FIBONACCI

Referat von Felix Schramm & Johannes Vogel

LEONARDO FIBONACCI

- * 1180 in Pisa † 1241 in Pisa
- Gilt als bedeutendster Mathematiker im Mittelalter
- Rechnete damals schon mit den „neun indo-arabischen Ziffern“
- Hauptwerk: *Liber abbaci* („Buch der Rechenkunst“)
- Geht in seinen Ansprüchen weit über alles hinaus, was dem lateinischen Mittelalter bis dahin bekannt war
- Legt besonderen Wert auf Beweise seiner Lösungen

DIE FIBONACCI - FOLGE

Die Fibonacci-Folge ist eine unendliche Folge von Zahlen (den Fibonacci - Zahlen), bei der sich die jeweils folgende Zahl durch Addition ihrer beiden vorherigen Zahlen ergibt.

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, …

0 + 1 = 1

1 + 1 = 2

1 + 2 = 3

2 + 3 = 5

BILDUNGSGESETZ

$$f_n = f_{n-1} + f_{n-2} \quad \text{für} \quad n \geq 2$$

mit den Anfangswerten

$$f_0 = 0 \text{ und } f_1 = 1$$

BEZIEHUNGEN ZWISCHEN DEN FOLGEGLIEDERN

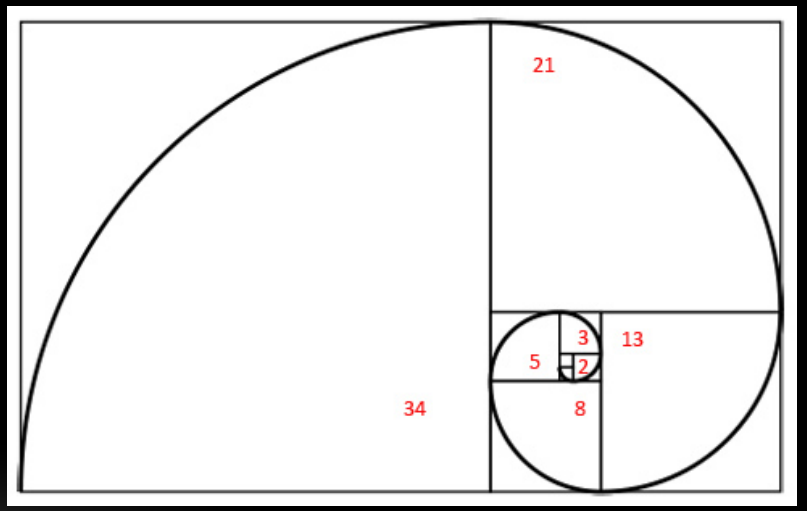
- $f_{m+n} = f_{n+1} * f_m + f_n * f_{m-1}$
- $f_{2n-1} = (f_n)^2 + (f_{n-1})^2$
- $f_{2n} = f_n * (f_{n+1} + f_{n-1}) = (f_n)^2 + 2 * f_n * f_{n-1}$
- $f_{2n+1} = (f_n)^2 + (f_{n+1})^2$
- $(f_n)^2 - f_{n+k} * f_{n-k} = (-1)_{n-k} * (f_k)^2$ *(Identität von Catalan)*
- $f_{n+1} * f_{n-1} – (f_n)^2 = (-1)^n$ *(Identität von Cassini)*

