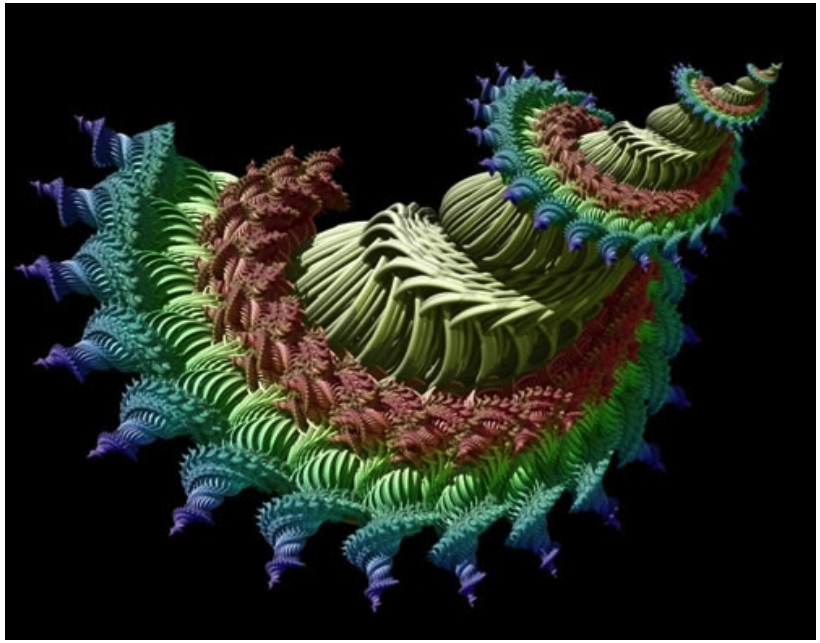
La Ricorsione



Terza Parte

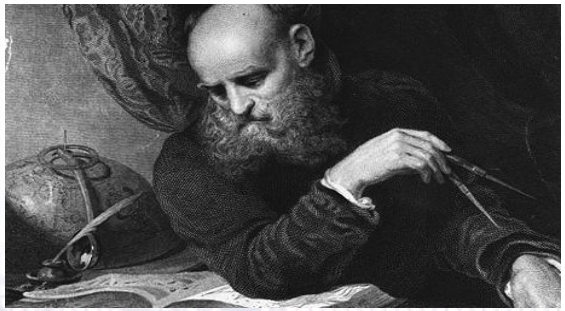
Giselda De Vita

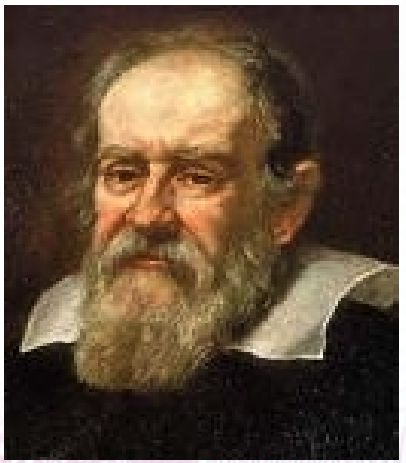
La natura è ricorsiva

La natura è matematica

"Il libro della natura è scritto in lingua matematica ed i suoi caratteri sono triangoli, cerchi ed altre figure geometriche, senza i quali mezzi è impossibile intenderne umanamente parola; senza questi è un aggirarsi vanamente per un oscuro labirinto."

(Galileo Galilei)





