8.3 ガウス素数

複素数の登場で世界が変わってしまった。とくに変わってしまうのが素数の世界である。なぜなら、いままでは 2 は素数の扱いをしていたが $2 = (1+i)(1-i)$ に分解できるからだ。しかし 3 はそうならない。3 は複素数の範囲にまで広げても素数であり続ける。

実数の世界で素数であったものが、複素数の世界で合成数になってしまう数にはどんなものがあるだろうか。そのようになる実数 x は

$$x = (a+bi)(a-bi)$$

を満たしているはずである。つまり、共役複素数の積でないといけない。そういうものを見つけるには **Python3** が適している。

[py script]

```
>>> def gprime(n):
...   for i in range(1, int(math.sqrt(n))+1):
...     m = math.sqrt(n - i * i)
...     if m == int(m):
...       print('%d = (%d + %di)(%d - %di)' % (n, i, m, i, m))
...       break
...   else:
...     print('ガウス素数')
...
>>> gprime(2) # 実行前に import math
2 = (1 + 1i)(1 - 1i)
>>> gprime(3)
ガウス素数
```

スクリプトは、素数 n が複素数の世界で分解できるか、できないのかを示すものである。しかし、このスクリプトは危険な入力を避けられない。つまり、どんな n に対しても処理をしてしまうので、入力する n は確実に素数であることが分かってないと困る。それが心配なら、以前の pchk 関数を組み込むとよい。

ガウス素数を見つけるには、入力された n から順に 1*1、2*2、3*3、4*4、... を引いたとき、残りが平方数になっているかどうかで判定している。だから、調べる範囲は $\sqrt{n}$ までで十分だ。もし、平方数を引いて平方数が残れば、それが求める積になるので出力すればよい。print 文の書式が新しいだろう。' 'で囲まれた部分は、そこに書かれた通りの形で出力される。ただし、%d には "%区切り" の後ろの ( ) にある変数が順に対応する。要するに、' ' 内に%d が 5 個あるので、それらが順に後ろの ( ) にある n、i、m、i、m に対応するのだ。

$x = (a + bi)(a - bi)$ に分解できるということは $x = (b + ai)(b - ai)$ にもできるということだから、ガウス素数でなければいくつかの分解の仕方がある。あまり自明の解を表示してもうるさいだけなので、break 文で抜けている。全部の分解を表示したければ break を削ってほしい。ところでこの break は for ループを抜けるためのものである。その下の else は for ブロックの一部であることに注意してもらいたい。だから、素数が分解できたら、else は実行されないのである。

では、else が実行されるのはいつだろうか。それは、for 文の処理がひとしきり終わったときだ。for 文の処理がひとしきり終わるというのは、調査範囲の最後まで計算してみたものの、素数が分解できなかった場合を指す。このときは一度も break 文が実行されていないときで、ここではじめて “ガウス素数” と表示されるのである。

一般の複素数にも、分解できるものとできないものがある。分解できれば合成数で、できないものがガウス素数と呼ばれる。たとえば $4 + 3i = (1 + 2i)(2 - i)$ であるから合成数だ。一方、$2 + 3i$ はガウス素数である。それはどこで分かるのだろうか。実は、$a + bi$ に対して、$a^2 + b^2$ が素数になればガウス素数であることが分かっている。実際、$4 + 3i$ は $4^2 + 3^2 = 25$（合成数）、$2^2 + 3^2 = 13$（素数）となっている。そういうことなら、ガウス素数の判定は簡単である。

[py script]

```
>>> def gpmsg(n):
...   msg = 'ガウス素数'
...   for i in range(2, n):
...     if n % i == 0:
...       msg = '合成数'
...   return msg
...
>>> def gpfind(c):
...   n = int(c.real**2 + c.imag**2)
...   print(gpmsg(n))
...
>>> gpfind(1+2j)  # 実行前に import math
ガウス素数
>>> gpfind(3+4j)
合成数
```

複素数 $a + bi$ がガウス素数かどうかは、$a^2 + b^2$ が素数かどうかにかかっているので、その判定はさほど効率的ではないけれど以前の pmsg 関数がそのまま使える。ただし、単なる素数でなくガウス素数の判定なので、関数名は gpmsg に変えておいた。だから gpfind 関数は、入力された複素数 c の $a^2 + b^2$ を判定するだけのことである。

複素数の**実部**と**虚部**はどうやって調べるのだろうか。**Python3** には**属性**という概念があり、複素数 c は実部の属性と虚部の属性を持っている。それは c.real と c.imag で取り出すことができ

る。それを返せば終了だ。

ところで、gpmsg 関数が値を返すときは return 文で、gpfind 関数が値を返すときは print 文である。この違いが何かというと、return 文が **Python3** 自身が計算した値をそのまま返すのに対して、print 文はわれわれが見やすい書式に変えて返す点だ。**Python3** 自身は内部で 2 進数や 16 進数を用いているので、われわれ 10 進数の世界とはだいぶ違う。どう違うか知りたければ、gpfind 関数の print(gpmsg(n)) を return n に変えて試すとよいだろう。いろいろなことが発見できるのではないかな？