5... の豆旅

5.1 黄金比

さあ、これから 5... の豆旅が始まる。と言っても 5 から豆旅が始まるのではなく、豆旅の途中に 5 と出会うのがいままでとは違うところだけど。

まず、フィボナッチ数列の 2 項の比

$$\frac{1}{1}, \frac{2}{1}, \frac{3}{2}, \frac{5}{3}, \frac{8}{5}, \frac{13}{8}, \frac{21}{13}, \frac{34}{21}, \cdots$$

に再び登場してもらおう。

プログラム `[Fratio.java]` で計算したから分かっているだろうが、比の値は 1.618033... となる。ところでフィボナッチ数列の 2 項の比は、(後の項)/(前の項) で求めたものだ。しかし、比なんだから別に (前の項)/(後の項) で求めたっていいだろうに。

EX. フィボナッチ数列の比を (前の項)/(後の項) で計算してみよ。電卓でも計算できるが、だいぶ以前に作った `[Fratio.java]` の 8:行目を変更すればすぐだ。

さあ、比の値はどうなった？ ちょっと面白い結果になったのではないかい？ 比の値は 0.618033... である。きっちり 1 だけ違っているだろう。誤差はないのかって？ そう、まったく誤差はないのだ。この面白い性質を持つ比の値は**黄金比**と呼ばれている。一般には、1.618033... の方を黄金比と呼ぶことが多い。

黄金比とは一体どういう数なのだろうか。計算は簡単にできるので確かめてみよう。

まず、1.618033... に収束する真の値を x としよう。あとから求めた比 0.618033... は、x の逆比 $1/x$ のことである。それが x より 1 小さい値に等しいのだから

$$\frac{1}{x} = x - 1 \tag{1}$$

が成り立つ。(1) は簡単な 2 次方程式となるが。両辺に x を掛けて整理すると $x^2 - x - 1 = 0$ だか

ら、解の公式を使えば一発で解ける。フィボナッチ数の比は正の値なので、$x > 0$ の解を求めると

$$x = \frac{1+\sqrt{5}}{2}$$

であることが分かる。黄金比は**無理数** $\sqrt{5}$ を含む数なのだ。ここに 5 が登場してきた。解は正の数だけに限っているものの、方程式から一つの解しか得られないということは、(1) の性質を満たす数は黄金比だけということである。

Java にシミュレートさせてみよう。ここでは (1) を移項して

$$x = \frac{1}{x} + 1 \tag{2}$$

で x を探ることにする。その方が x の値を直接計算できるからだ。ところで、探るという言葉が気になるね。なぜそのような言葉遣いをしたかはプログラムを見てもらいたい。

programming list [GRsimula.java]

```
 1: import java.util.Scanner;
 2:
 3: public class GRsimula {
 4:
 5:     public static void main(String[] args) {
 6:         Scanner s = new Scanner(System.in);
 7:
 8:         System.out.print("input an initial value: ");
 9:         double x = s.nextInt();
10:
11:         for(int i = 1; i < 20; i++) {
12:             x = 1 / x + 1;
13:             System.out.println(x);
14:         }
15:     }
16: }
```

いつもと変わらぬ見慣れた景色の 9:行目までは素通りしよう。

プログラムは実に簡単だ。9:行目で取り込んだ変数 `x` を、11:行目からの `for` 構文で繰り返すだけだ。`x` は実数値を想定している。`for` 構文は 20 回ほどの繰り返しだが、このぐらいで十分収束してくれる。繰り返す計算は 12:行目の `1 / x + 1` の計算で、次に再利用できるよう再び `x` へ代入している。

その経過を表示しているのが `for` 構文に組み込まれた `System.out.println;` 文である。これによって 19 回先までの `x` の値が表示される。

TRY!　入力要求に対して、色々な浮動小数点数を入力してみよ。初期値が何であれ、黄金比に収束する様子が分かるだろう。

あれ？　**Java** が 2 次方程式を解くんじゃないんだ、と感じただろう。そう、コンピュータは計算をする機械であって、問題を解く機械ではない。問題を解く手順は人間が与えるのだ。では、この手順は何をしているのだろう。どう見ても (2) と同等な—つまり両辺に x を掛けた—方程式

$$x^2 = 1 + x$$

を解いているのではないね。でも、解である黄金比が求められている。

それなら $x^2 = 1 + x$ を移項した式、$x = x^2 - 1$ を使っても同じことだろう。そこでプログラムの 12:行目を `x = x * x - 1;` に変えてみる。さあ、準備は整った。プログラムを走らせよう。

TRY!　入力要求に対して、いろいろな浮動小数点数を入力してみよ。初期値が何であれ黄金比に収束する、とは言えないはずだ。

どうして？　同じ関係の方程式を使ったはずなのに。その秘密はグラフを描けば見えてくる。そして、プログラムがしていることを追ってみれば分かるのだけれど、いまはそこに足を踏み入れないでおこう。