6.2 友愛数・婚約数

完全数には親戚のようなものが存在する。220 の自分自身を除く約数の和は

$$1+2+4+5+10+11+20+22+44+55+110=284$$

である。もちろん完全数ではない。ところが、284 についても同じことをすると

$$1+2+4+71+142=220$$

である。なんと、互いに完全数の性質を補完し合っているみたいではないか。このようなペアを友愛数と呼んでいる。ここでは友愛数のペアを求めてみよう。

```
\newcount\n \newcount\d \newcount\m \newcount\sum
\newcommand\addmeasures{%
  \d=1 \sum=0
  \loop
    \n=\m \divide\n\d \multiply\n\d \advance\n-\m
    \ifnum\n=0 \advance\sum\d \fi
  \advance\d1 \ifnum\d<\m \repeat
}
\makeatletter
\newcommand\fnfind[1]{%
  \newcount\t
  \m=6
  \@whilenum\m<#1 \do{%
    \t=\m \addmeasures \m=\sum \addmeasures
    \ifnum\sum=\t \ifnum\t<\m (\number\t, \number\m) \fi\fi
    \m=\t \advance\m1}%
}\makeatother
```

これで、『3000 未満の友愛数は`\fnfind{3000}`である。』と書けば『3000 未満の友愛数は (220, 284) (1184, 1210) (2620, 2924) である。』が出力される。それも、かなり長い時間をかけてだ。

約数の和を計算する`\addmeasures` 関数は、完全数を求めるときに使ったものと同じである。マクロ`\fnfind` は、与えられた引数までにある友愛数のペアを求めるようになっている。実際、3000 未満の友愛数を求めると 3 組あった。(220, 284) は古くから知られたもので、(2620, 2924) をはじめとしてオイラーは 60 組以上の友愛数を見つけたようである。ところが、(1184, 1210) のペアを見逃していたというのだから不思議だ。

さて、マクロは 6 から始めているけれど、この初期値は単に完全数を見つけるマクロの名残である。すべきことは、`\m` の約数の和`\sum` を求めたら、もう一度`\sum` の値から約数の和をもめることである。それがもとの`\m` に等しいかどうかで友愛数を判定している。`\ifnum` 命令が 2 重になっ

ているのは、完全数や同じ組の友愛数を出力しないためにある。つまり、完全数の約数の合計はもとの数になるから—それが完全数の定義だから—広い意味で完全数も友愛数である。しかし、わざわざそれを出力することはないだろう。プリンタトナーの無駄遣いだし。また、n の友愛数 m を見つけたら、同時に m の友愛数は n なので、同じペアが重ねて出力されてしまう。これらを除くために`\ifnum\t<\m` を条件に加えている。

友愛数に似たペアに婚約数がある。完全数も友愛数も約数の合計に 1 は加えるが自分自身を加えない。そもそも 1 と自分自身はどんな数でも持っている自明の約数だから、そのうち 1 は加えて自分自身を加えないのはバランスを欠いているのではないだろうか。そこで、約数の合計に 1 を加えないとどうなるか、と考えるのは自然な流れだ。ここでは 1 と自分自身を含まない約数の和を、真の約数の和と表現しておこう。

これを調べるマクロは、`\addmeasures` 関数において`\d=2` とするだけである。

```
\newcount\n \newcount\d \newcount\m \newcount\sum
\newcommand\addmeasuresB{%
  \d=2 \sum=0
  \loop
    \n=\m \divide\n\d \multiply\n\d \advance\n-\m
    \ifnum\n=0 \advance\sum\d \fi
  \advance\d1 \ifnum\d<\m \repeat
}
\makeatletter
\newcommand\fnfind[1]{%
  \newcount\t
  \m=6
  \@whilenum\m<#1 \do{%
    \t=\m \addmeasuresB \m=\sum \addmeasuresB
    \ifnum\sum=\t \ifnum\t<\m (\number\t, \number\m) \fi\fi
    \m=\t \advance\m1}%
}\makeatother
```

これで、『3000 未満の婚約数は`\fnfindB{3000}`である。』と書けば『3000 未満の婚約数は (48, 75) (140, 195) (1050, 1925) (1575, 1648) (2024, 2295) である。』が出力される。もちろん、友愛数と同等の時間がかかる。

3000 未満の婚約数を調べてみると、友愛数とはおもむきの違うペアが現れる。友愛数のペアと婚約数のペアを見比べて気づくことはないだろうか。そう、友愛数はすべて偶数のペアで、婚約数はすべて奇数と偶数のペアになっていることだ。これは非常に不思議なことで、これまでに見つかった友愛数と婚約数の組み合わせは、すべてそうなのだ。友愛数とか婚約数の名称がどうやってついたかは知らない。ただ、古代ギリシアの数字感に、1 は神、2（偶数）は女、3（奇数）は男、5（$2+3$）は結婚、...というものがある。それに照らすと、偶数どうしのペアが友愛数、奇数と

偶数のペアが婚約数という名称は的を射ている。

さらにこの考えを進めると、a の真の約数の和 $\to b$、b の真の約数の和 $\to c$、c の真の約数の和 $\to a$ のように 3 数で巡回する組や、4 数で巡回する組など、いくらでも拡張することができる。有名なところでは $12496 \to 14288 \to 15472 \to 14536 \to 14264$ という 5 数の巡回がある。このような数は社交数と呼ばれる。社交数には 5 数以外の社交の輪がいくつも見つかっているが、不思議なことに 3 数の社交の輪は見つかっていないようだ。三角関係は成り立たないのだろうか。

ただし、社交数の組を見つけるマクロを書くのはちょっと苦労する。というのは、ある数の真の約数を調べるのは簡単なことでも、何組でもとに戻るかが最初に分かってないので、比較を何回すればよいか未定であることだ。つまり、数 a の巡回を 20 回やって最初の約数の和に一致しないからといって、a が社交の輪にないとは言い切れないのである。実際に、28 数で巡回するものがあるというから一筋縄ではいかない。まあ、100 数以内で見つからなければあきらめる、というように決めていれば、さほど難しくはない。`\addmeasuresB` を繰り返す構造にすればよいからだ。しかし問題は別のところにある。無駄が多い約数の和を求める関数を、どれだけ効率よく動作するようにできるかだ。暇つぶしにはちょうどよいかもしれないが、あとで効率を上げたマクロを示すことにしよう。

暇つぶしのついでに、この道すがらのマクロが利用できそうな話題を提供してみよう。完全数はとても少なく、ほとんどの数は完全数にならない。自分自身を除く約数の和が自分自身になる数はよほど恵まれている。では、一般の数はどのような状況にあるのだろうか。たとえば 10 の約数については $1+2+5=8<10$ が言える。また、12 の約数については $1+2+3+4+6=16>12$ が言える。このように完全数でない数は、10 のように約数の和が自分自身に満たない数と、12 のように約数の和が自分自身を超える数に分かれる。自分自身に満たない数は**不足数**、自分自身を超える数は**過剰数**と呼ばれる。

どんな数が不足数や過剰数になるのだろう。そして、不足数や過剰数の割合や性質はどうなっているのだろう。ここまでの知識があれば、完全数以外のすべての数について不足数か過剰数を示すマクロが書けるはずだ。なかなか大変かもしれないが、勇気があればがんばってみよう。