0.3 有理数への継承

さて、整数を使えばどんな大きな数でも、またどんな小さな負の数でも表すことができる。ただし、それは数学の中での話であって、コンピュータでは数の大きさに制限がついてまわるが。ところで整数だけでは端数を表すことができない。端数を表すために私たちは**分数**や**小数**を使っているのだ。身近に分数や小数を使っていると、それらの数は自然発生的に生まれたように感じてしまうものだ。しかし、分数や小数は私たち人間が作り出した数なのである。

分数や小数はどちらも "数" と名乗っているものの、性格には大きな違いがある。今では 1/8 と 0.125 は、どちらも同じ量を表す数の扱いを受けている。けれども 1/8 は、8 つの量に対する 1 の量という比を意味し $1:8$ と表す。すなわち $1/8 = 1:8$ だ。一方 0.125 は、1 ほどに大きくない量の 0.1 に、0.1 ほどに大きくない 0.02 を加え、さらに 0.01 ほどに大きくない 0.005 を加え合わせた量である。どっちも同じじゃないかって？　そう、量という点ではね。さらに付け加えるなら、どちらも実数を表すことに関して "不完全" であることだ。おいおい話すことになるが、分数も小数も所詮人間が作ったものだから、完璧には程遠いのである。

結局のところ、分数でも小数でも細かい量—これは精密な量と言っても構わない—を表せることに変わりがない。例えば 10/3 や 3 という値は、**円周率**を大まかに表している。もちろん円周率というのは、直径に対する円周の比のことだ。でも、これよりも 22/7 や 3.14 とすることで、さらに 355/113 や 3.141592 とすることで、円周率をより精密に表せる。ここで注目してもらいたいのが、分数の作り方である。分数はあくまでも自然数の比で表している。一方、小数は数字を増やすことで対処しているが、これは小数に小数を足していく操作になっている。つまり、分数が自然数を継承して作られる数であるのに対し、小数は自分自身を継承している点に注意してほしいのだ。ここに分数と小数の性格の違いを見ることができるわけだ。

今日では、私たちは $a:b$ の比は a/b という分数で表すのが一般的になっている。このように整数 a, b（ただし $b \neq 0$）を用いて a/b で表すことができるすべての数を**有理数**と呼ぶことにする。有理数の呼称は慣れないと戸惑うかもしれない。a/b で表される数は rational numbers（比の数）と呼ばれるが、日本語訳が有理数となっているだけのことなのだ。すると、この約束に従えば -5 は $-5/1$、0 は $0/1$ と表すことができる数なので、整数も有理数であると言えるのである。

先ほど、分数と小数は同じ量を表す点で同じであると言った。しかし、分数の集合と小数の集合は同じ濃度ではない。このことは後で示すが、その前にコンピュータでも小数を扱えるようにしておきたい。

プログラム `[parrot2.cpp]` によって、コンピュータで数値入力が繰り返し行えるようになった。しかし変数 `n` は整数が前提だったので、このプログラムにうっかり小数を入力すると期待を裏切られる。このプログラムは変数 `n` を整数でのみ受け付け、整数のみで出力するようにできているから

だ。そのため、小数を扱いたいのならプログラムに若干の修正をしなくてはならない。

programming list [float.cpp]

```
#include <iostream>

int main() {
    float n;
    while(1) {
        std::cout << " input N : "; std::cin >> n;
        std::cout << "output N : " << n << std::endl;
    }

    return 0;
}
```

コンピュータで扱う小数は**浮動小数点数**と呼ばれる。今は詳しい説明を省くが、早い話、コンピュータでも小数の計算はできるということだ。そのためのキーワードには `float` を使うとよいだろう。[parrot2.cpp] との違いはこの 1 点だけである。だから、プログラムに関して話すことはないもない。でも、“名前” に関する話を少しだけ付け加えよう。

前に、`cout` が characters output の略であると言ったが、多くのプログラム言語で使われる命令の名前はそれなりの意味を持つものだ。と同時に、プログラマーが使う変数の名前も、習慣として大体似たようなものとなる。例えば、変数名は `a` でも `x` でも `variableX` でも好きにすればよいのだが、`n` が好まれるのは数（numbers）を意識してのことだろう。私もそれなりの名付けをしているので、しっくりこなければ各自好みの名前でコードを書けばよい。

TRY! “ input N : ” の入力要求に対して、いろいろな小数や整数を入力してみよ。どのように表示されるか？