

7...の旅

7.1 不思議な循環小数

$\frac{1}{7} = 0.142857 \dots$ は不思議な性質を持っている。正確には、循環節 142857 の性質が面白いのだ。それは

$$142857 \times 2 = 285714$$

$$142857 \times 3 = 428571$$

$$142857 \times 4 = 571428$$

$$142857 \times 5 = 714285$$

$$142857 \times 6 = 857142$$

となるからだ。なんと、循環節の数字が巡回しているではないか。

こうなってくると、他にも不思議な循環節を持つ分数を調べたくなるだろう。以前、循環節を調べたことを覚えているかい？ 旅を始めたばかりの頃は、 $\frac{1}{113}$ がどの程度の循環節を持つか調べるのに苦労したね。余りをザーッと表示させて、最初の余りと同じものがあるかを調べたんだ。これは手間のかかる作業だし、第一、商がどうなっているのか分からずじまいだった。当時は、循環節を表示する力量が不足していたので、仕方なく余りを表示することで目的を達していたのだ。ところが、今や力量はかなり上昇している。循環小数の循環節を見せてくれるマクロを組んでみよう。

対象とする分数は $\frac{1}{n}$ の形で十分だが、 n の値によっては困ることがある。循環しないで割り切れてしまう場合があるからだ。割り切れてしまう分数は、必ず $\frac{1}{2^r 5^s}$ の形をしている。従って、 n には 2 と 5 が含まれていないことが条件だ。ただ、この条件の分数だけを対象とすると、 $\frac{1}{6}$ のような循環小数も除外されてしまう。しかし $\frac{1}{6}$ は、 $\frac{1}{3}$ の循環小数を $\frac{1}{2}$ で割り切っていると見れば、本質は $\frac{1}{3}$ と同じだ。よって 2 や 5 がひとつでも含まれている分数は、対象から外して構わないだろう。そのほうがマクロも簡単にできる。

programming list [CycleDec.vba]

```

1: Sub CycleDec()
2: Dim b, r, i, col As Integer
3:
4:     b = Sheet1.Cells(1, 1).Value
5:
6:     col = 0
7:     If (b Mod 2 = 0) Or (b Mod 5 = 0) Then
8:         Sheet1.Cells(1, 2) = "無効"
9:     Else
10:        r = 1
11:        For i = 1 To cycle(b)
12:            col = col + 1: Sheet1.Cells(2, col) = (r * 10) \ b
13:            r = (r * 10) Mod b
14:        Next i
15:
16:        col = col + 1: Sheet1.Cells(2, col) = "..."
17:    End If
18: End Sub
19:
20: Function cycle(bb)
21: Dim r, cyc As Integer
22:
23:     r = 1: cyc = 0
24:     Do
25:         r = (r * 10) Mod bb
26:         cyc = cyc + 1
27:     Loop While r <> 1
28:
29:     cycle = cyc
30: End Function

```

マクロは、 $\frac{1}{7}$ なら A2 セルから右へ順に “1 4 2 8 5 7 ...” と表示することにした。... 以下が循環するという意味である。もし、小数を表示していることを明確にしたければ、A2 セルには “0.” と表示し B2 セルから右へ順に “1 4 2 8 5 7 ...” と表示するようにすればよいだろう。

以前のマクロ [Recurdec.vba] では、 $\frac{1}{n}$ の余りを $(n-1)$ 回表示させた。これだと $\frac{1}{9}$ のような分数では、最初の余りから 8 回目の余りまで 1 が表示されてしまう。今回は商の表示を目的としているので、 $\frac{1}{9}$ は何も “1 1 1 1 1 1 1 1 ...” でなく “1 ...” となれば十分だ。それに、余分な循環節を表示するようでは、 $\frac{1}{113}$ が本当に 112 の循環節を持つ小数かどうかは、綿密に調べなくてはならず効率が悪い。

そのために、循環節がいくつか調べる関数 cycle(bb) が必要になったのだ。先に、そこから説明しよう。

20: 行目の cycle 関数は $\frac{1}{n}$ の分母を変数 bb で受け取る。このとき最低でも 1 回の割り算が実

行されるはずだから、Do-Loop 構文を用いている。

Do-Loop 構文と While-Wend 構文が明確に違うのは次の点だ。Do-Loop 構文は、まず Do から Loop までの処理をしてから While の判定をするので、一度は必ず Do-Loop 構文内の処理をする。それに対して While-Wend 構文は、先に While の判定がされるので、場合によっては While-Wend 構文内の処理をまったくしないことがある。特に使い分けなくてもマクロは組めるので、それほど神経質になることはないだろう。

23: 行目の r はこの時点では分子の 1 を表すが、順次余りが代入される変数である。また、変数 cyc は循環の回数を記憶するために必要で、最初は 0 で初期化しておかなくてはならない。

24: ~ 27: 行目にかけての処理は、[RecurDec.vba] で使った手法と同じである。違いは 26: 行目で循環の回数をカウントしていることと Do-Loop 構文を使っている点だ。小数が循環することは、余り r が 1 になることで判断する。なぜなら始めの分数は分子が 1 だからだ。もし $r = 1$ になったら、27: 行目の While 文がそれを判断し、29: 行目で循環の回数として値が返される。

ところで 1: 行目からの CycleDec() プロシージャにおいては、見ての通り分母として入力される b 、余りを順次代入する変数 r 、それにカウント変数の i が用意されている。 r は cycle 関数にも使われているけど大丈夫？ 大丈夫である。それは cycle 関数と CycleDec() プロシージャは独立しているからだ。そのため、同じ変数名を使っても問題はない。

また 4: 行目で、分母にあたる b の値を A1 セルから読み取るが、冒頭に述べたように、分母に 2 または 5 を因数に持つ数は除外したい。そこで、If 文によって入力を制限している。ここで使われている記号 Or は論理演算子の仲間である。 $A Or B$ は、 A か B のどちらかが真であれば、 $A Or B$ は真になる。少なくとも一方が真であれば、マクロは数値が無効であることを知らせるために、8: 行目を実行して終了する。 $A Or B$ が偽と判断されるためには、 A と B の両方が偽であることが必要だ。この場合に限りプログラムは b の値を Else 以下の処理へ回すのだ。

さあ、無事に b の値を手に入れたら割り算の開始だ。

10: 行目の 1 は分子の 1 だが、計算が進むたびに分子の値が変わるので、変数 r を用いて 1 を代入したのだ。

12: 行目が商の表示、13: 行目が次回使う余りになることは分かるね。これも [RecurDec.vba] と同様だ。

最後に 16: 行目で ... 付けて体裁を整えたら完了だ。これで安心して循環節が分かるようになった。遠慮せずに $\frac{1}{n}$ を計算させよう。