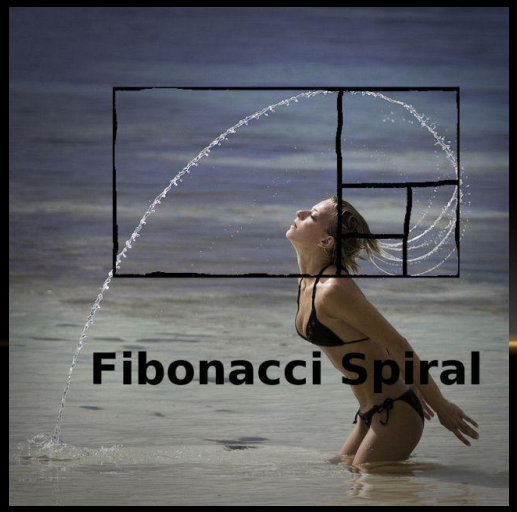
VERWANDTSCHAFT MIT DEM GOLDENEN SCHNITT

- $\lim_{n\to\infty}\left(\frac{f_{n+1}}{f_n}\right) = \lim_{n\to\infty}\left(\frac{\Phi^{n+1}}{\Phi^n}\right) = \Phi \approx 1{,}618$
- Φ ist eine irrationale Zahl. Das bedeutet, dass sie sich nicht durch ein Verhältnis zweier ganzer Zahlen darstellen lässt.
- Am besten lässt sich Φ durch Quotienten zweier aufeinander folgender Fibonacci-Zahlen darstellen.
- Dies gilt auch für entartete Fibonaccifolgen, bei denen f0 und f1 beliebige natürliche Zahlen annehmen.

FIBONACCI - FOLGE IN DER NATUR

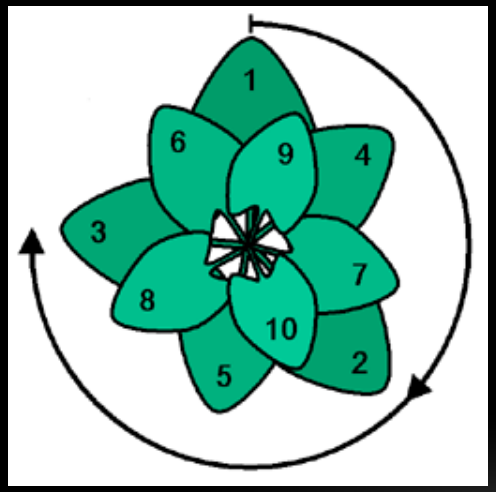
- Kettenbruchdarstellung
- $\frac{1}{1} = 1\,,\ \frac{2}{1} = 1 + \frac{1}{1}\,,\ \frac{3}{2} = 1 + \frac{1}{1+\frac{1}{1}}\,,\ \frac{5}{3} = 1 + \frac{1}{1+\frac{1}{1+\frac{1}{1}}}$
- Die Fibonacci-Spirale zeichnet sich dadurch aus, dass sich ihr Radius alle 90° im Verhältnis des goldenen Schnitt verändert. Dies ist vereinfacht in folgender Grafik dargestellt.