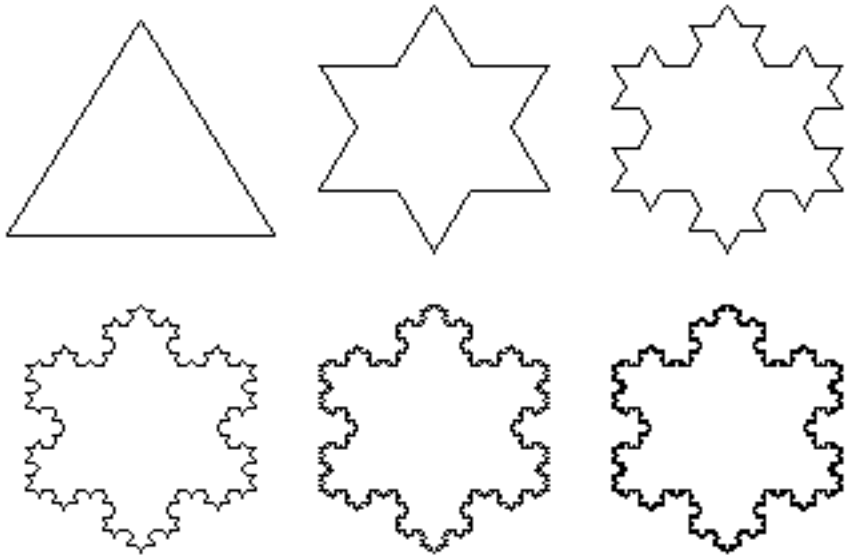
La natura è irregolare

"*La geometria euclidea è incapace di descrivere la natura nella sua complessità, in quanto si limita a descrivere tutto ciò che è regolare […] Le montagne non sono dei coni, le nuvole non sono delle sfere, le coste non sono dei cerchi, ma sono oggetti geometricamente molto complessi.*"

(Benoit Mandelbrot)

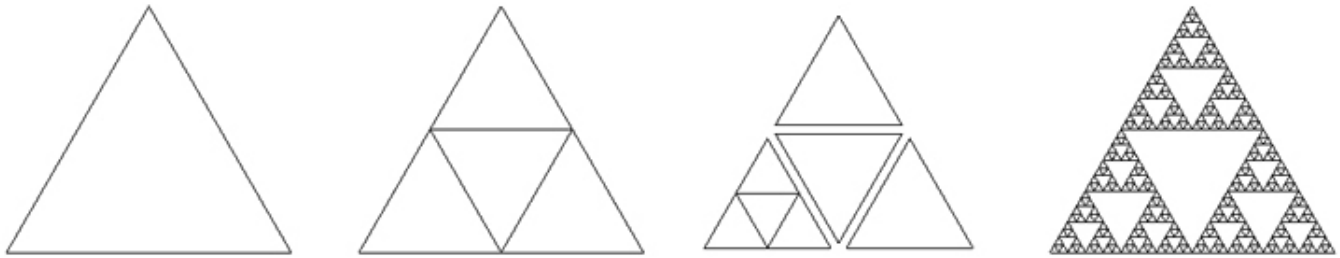
I frattali

I frattali sono figure geometriche caratterizzate dal ripetersi sino all'infinito di uno stesso motivo su scala sempre più ridotta.



Ricorsività e frattali

I procedimenti iterativi, o meglio ricorsivi, sono lo strumento base per lo studio delle immagini frattali.



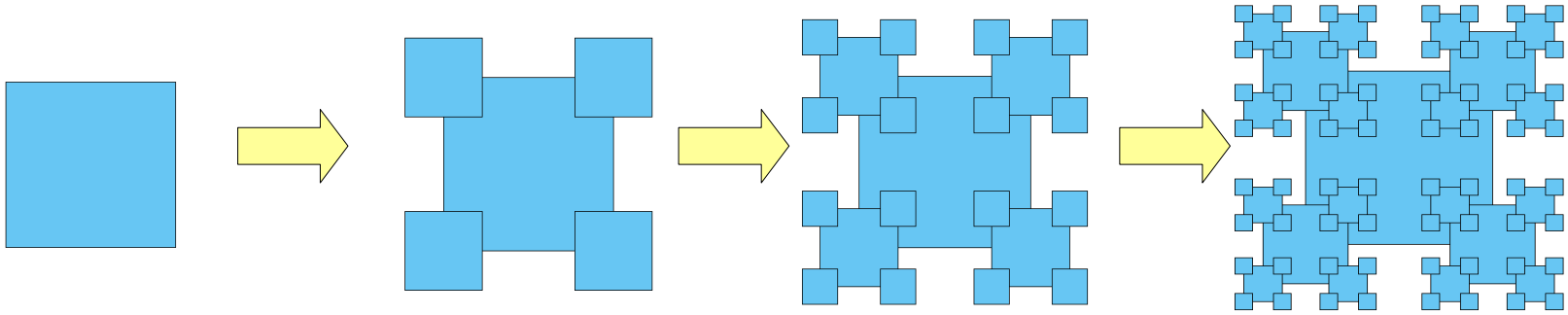
StdDraw.java

Inserire nel proprio progetto la classe:

