ÜBUNG: REKURSION

1 Füllen von pixeldefinierten Flächen (Papierübung)

Folgende Methode `colorArea()` füllt bzw. färbt eine Fläche pixelweise mit einer bestimmten Farbe (`fillColor`). Die Ausdehnung der zu färbenden Fläche ergibt sich durch die anders gefärbte Umgebung (`outsideColor`). Aufgerufen wird die Prozedur mit den Koordinaten (`x`, `y`) eines zu färbenden Pixels.

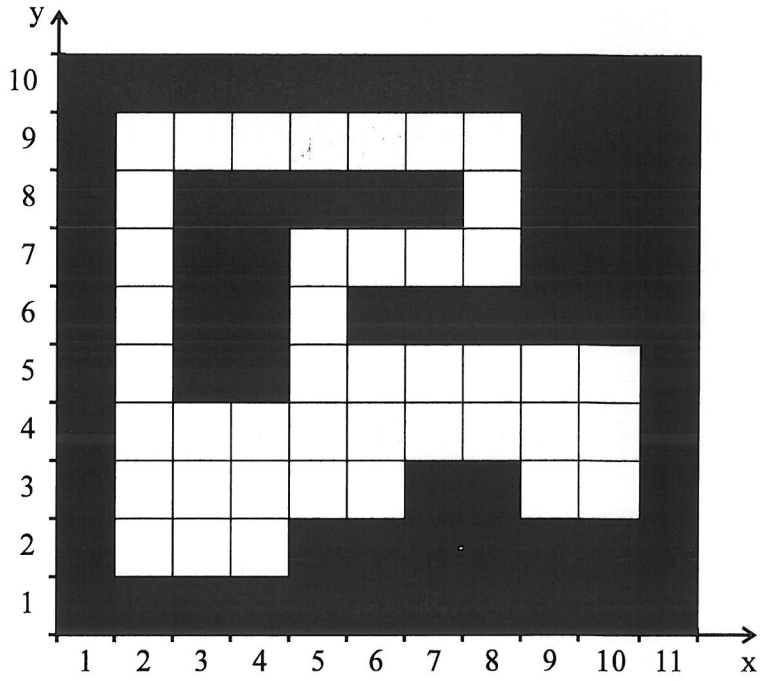
```
void colorArea(int x, int y, Color fillColor,
Color outsideColor) {

  Color actualColor;
  actualColor = getActualColor(x, y);

  if ((actualColor != outsideColor) &&
  (actualColor != fillColor)) {
    colorPixel(x, y, fillColor);
    colorArea(x+1, y, fillColor, outsideColor);
    colorArea(x, y+1, fillColor, outsideColor);
    colorArea(x-1, y, fillColor, outsideColor);
    colorArea(x, y-1, fillColor, outsideColor);
  }
}
```

Nummerieren Sie in folgendem Bild die Pixel gemäss der Reihenfolge, wie sie von "weiss" auf "grau" gefärbt werden. Die zu färbende Fläche wird durch die "schwarze" Umgebung definiert. Die Methode wird wie folgt aufgerufen:

```
colorArea(5, 4, Color.gray, Color.black);
```



2 Fibonacci-Zahlen (Leonardo von Pisa, 1202)

Schreiben Sie ein Applet, das die n-te Fibonacci-Zahl F_n berechnet. n kann dabei auf eine geeignete Art vom Benutzer eingegeben werden.

