```cpp
/*Write C++ program to draw the following pattern.Use DDA line and
Bresenham drawing algorithm.Apply the concept of encapsulation.*/
#include <iostream>
#include <graphics.h>
#include <bits/stdc++.h>
    using namespace std;
class algo
{
public:
    void dda_line(float x1, float y1, float x2, float y2);
    void bresneham_cir(int r);
};
void algo::dda_line(float x1, float y1, float x2, float y2)
{
    float x, y, dx, dy, step;
    int i;
    // step 2
    dx = abs(x2 - x1);
    dy = abs(y2 - y1);
    cout << "dy=" << dy << "\tdx=" << dx;
    // step 3
    if (dx >= dy)
        step = dx;
    else
        step = dy;
    cout << "\n"
         << step << endl;
    // step 4
    float xinc = float((x2 - x1) / step);
    float yinc = float((y2 - y1) / step);
    // step 5
    x = x1;
    y = y1;
    // outtextxy(0,0,"(0,0)");
    // step 6
    i = 1;
    while (i <= step)
    {
        // cout<<endl<<"\t"<<i<<"\t(x,y)=("<<x<<","<<y<<")";
        putpixel(320 + x, 240 - y, 4);
        x = x + xinc;
        y = y + yinc;
        i = i + 1;
        // delay(10);
    }
}
void algo::bresneham_cir(int r)
{
    float x, y, p;
    x = 0;
    y = r;
    p = 3 - (2 * r);
    while (x <= y)
    {
        putpixel(320 + x, 240 + y, 1);
        putpixel(320 - x, 240 + y, 2);
        putpixel(320 + x, 240 - y, 3);
        putpixel(320 - x, 240 - y, 5);
        putpixel(320 + y, 240 + x, 6);
        putpixel(320 + y, 240 - x, 7);
        putpixel(320 - y, 240 + x, 8);
        putpixel(320 - y, 240 - x, 9);
        x = x + 1;
        if (p < 0)
        {
            p = p + 4 * (x) + 6;
        }
        else
        {
            p = p + 4 * (x - y) + 10;
            y = y - 1;
        }
        // delay(20);
    }
}
int main()
{
    algo a1;
    int i;
    float r, ang, r1;
    initwindow(630, 480);
    cout << "Enter radius of circle";
    cin >> r;
    a1.bresneham_cir((int)r);
    ang = 3.24 / 180;
    float c = r * cos(30 * ang);
    float s = r * sin(30 * ang);
    a1.dda_line(0, r, 0 - c, 0 - s);
    a1.dda_line(0 - c, 0 - s, 0 + c, 0 - s);
    a1.dda_line(0 + c, 0 - s, 0, r);
    r1 = s;
    a1.bresneham_cir((int)r1);
    getch();
    closegraph();
    return 0;
}
```