http://introcs.cs.princeton.edu/java/stdlib/StdDraw.java

E’ utilissima per effettuare tutte le operazioni basilari di disegno dei frattali lineari che andremo a implementare.

Frattale Quadrati ricorsivi



Da un quadrato di partenza, costruiamo ad ogni iterazione un quadrato centrato in ogni vertice che ha esattamente il lato dimezzato rispetto al lato del quadrato originario.

Implementazione

La funzione ricorsiva di disegno del quadrato si implementa in queste poche righe:

```
// plot an order n recursive squares pattern
// centered on (x, y) of the given side length
public static void draw(int n, double x, double y, double size) {
    if (n == 0) return;

    drawSquare(x, y, size);

    // 2.2 ratio looks good
    double ratio = 2.2;

    // recursively draw 4 smaller trees of order n-1
    draw(n-1, x - size/2, y - size/2, size/ratio);    // lower left
    draw(n-1, x - size/2, y + size/2, size/ratio);    // upper left
    draw(n-1, x + size/2, y - size/2, size/ratio);    // lower right
    draw(n-1, x + size/2, y + size/2, size/ratio);    // upper right
}
```

Implementazione

La funzione ricorsiva di disegno richiama, ad ogni esecuzione, la funzione drawSquare che è quella che disegna il quadrato:

```
// plot a square, centered on (x, y) of the given side length
// with a light blue background and black border
public static void drawSquare(double x, double y, double size) {
    StdDraw.setPenColor(StdDraw.BOOK_LIGHT_BLUE);
    StdDraw.filledSquare(x, y, size/2);
    StdDraw.setPenColor(StdDraw.BLACK);
    StdDraw.square(x, y, size/2);
}
```

Funzioni standard della libreria StdDraw per settare il colore

Disegna solo i bordi di un quadrato

Disegna un quadrato pieno di centro x,y e lato size/2

Implementazione

Ipotizziamo una metodo main che effettua la prima chiamata a draw()

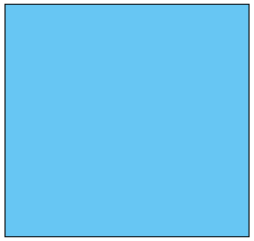
```
// read in a command-line argument N and plot an order N recursive
// squares pattern
public static void main(String[] args) {
    int N = Integer.parseInt(args[0]);
    double x = 0.5, y = 0.5;   // center of square
    double size = 0.5;         // side length of square
    draw(N, x, y, size);
}
```

Numero di iterazioni sulla ricorsione

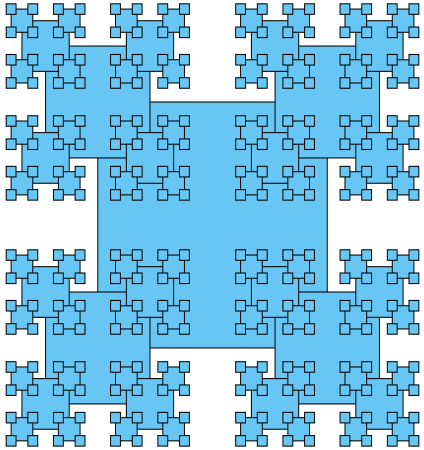
Centro della finestra di dimensione 1

Prima chiamata a draw()