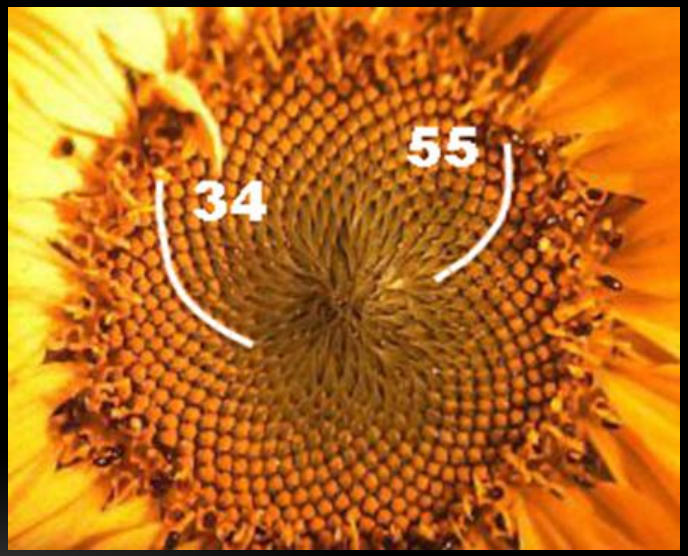




FIBONACCI - FOLGE IN DER NATUR

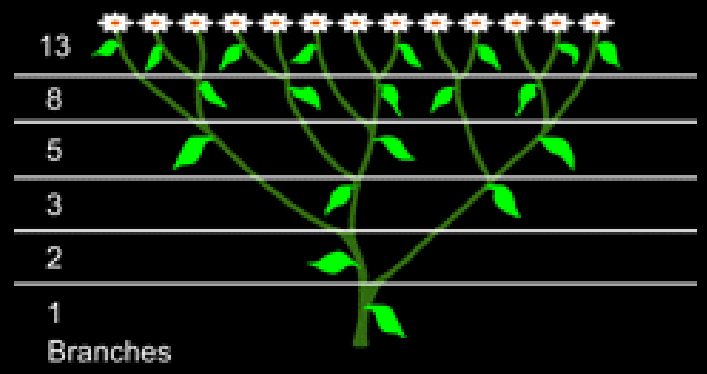
- Die Kerne der Sonnenblume sind in Fibonacci-Spiralen angeordnet. Die Zahl der Spiralen sind dabei aufeinanderfolgende Fibonacci-Folgenglieder.
- Der goldene Winkel teilt einen Kreis im Verhältnis des goldenen Schnitts, er beträgt ca. 137,5°. Die Abweichung vom mathematischen Goldenen Winkel beträgt bei den Sonnenblumenkernen weniger als 0,01 %.





FIBONACCI - FOLGE IN DER NATUR

- Die Abbildung zeigt eine Pflanze, die nach den Gesetzmässigkeiten der Fibonoccifolge ihre Seitentriebe bildet.
- In der ersten Phase des Wachstums eines Triebs werden keine Seitentriebe gebildet, in der zweiten und in allen folgenden Phasen wird jeweils ein Seitentrieb mit Blatt angelegt. Auch die Anzahl der gebildeten Blätter und Blüten sind Fibonaccizahlen.
- Die Schafgarbe Achillea ptarmica wächst in etwa gleich wie die beschriebene Pflanze.



FIBONACCI FREEZE

Write a program to calculate the Fibonacci Numbers.

DIE AUFGABE

- The input to your program would be a sequence of numbers smaller or equal than 5000, each on a separate line, specifying which Fibonacci number to calculate.
- Your program should output the Fibonacci number for each input value, one per line.
- Time limit: 3 seconds

DIE 5000. FIBONACCI ZAHL

3878968454388325633701916308325905312082127714646245106160597214895550139044037097010822916462210669479293452858882973813483102008954982940361430156911478938364216563944106910214505634133706558656238254656700712525929903854933813928836378347518908762970712033337052923107693008518093849801803847813996748881765554653788291644268912980384613778969021502293082475666346224923071883324803280375039130352903304505842701147635242270210934637699104006714174883298422891491273104054328753298044273676822977244987749874555691907703880637046832794811358973739993110106219308149018570815397854379195305617510761053075688783766033667355445258844886241619210553457493675897849027988234351023599844663934853256411952221859563060475364645470760330902420806382584929156452876291575759142343809142302917491088984155209854432486594079793571316841692868039545309545388698114665082066862897420639323438488465240988742395873801976993820317174208932265468879364002630797780058759129671389634214252579116872755600360311370547754724604639987588046985178408674382863125

1.045 Stellen

REKURSIVE LÖSUNG

```
private static int fibonacci (int a) {
        if (a == 1 || a == 2)
                return 1;
        else
                return fibonacci (a-1) + fibonacci (a-2);
}
```

→ Sehr zeitaufwändig

IDEE

- $f_{2n} = (f_n)^2 + 2 * f_n * f_{n-1}$
- Mit dieser Formel lässt sich f_{2n} berechnen, ohne f_{n+1} bis f_{2n-2} zu kalkulieren. Wenn nun noch f_{2n-1} bekannt wäre, könnte f_{4n} berechnet werden usw.
- $f_{2n-1} = (f_n)^2 + (f_{n-1})^2$

