

```

import java.applet.Applet;
import java.awt.*;

////////////////////////////////////
//
// The following program creates a fractal called the Mandelbrot set.
// It uses the class Raster to set the pixel.
//
// WRITTEN BY:
// ALAIN DADAIAN
//
////////////////////////////////////

public class Mandelbrot extends Applet
{
    static final int MAXCOL = 800, MAXROW = 600, MAXCOLOR = 16, LIMIT = 4, MAXITER = 160;
    static double modulus, pmin, pmax, qmin, qmax, dp, dq;
    static colors c[] = new colors [16];

    //-----
    // Holds the rgb values of the colors used.
    //-----
    public static void setColors()
    {
        c[0] = new colors(0, 0, 0); //black
        c[1] = new colors(255, 0, 0); //red
        c[2] = new colors(0, 255, 0); //green
        c[3] = new colors(0, 0, 255); //blue
        c[4] = new colors(255, 255, 0); //yellow
        c[5] = new colors(0, 255, 255); //cyan
        c[6] = new colors(255, 0, 255); //magenta
        c[7] = new colors(64, 64, 64); //dark grey
        c[8] = new colors(128, 128, 128); //grey
        c[9] = new colors(192, 192, 192); //light grey
        c[10] = new colors(255, 200, 0); //orange
        c[11] = new colors(255, 175, 175); //pink
        c[12] = new colors(64, 192, 255); //light blue
        c[13] = new colors(174, 29, 29); //brown
        c[14] = new colors(139, 79, 40); //burgundy
        c[15] = new colors(55, 153, 17); //dark green
    }

    //-----
    // Initializes pmax, pmin, qmax, and qmin, and does the calc for dp and dq.
    //-----
    public void initialize()
    {
        pmax = 0.5;
        pmin = -1.6;
        qmin = -1.1;
        qmax = 1.1;

        dp = (pmax - pmin) / (MAXCOL - 1);
        dq = (qmax - qmin) / (MAXROW - 1);
    }

    public void init()
    {
        setBackground(Color.white);
    }

    public void paint(Graphics g)
    {
        setColors();

        double p, q, xlast, ylast, xcur, ycur;
        int color;
        initialize();
        int col, row;

        for (col = 0; col < MAXCOL; col++)
            for (row = 0; row < MAXROW; row++)
            {
                p = pmin + col * dp;
                q = qmin + row * dq;

                xlast = 0.0;

```

```

y1ast = 0.0;
modulus = 0.0;
color = 0;

while((modulus < LIMIT) && (color < MAXITER))
{
    color++;
    xcur = p + x1ast * x1ast - y1ast * y1ast;
    ycur = q + 2 * x1ast * y1ast;

    x1ast = xcur;
    y1ast = ycur;

    modulus = xcur * xcur + ycur * ycur;
}

if(color == MAXITER)
    color = 0;

    color %= MAXCOLOR;

// sets the color for the dot
g.setColor(new Color(c[color].r, c[color].g, c[color].b));

// draws the the dot or sets pixel
g.drawRect(col, row, 1, 1);
}
}

////////////////////////////////////
//
//          Creates an object colors to hold the rgb values for the colors.
//
//          //
////////////////////////////////////

class colors
{
    int r, b, g;

    //-----
    // Constructor for the colors class.
    //-----
    public colors()
    {}

    //-----
    // Second Constructor for the colors class it takes in three arguements
    // r, g, b all of type int and is used to store the rgb values of the
    // colors.
    //-----
    public colors(int r, int g, int b)
    {
        this.r = r;
        this.g = g;
        this.b = g;
    }
} // end of class colors

////////////////////////////////////

```