Es gilt: $F_1 = 1, F_2 = 1$

$F_n = F_{n-1} + F_{n-2}$ für $n > 2$

Jede Zahl ist also die Summe der beiden vorangegangenen. Die Folge der Fibonacci-Zahlen beginnt somit wie folgt:

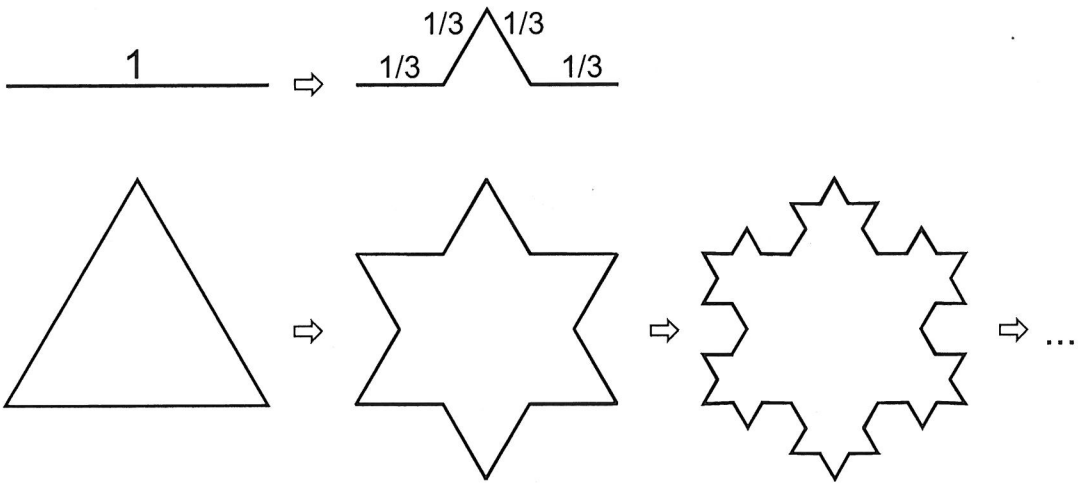
1, 1, 2, 3, 5, 8, 13, 21, ...

Implementieren Sie eine *iterative* und alternativ eine *rekursive* Methode!

Bemerkung: Für grosse n lässt sich F_n näherungsweise auch wie folgt berechnen. Falls Sie's interessiert, können Sie auch noch den Näherungswert ausgeben, z.B. $F_{14} \approx 377{,}001$ (exakt: $F_{14} = 377$).

$$F_n \approx \frac{1}{\sqrt{5}} \cdot \left[\frac{1}{2} \cdot \left(1+\sqrt{5}\right)\right]^n$$

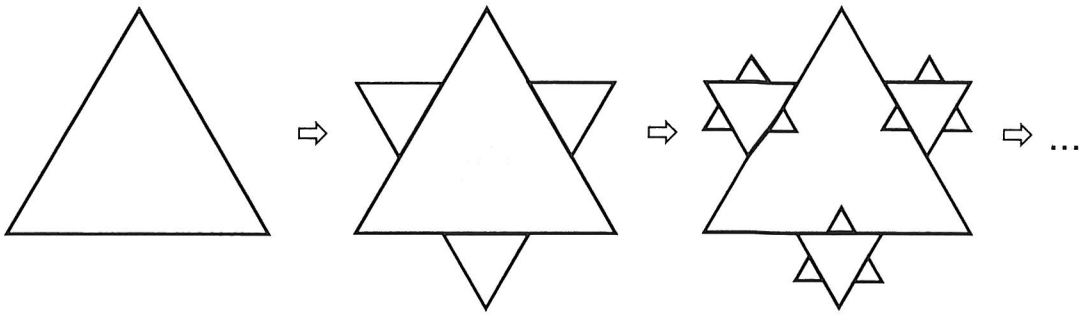
3 Koch'sche Insel (H. von Koch, 1904)



Aus den obigen Illustrationen sollte ersichtlich sein, wie die Koch-sche Insel sukzessive konstruiert werden kann. Programmieren Sie ein Applet, das die Koch'sche Insel zeichnet. Via Scrollbar kann die Länge einer Teilstrecke beeinflusst werden. Die Insel wird erst dann gezeichnet, wenn die Länge den vorgegeben Wert unterschreitet.

