

4.3 双子素数

素数の話題は尽きることがない。話題の一つに双子素数というものがある。2を除けば素数は奇数であるが、ときどき 17, 19 や 41, 43 のように連続して素数が現れることがある。これらを双子素数と呼んでいる。

素数が無数にあることは証明されている。それはユークリッド¹⁾による証明が有名である。証明の大筋は次のようなものだ。

素数が有限個しかないと仮定し、素数を $2, 3, 5, \dots, P$ とする。このとき $2 \cdot 3 \cdot 5 \cdots P + 1$ なる数を作れば、これは P より大きい新たな素数となるから、素数が P までしかないことに矛盾する。この矛盾は素数が有限個であると仮定したことに起因している。すなわち素数は有限個ではない。

このように、ある仮定から始めて矛盾を引き出し、結果、仮定が間違っていると結論する証明方法は背理法と呼ばれる。

ところで、素数がどれぐらいの頻度で現れるかとか、双子素数も無数にあるのかなど、素数には未解決の問題が多い。素数の頻度というのも変な感じであるが、ガウス²⁾は素数が現れる頻度について、次のような見解を持っていた。

$$1 \text{ から } N \text{ までには、素数はおよそ } \frac{N}{\log_e N} \text{ 個含まれる}$$

たとえば $N = 100$ とすると、

$$\frac{100}{\log_e 100} \approx 21.7$$

だが、実際は 25 個の素数がある。ぴったりではないが、まあ近いところをついている。式は N が大きいほど、より正しい頻度を教えてくれる。もし N を与えたとき、本当にぴたり正確な素数の数を求める式が発見されれば、画期的なことであるが、いまのところそのような式は発見されていない。またガウスの式とは別の、素数の頻度を計算する式はあるが、ガウスの式の美しさにはかなわないのだ。何とも不思議なことである。

双子があれば三つ子も考えたくなるのが人情で、3, 5, 7 のような三つ子素数がどれくらいあるかと考えてしまう。だが結論を言おう。これ以外の三つ子素数はない。連続する三つの奇数は三つ子

1) ユークリッド (330?B.C.-275?B.C.): ギリシアの数学者。

2) カール・フリードリヒ・ガウス (1777-1855): ドイツの数学者。

素数にはならないのだ。少し考えれば理由は分かるだろう。

三つ子素数がないと分かったところで、双子素数に集中しよう。と言っても場当たりの双子素数を探すのでは効率が悪い。徹底した調査で、どのような条件のとき双子素数が現れるのかが分かれば、スクリプトも書きやすくなる。そこで、ここでは次の単純な事実をもとに、双子素数を見つけていこう。

2, 3 を除くすべての素数は $6m \pm 1$ の形をしている

そう、素数は決して $6m \pm 2$ や $6m \pm 3$ のような形をしていないのだ。誤解しないでもらいたいことがある。私は「素数は $6m \pm 1$ の形をしている」と主張しているのであって、決して「 $6m \pm 1$ の形をしている数は素数である」と言っているのではない。逆は必ずしも正しくないとは、正にこのことを言う。そして、ここでも少し考えれば、すべての素数が $6m \pm 1$ の形をしている理由も分かると思う。要するに $6m \pm 2$ 、 $6m \pm 3$ 、 $6m \pm 4$ は 2 や 3 で割れるので素数であるわけがない。残る $6m \pm 5$ は

$$6m \pm 5 = 6m \pm (6 - 1) = 6m \pm 6 \mp 1 = 6(m \pm 1) \mp 1$$

だから、 $6m' \pm 1$ のことである。つまり、素数は必ず $6m \pm 1$ で表される数でなくてはならない。

さて、そういうことなら話は簡単だ。 $6m - 1$ が素数であるかどうか調べ、素数であるときに限り $6m + 1$ が素数であるかどうか調べればよい。運良く二つとも素数なら、それが双子素数ということだ。スクリプトは二つの関数に分けてみた。一つは素数をチェックする関数で『

```
\begin{luacode*}
function pchk(n)
  i = 3
  while i <= math.sqrt(n) do
    if n % i == 0 then
      return false
    end
    i = i + 2
  end
  return true
end
\end{luacode*}
```

