

# 1...の旅

## 1.1 フィボナッチ数列

数列とは、数が何らかの規則で並んでいるものを言う。

2, 4, 6, 8, 10, ...

は偶数の数列だが、偶数列を作る規則は“前の項に 2 を加える”というものである。

さて、ここに 1, 1 から始まる数列がある。次の項を作る規則は“直前の 2 項の和”である。すなわち 1, 1 の次に来る項は 2 となり、数列は 1, 1, 2 と延びる。規則を繰り返し当てはめれば、次の項は 3 で数列は 1, 1, 2, 3 と延び、さらに次の項は 5 である。これが延々となされ

1, 1, 2, 3, 5, 8, 13, 21, 34, ... (1.1)

なる数列が出来上がる。このようにしてできる数列をフィボナッチ<sup>1</sup>数列と呼ぶ。

フィボナッチ数列をマクロで再現してみよう。

---

programming list [Fibonacci.vba]

```
1: Sub Fibonacci()  
2: Const SUP As Integer = 20  
3: Dim f0, f1, i As Integer  
4:  
5:     f0 = 1: f1 = 1  
6:     For i = 3 To SUP  
7:         Sheet1.Cells(i, 1) = f1 + f0  
8:         f1 = f1 + f0  
9:         f0 = f1 - f0  
10:    Next i  
11: End Sub
```

---

始めに気が付くのは 2: 行目の Const SUP As Integer = 20 だろう。Const は定数を表す。よって、SUP は 20 という定数値を与えられたということだ。すると以後 SUP が出るたびに、それは 20 に置き換わる。今は 20 項までの計算をさせたいため、あらかじめ SUP を 20 にしている

---

<sup>1</sup>フィボナッチ (1174?-1250?): イタリア、ピサのレオナルドの通称。

わけだ。こうする利点は、後で 100 項まで計算させたくないとき、この行の変更だけで済ませることができる。このような短いマクロでは恩恵が分かりにくい、あちこちに一定の値が使われるマクロなら、Const で定義しておくといいたい。SUP を大文字で書いたのは定数を目立たせるため、他の変数のように小文字でも構わない。

次に 3: 行目で、フィボナッチ数列の計算に必要な 2 つの変数  $f_0, f_1$  を用意した。数列は 1, 1 の項から始まっているので、5: 行目で 1 を代入している。変数はどこかで初期値に当たる数を代入—すなわち初期化—する必要がある。もし、ここで初期化を怠ると、7: 行目でなされる  $f_1 + f_0$  の結果は 0 になってしまう。VBA では、変数を宣言すると 0 で初期化されるからだ。また、5: 行目は 2 つの文を 1 行にまとめていることに注意してほしい。VBA では文の区切りは : であることになっている。

6: 行目からの For-Next 構文は  $i = 3$  から始めて、SUP—今は 20 で定義されている—まで 1 ずつ増加させている。 $i = 3$  から始めた訳は、 $f_1 + f_0$  で与えられる項がすでに第 3 項になるからである。

7: 行目は  $f_1 + f_0$  の計算された値を ( $i$  行 1 列) のセルに表示する。このように計算式のまま代入することも可能である。

8:, 9: 行目はかなりトリッキーなことをしている。フィボナッチ数は常に  $f_1 + f_0$  で計算するので、ここの  $f_1 + f_0$  の値が次は  $f_1$  となり、今まで  $f_1$  だった値が  $f_0$  にならなくてはならない。だったら、 $f_1 = f_1 + f_0$ 、 $f_0 = f_1$  でいいんじゃない?と思うだろう。残念だがそれは違う。ならば代入の順番を変えて、 $f_0 = f_1$ 、 $f_1 = f_1 + f_0$  としたらどうだろう。だが、これもだめなのだ。理由は 8: 行目で  $f_1$  の値が変化してしまうからだ。

そこでトリッキーな解決をしている。8: 行目が終わったとき  $f_1$  には  $f_1 + f_0$  が代入される。これは予定通りだ。次に  $f_0$  には“古い”  $f_1$  の値がほしいのだが、この時点での  $f_1$  は“新しい”ものになってしまった。しかし安心してほしい。“新しい”  $f_1$  は“古い”  $f_1$  を含んでいる。なぜなら  $f_1 = f_1 + f_0$  だからだ。 $f_0$  がなければ“古い”  $f_1$  を取り出せる。そこで 9: 行目の式となる。見事に  $f_0$  に“古い”  $f_1$  の値が入っただろう。

このマクロでは、あまり先の項まで調べることは無理だが、ある程度のところまでのフィボナッチ数列を感じることができるはずだ。Const SUP As Integer = 20 の値を変更して、もう少し先のフィボナッチ数を求めてもらいたい。

さあ、この地まで旅をしてきてだいぶ見通しが良くなってきたことだろう。大過なく旅が続けられているだろうか。しかし、マクロはちょっとした気のゆるみで、とんでもなくおかしい動作をすることがある。今 For-Next 構文を上限を決めて実行できるようになったのだが、上限の値を不正なものにするとマクロが暴走する危険だってある。だから、上限に設定する値は常に確認すること

が重要だ。そしてもっと重要なのは、マクロが暴走したり無限ループに入り込んだとき、それを止める方法を知っていることである。