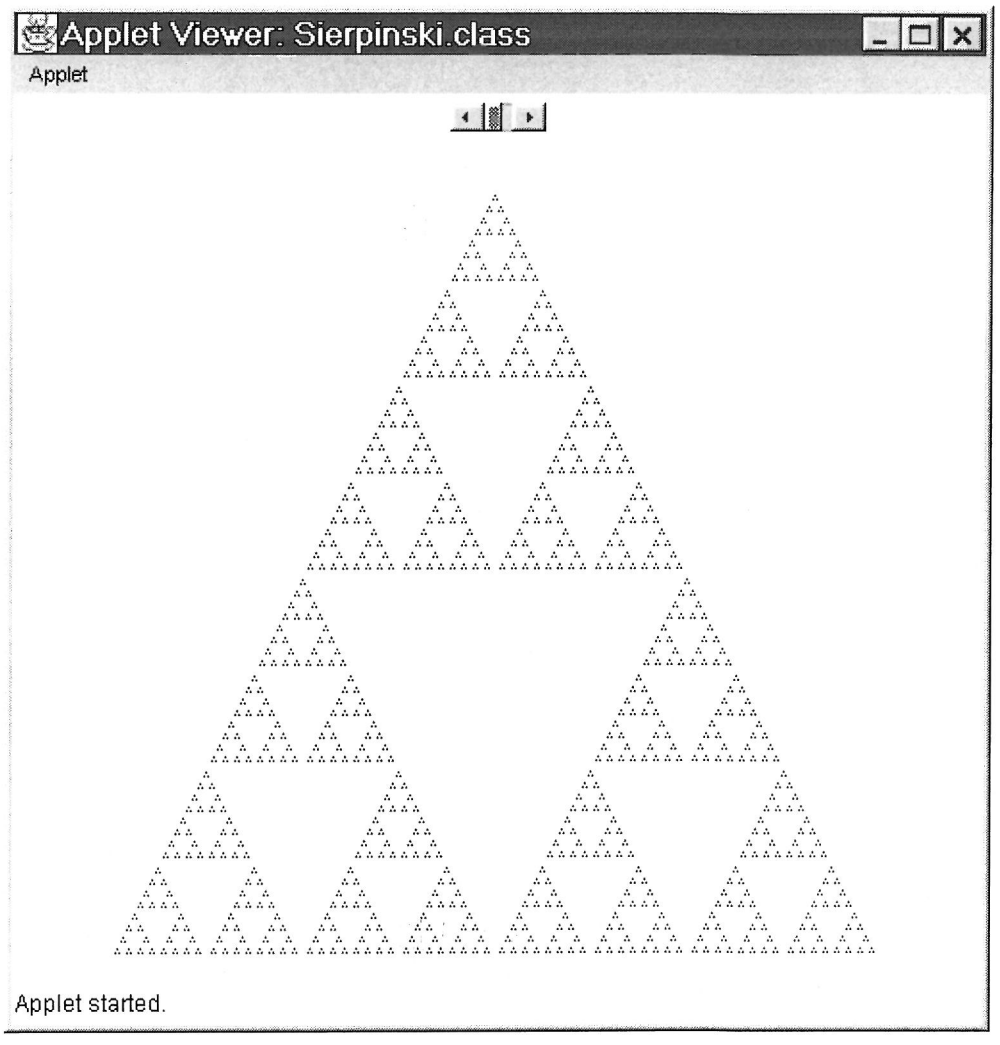
Bemerkungen: Mit jedem Schritt nimmt der Umfang bzw. die Küstenlinie der Insel um den Faktor 4/3 zu. Im Grenzfall resultiert ein *unendlich langer Umfang*, obwohl die *eingeschlossene Fläche endlich* ist! Dieses mathematische Gebilde besitzt die Dimension 1,2618...! Nach Mandelbrot (1977) werden Gebilde mit gebrochener Dimension als *Fraktale* bezeichnet.