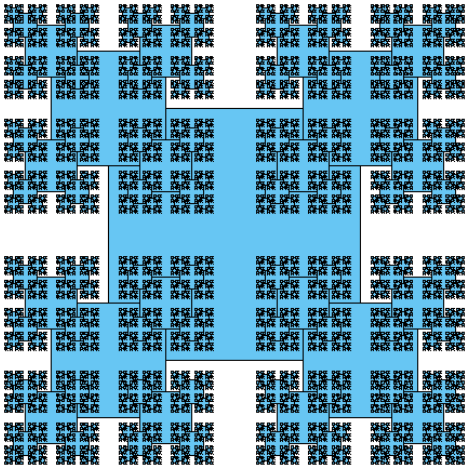
Iterazioni



Ouput per 1 sola iterazione

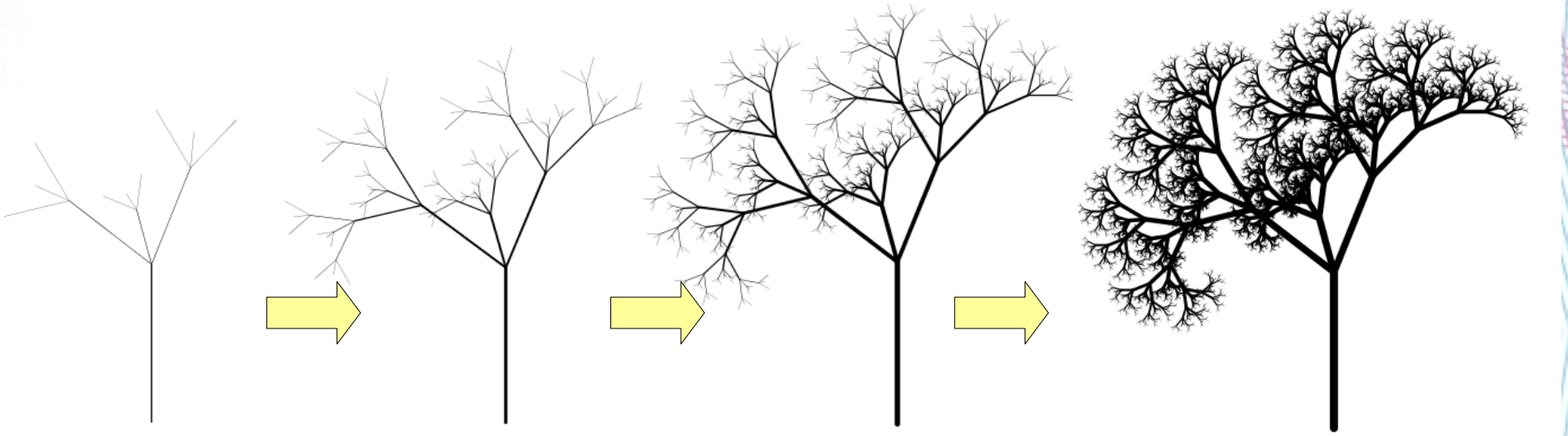


Ouput per 5 iterazioni



Ouput per 8 iterazioni

Frattale albero



Una linea che si divide in tre con una leggera pendenza a sinistra per rendere il nostro albero più credibile.