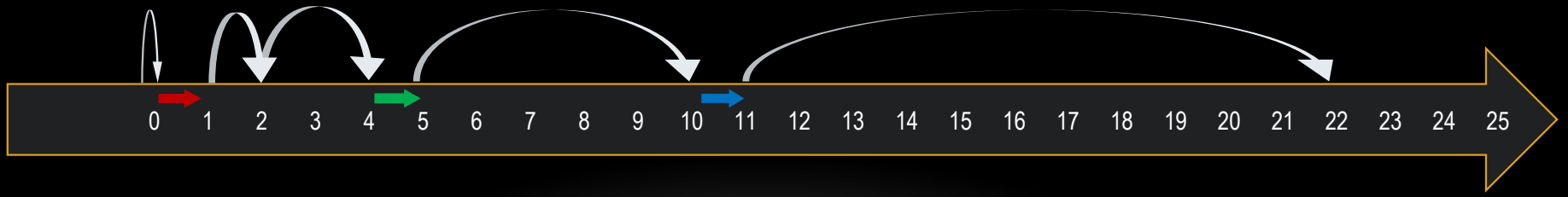
IDEE

- Damit lassen sich f_{2n} für große n erheblich schneller berechnen (falls mit f_2 begonnen wird; allgemein: $f_{(k * 2^n)}$).
- Allerdings sollen auch Fibonacci-Zahlen zwischen Zweierpotenzen berechnet werden können.
- Um Zwischenwerte erreichen zu können, berechnet man nun nach einer Verdoppelung gegebenenfalls noch den Nachfolger nach der herkömmlichen Berechnungsmethode.
- $F_{2(2(2(1+x1)+x2)+x3)....}$ x1 steht hierbei für eine Anwendung der herkömmlichen Methode, vor der ersten Verdoppelung usw.
  → Man nimmt n in der Binärschreibweise und geht es Bit für Bit durch. Bei jeder 1 wird die oben genannte Operation durchgeführt. x1 stellt das “höchste Bit” dar.

VISUALISIERUNG DER FUNKTIONSWEISE

$n = 22_{10} = 1\ 0\ 1\ 1\ 0_{2}$

1. $n_1 = (2 * 0) + 1 = 1$
2. $n_2 = (2 * n_1) + 0 = (2 * 1) + 0 = 2$
3. $n_3 = (2 * n_2) + 1 = (2 * 2) + 1 = 5$
4. $n_4 = (2 * n_3) + 1 = (2 * 5) + 1 = 11$
5. $n_5 = (2 * n_4) + 0 = (2 * 11) + 0 = 22 = n$



LÖSUNG NACH PROF. O. FORSTER 1/2

```
private final static BigInteger TWO = BigInteger.valueOf (2);

public static BigInteger fib (int value) {
        BigInteger n = BigInteger.valueOf (value);
        if (n.compareTo (BigInteger.ONE) <= 0) {
        return n;
        }

        BigInteger x = BigInteger.ONE;
        BigInteger y = BigInteger.ZERO;
```

Erstellung einer Konstante mit dem Wert 2

Konvertierung von *int* in *BigInteger*

Bei 0 oder 1 soll 0 oder 1 zurückgegeben werden

x und y mit BigInteger 1 und 0 vorbesetzen

LÖSUNG NACH PROF. O. FORSTER 2/2

```
        for (int k = n.bitLength() - 1; k > 0; k--) {
                BigInteger powered = x.pow(2);
                x = powered.add (TWO.multiply(x).multiply(y));
                y = powered.add (y.pow(2));
                if (n.testBit(k)) {
                        BigInteger temp = x;
                        x = x.add(y);
                        y = temp;
                }
        }
        return x;
}
```

Schleife läuft „Länge der Binärzahl -1“ mal durch

$f_{2n} = f_n^2 + (2 * f_n * f_{n-1})$

$f_{2n-1} = f_n^2 + f_{n-1}^2$

Wenn Bit an Stelle k gleich 1, dann
x sichern,
x = x+y,
y = Sicherung von x