』のようにしてみた。以前のスクリプトより効率化してある。

もう一つは双子素数を出力するための関数で『

```

\begin{luacode*}
function twinp(n)
  m = 1
  while 6 * m < n do
    if pchk(6 * m - 1) and pchk(6 * m + 1) then
      tex.print('(' .. 6 * m - 1 .. ', ' .. 6 * m + 1 .. ')')
    end
    m = m + 1
  end
end
\end{luacode*}

```

』のようにした。これで『`\directlua{twinp(100)}`』と書いて『(5, 7) (11, 13) (17, 19) (29, 31) (41, 43) (59, 61) (71, 73)』が出力される。

スクリプトは、まず素数判定用の `pchk` 関数を定義している。`pchk` 関数が素数を見つける方法について話しておこう。

たとえば 49 が素数かどうか判定するには、2 から 48 までの数で割ってみればよい。何一つ割ることができなければ 49 は素数だ。しかし考えてみてほしい。素数は 2 を除いてすべて奇数である。奇数は偶数で割れっこない。したがって偶数での割り算を試す必要はないのだ。その結果、49 が素数かどうか調べるには、3 から 47 までの奇数で割ってみるだけでよい。これで計算量を半分に減らせる。

さらに、 n が素数かどうか調べるのに、 $n - 1$ までの数で割る必要などない。結論を言えば \sqrt{n} までの数で割るだけで十分である。たとえば 51 は 3 で割れる—商は 17 である—から、同時に 17 でも割れてしまう。数 n が p で割れるなら、同時に $\frac{n}{p}$ でも割れているのだ。このことから n が素数かどうか調べるのに、 \sqrt{n} までの数で割ればよいことが結論できる。以上の考察によって、素数を判定するアルゴリズムは次のようになる。

- a) 奇数 n だけを対象に素数かどうかを調べる
- b) 候補となる n を 3 から \sqrt{n} までの奇数で割る
- c) 割れれば合成数、割れなければ素数 → b) へ戻る

この手順を `pchk` 関数に収めた。 n を 3 から順に割っていき、割れたら素数ではないので偽を意味する `false` を返す。最後まで割れなければ素数だから真である `true` を返すのだ。ちなみに Lua

では、偽を意味するものは `false` と `nil` のみで、それ以外は数値であれ文字であれ、すべて真と判断する。

ところで、`while` 構文の判定は `i <= math.sqrt(n)` で行なったが、`while i*i <= n` と書いても同じことである。こっちの方が入力文字数も少なくお得かもしれない。

`pchk` 関数は、前に素数の一覧を表示した `primes` 関数と違うね。なぜ、そうしたのだろうか。それは、`primes` 関数は一覧表にするのに適しているものの、与えられた数の判定には不向きだからである。ひとくちに素数かどうか調べるといっても、スクリプトの書き方はいろいろあるのだ。

さて、実際に双子素数を見つける関数は、`twinp(n)` のわずか数行分である。

m は $6m \pm 1$ の m に対応した整数変数であり、初期値は 1 である。これは 6 倍して用いられ、6 の倍数近辺の双子素数を調査するために使われる。調べる上限は関数実行時に与えられる n となる。`twinp(n)` も `pchk(n)` も `while` 構文を用いているが、もちろん `for` 構文に直せる。

さて、このスクリプトの実質的な内容は `and` 文に凝縮されている。まず `pchk(6 * m - 1)` だが、 $6 * m - 1$ の値—最初は 5 ということだ—を関数 `pchk(n)` に渡して、素数かどうかの判定を仰いでいる。そして最初の 5 は素数なので、`pchk(n)` は真を返してくる。

また、`pchk(6 * m + 1)` も同様だ。これは最初 7 を判定するので、やはり真を返してくるのである。すると問題は `and` が何をするのかということだ。A `and` B は論理演算子の一つで、A と B が共に真のときに限り、A `and` B は真になる。それ以外は偽と判定される。さらに A `and` B の特徴は、A が偽になれば B の真偽に関わらず、A `and` B は偽であるから、そのときは B をいちいち評価しないのだ。このことは計算量を押さえる役に立つ。

いずれにせよここでは、 $6m \pm 1$ が共に素数になったときだけ、真と判定されるというのが大事な点だ。それによって、`tex.print` 文で双子素数だけが表示されるのである。双子素数は各行にペアで表示されるようにした。また、判定が偽の場合は何もする必要がないので、`else` 文は省略してある。