Implementazione

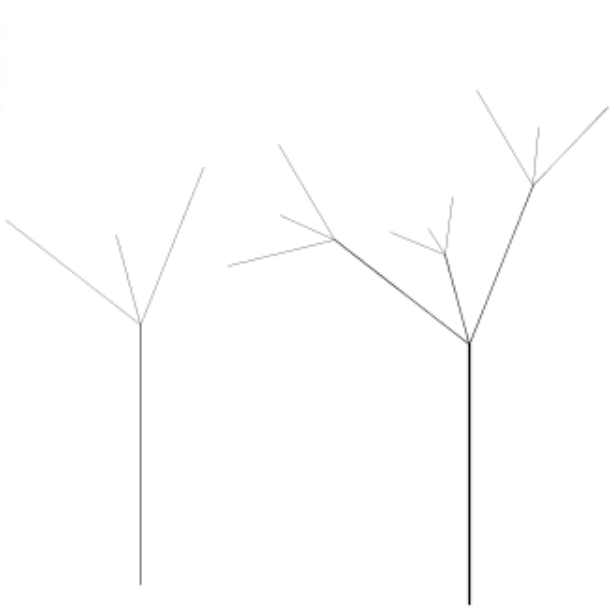
```
public class Tree {

      final static double bendAngle   = Math.toRadians(15);
      final static double branchAngle = Math.toRadians(37);
      final static double branchRatio = .65;

   public static void tree(int n, double x, double y, double a, double branchRadius) {
      double cx = x + Math.cos(a) * branchRadius;
      double cy = y + Math.sin(a) * branchRadius;
      StdDraw.setPenRadius(.001 * Math.pow(n, 1.2));
      StdDraw.line(x, y, cx, cy);
      if (n == 0) return;

      tree(n-1, cx, cy, a + bendAngle - branchAngle, branchRadius * branchRatio);
      tree(n-1, cx, cy, a + bendAngle + branchAngle, branchRadius * branchRatio);
      tree(n-1, cx, cy, a + bendAngle,               branchRadius * (1 - branchRatio));
   }

   public static void main(String[] args) {
      int N = Integer.parseInt(args[0]);
      tree(N, .5, 0, Math.PI/2, .3);
   }
}
```

Costanti per dare La forma all'albero

Costante di riduzione della size del ramo

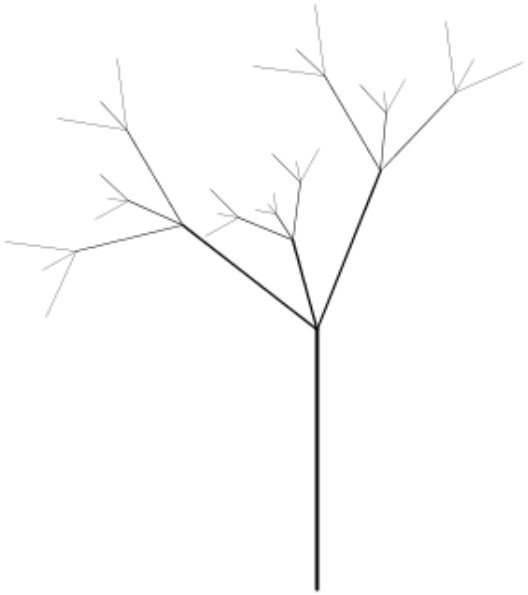
Formula per disegnare i rami mano mano più sottili

Iterazioni

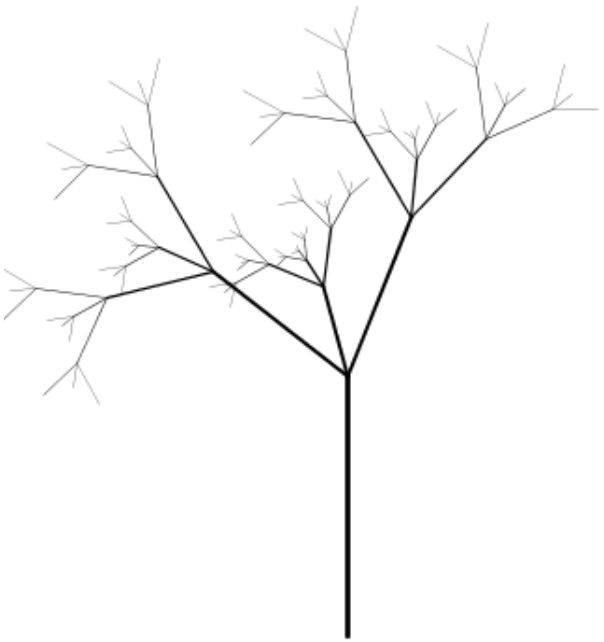


Ouput per 1 sola iterazione

Ouput per 2 iterazioni



Ouput per 3 iterazioni



Ouput per 4 iterazioni

Approfondimenti

- Frattali - http://www.rudimathematici.com/blocknotes/pdf/RMNC.pdf
- Java awt e Swing - http://www.sti.uniurb.it/lattanzi/SIM/esercitazioni/Es3.pdf