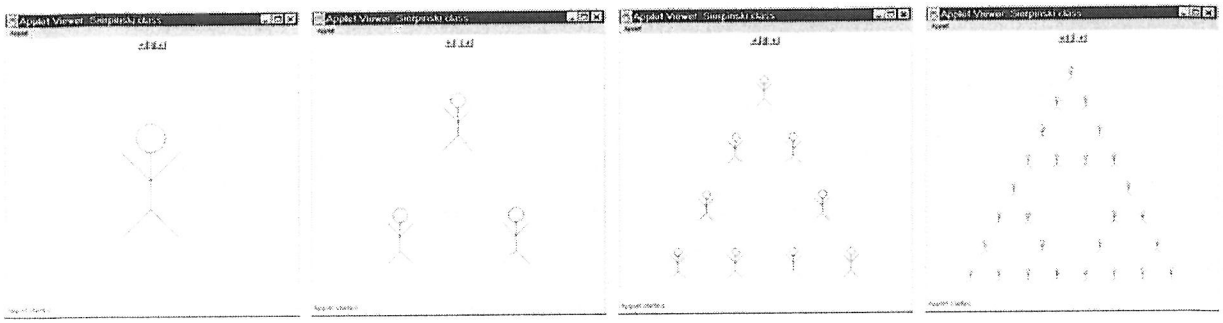
4 Schneeflocke



Programmieren Sie ein Applet, das wie oben illustriert eine Schneeflocke zeichnet. Mit einem Scrollbar kann die Länge einer Dreieckseite beeinflusst werden. Die Flocke wird erst dann gezeichnet, wenn die Länge den vorgegebenen Wert unterschreitet.

5 Sierpinski-Dreieck (1)





Das Sierpinski-Dreieck wird wie folgt erzeugt:

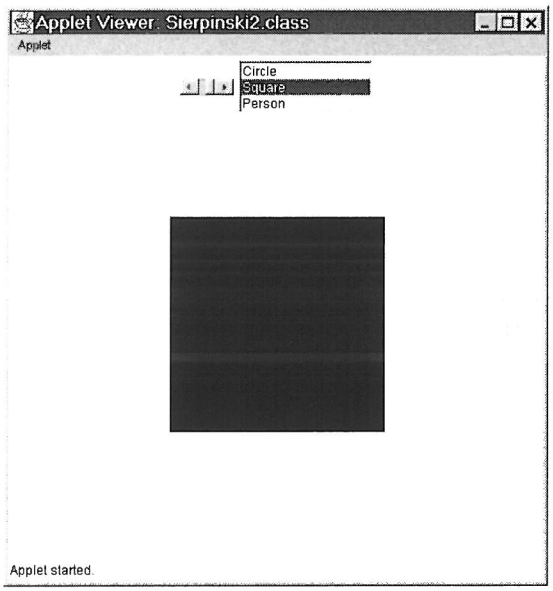
1. Die Originalfigur um 50% verkleinern.
2. Je eine Verkleinerung *links unten*, *rechts unten* sowie *oben* vorsehen.
3. Den 1. und 2. Schritt so lange wiederholen, bis die Figuren eine vorgegebene Maximalgrösse unterschritten haben; erst dann alle Figuren definitiv zeichnen.

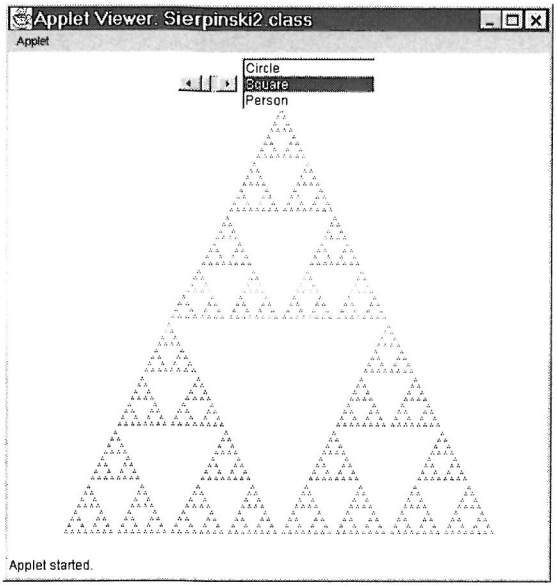
Unabhängig von einer bestimmten Originalfigur resultiert immer das Sierpinski-Dreieck! Das Endbild nennt man auch *Attraktor*.

