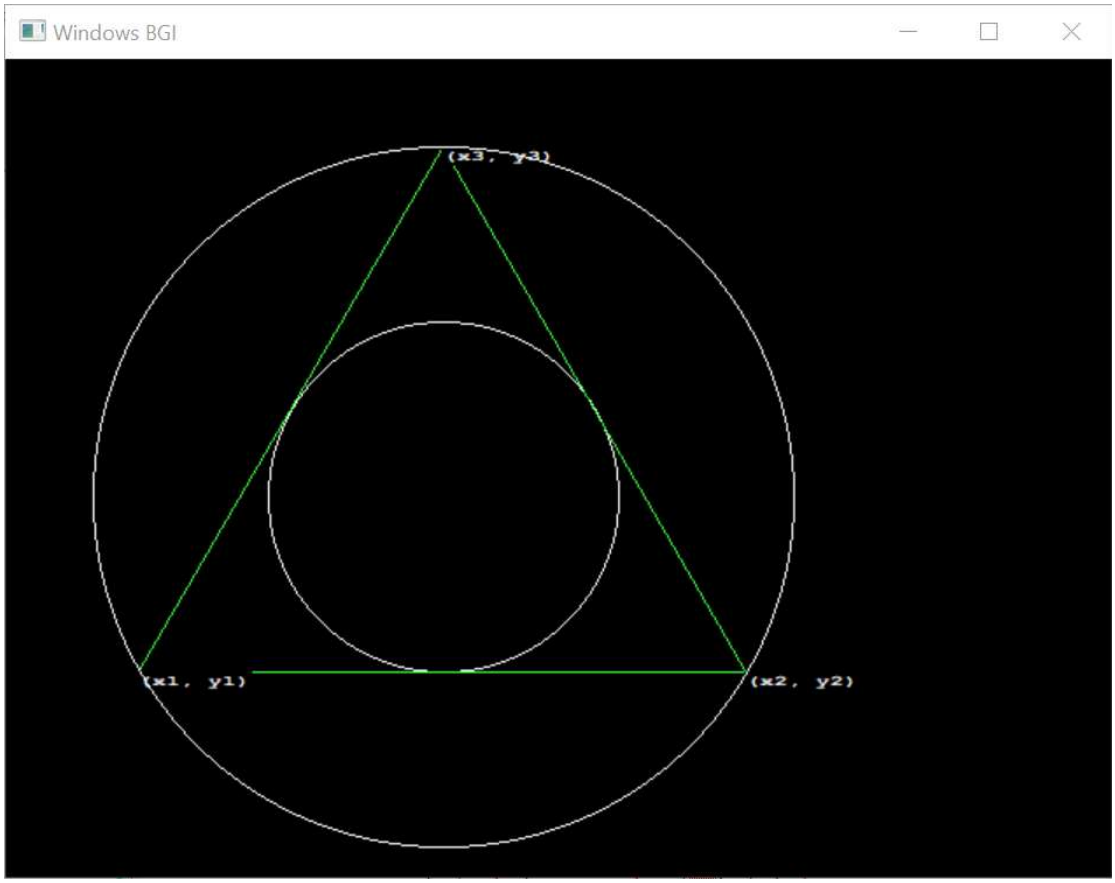
```
   if (abs(dx) > abs(dy))
      steps = abs(dx);
   else
      steps = abs(dy);
   xinc = dx / (float)steps;
   yinc = dy / (float)steps;
   x = x1;
   y = y1;
   putpixel(x, y, 10);
   for (i = 0; i < steps; i++)
   {
      x = x+xinc;
      y = y+yinc;
      putpixel(round(x), round(y), 10);
      delay(20);
   }
}

void display(int xc, int yc, int x, int y)
{
   putpixel(xc + x, yc + y, 15);
   putpixel(xc + y, yc + x, 15);
   putpixel(xc + y, yc - x, 15);
   putpixel(xc + x, yc - y, 15);
   putpixel(xc - x, yc - y, 15);
   putpixel(xc - y, yc - x, 15);
   putpixel(xc - y, yc + x, 15);
   putpixel(xc - x, yc + y, 15);
}

void Pattern :: bresCircle (int xc, int yc, int r)
{
   int x, y;
   float s;
   x = 0;
   y = r;
   s = 3 - 2*r;
   display(xc, yc, x, y);
   delay(20);
   while (x < y)
   {
      if (s <= 0)
      {
         s = s + 4 * x + 6;
         x = x + 1;
      }
      else
      {
         s = s + 4  * (x-y) + 10;
         x = x + 1;
         y = y - 1;
```

```
        }
        display(xc, yc, x, y);
        delay(20);
    }
}

int main()
{
    int gd = DETECT, gm;
    Pattern p1;
    p1.getdata();
    initgraph(&gd, &gm, "");
    cleardevice();
    p1.draw_pattern();
    getch();
    closegraph();
    return 0;
}
```

<---------->

Sample Output :-




Windows BGI
(x3, y3)
(x1, y1)
(x2, y2)