Bemerkung: Bei der *fraktalen Bild-Komprimierung* geht man genau den umgekehrten Weg. Ein Bild zerlegt man möglichst geschickt in verschiedene Attraktoren und versucht für diese einfache Erzeugungsvorschriften zu finden. Die Erzeugungsvorschriften selbst erfordern dann nur noch extrem wenig Speicherplatz.

6 Sierpinski-Dreieck (2)

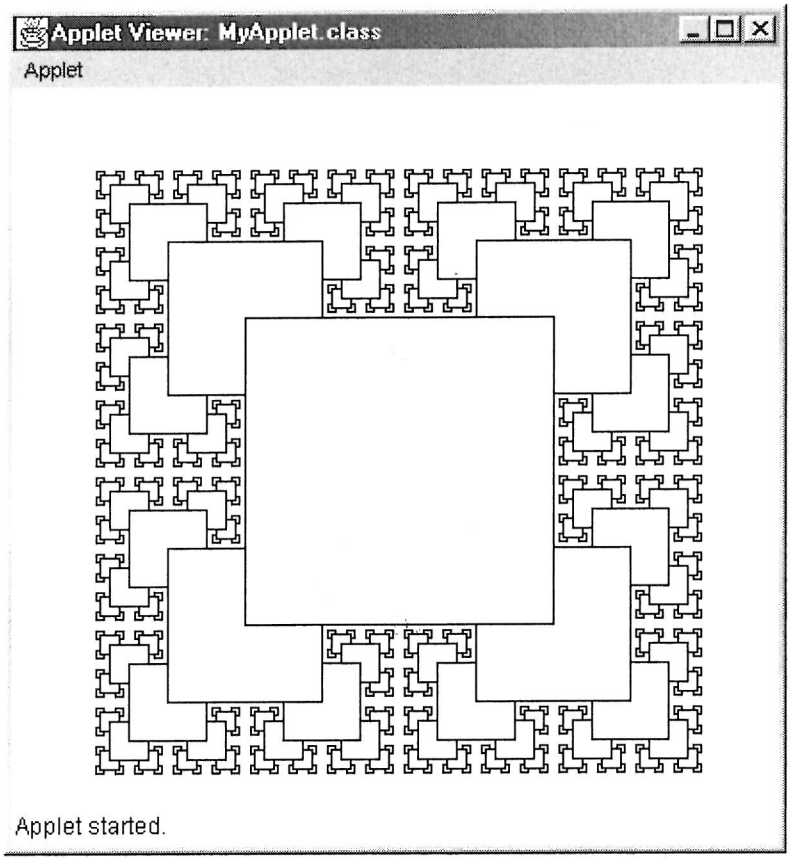
In dieser Aufgabe soll der Benutzer unter verschiedenen Originalfiguren auswählen können, z.B. Kreis, Quadrat und Strichmännchen (vgl. letzte Aufgabe). Die Auswahl wird mit Hilfe einer Liste realisiert (siehe `class List` und `interface ItemListener`).





7 Fraktal (2)

Schreiben Sie eine Methode `drawSpecial(Graphics g, int x, int y, int r)`, die wie folgt ein Fraktal zeichnet:



`x` und `y` Zeichnen legen das Zentrum des grossen Quadrates fest. `r` entspricht dem "Radius" des grossen Quadrates bzw. seiner halben Seitenlänge. Die "Radien" der vier nächst kleineren Quadrate betragen noch je r/2. Sobald der Radius kleiner gleich 1 ist, werden keine Quadrate mehr gezeichnet.

Implementieren Sie eine möglichst einfache und gut lesbare Methode. Etwa 10 bis 15 Zeilen Code sollten genügen!

Betten Sie die Methode in ein Applet ein.