Mathématiques et nombres

Quelques programmes et algorithmes reliés aux mathématiques et aux nombres.

- Théorie des nombres
- Nombre_remarquable
- ...

Calculs en précision arbitraire

Les calculs suivants renvoient des nombres avec des décimales bien particulières :

- $1/9^2$ = 0.012345679012345679012345679012345679012345679012345679012345 7...
- $1/99^2$ = 0.00010203040506070809101112131415161718192021222324252627282930313233343536373839404142434445464748495051525354555657585960616263646566676869707172737475767778798081828384858687888990919293949596979900010203040506070809101111...

Quelle est l'explication de ces particularités. Comment manipuler de tels nombres et les construire ?

L'idée est de s'intéresser au développement en série de Taylor de $1/x^2$ autour de a=1, ou de manière équivalente à la série de Maclaurin de $1/(1-x)^2$

Taylor, pour $f(x) = 1/x^2$:

$$f(x)=\sum _{n=0}^{\infty}{\frac {f^{(n)}(a)}{n!}}(x-a)^{n}}=1-2(x-1)+3(x-1)^2−4(x-1)^3+5(x-1)^4+...$$

Maclaurin :

$$1/(1-x)^2 \approx 1 + 2x + 3x^2 + 4x^3 + 5x^4 + 6x^5 + 7x^6 + 8x^7 + 9x^8 + ...$$

Ce développement va introduire des décimales particulières si x = 0.1, 0.01 ou 0.001

Programme

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
'Nombres magiques' 100/81 (et 10000/9801) dont les valeurs montrent des
```

```
décimales
consécutives

Librairie de multiprécision : http://mpmath.org/doc/current/basics.html

"""

from mpmath import mp # multiprecision library
n = 41
mp.dps = n
#mp.pretty = True
print('standard float value : ', 1/0.81)

print('Multiprécision :')
sumt = mp.mpf(0)
for i in range(40, 0, -1):
    t = mp.mpf(i) * (mp.mpf(1)/mp.mpf(10))**mp.mpf(i-1)

    sumt += t
    print(mp.nstr(t, n, strip_zeros=False, min_fixed=-n)[:n],
          mp.nstr(sumt, n, strip_zeros=False, min_fixed=-n)[:n])

print(100/mpf(9**2))

n = 51
mp.dps = n
#mp.pretty = True
print('standard float value : ', 1/0.9801)

print('Multiprécision :')
sumt = mp.mpf(0)
for i in range(25, 0, -1):
    t = mp.mpf(i) * (mp.mpf(1)/mp.mpf(100))**mp.mpf(i-1)

    sumt += t
    print(mp.nstr(t, n, strip_zeros=False, min_fixed=-n)[:n],
          mp.nstr(sumt, n, strip_zeros=False, min_fixed=-n)[:n])

print(10000/mpf(99**2))
```

Résultats

> standard float value : 1.2345679012345678
>
> Multiprécision :
>
> 0.000000000000000000000000000000000000040
> 0.000000000000000000000000000000000000040

0.00000000000000000000000000000000000390
0.00000000000000000000000000000000000430
0.00000000000000000000000000000000003800
0.00000000000000000000000000000000004230
0.00000000000000000000000000000000037000
0.00000000000000000000000000000000041230
0.00000000000000000000000000000000360000
0.00000000000000000000000000000000401230
0.00000000000000000000000000000003500000
0.00000000000000000000000000000003901230
0.00000000000000000000000000000034000000
0.00000000000000000000000000000037901230
0.00000000000000000000000000000330000000
0.00000000000000000000000000000367901230
0.00000000000000000000000000003200000000
0.00000000000000000000000000003567901230
0.00000000000000000000000000031000000000
0.00000000000000000000000000034567901230
0.00000000000000000000000000300000000000
0.00000000000000000000000000334567901230
0.00000000000000000000000002900000000000
0.00000000000000000000000003234567901230
0.00000000000000000000000028000000000000
0.00000000000000000000000031234567901230
0.00000000000000000000000270000000000000
0.00000000000000000000000301234567901230
0.00000000000000000000002600000000000000
0.00000000000000000000002901234567901230
0.00000000000000000000025000000000000000
0.00000000000000000000027901234567901230
0.00000000000000000000240000000000000000
0.00000000000000000000267901234567901230
0.00000000000000000002300000000000000000
0.00000000000000000002567901234567901230
0.00000000000000000022000000000000000000
0.00000000000000000024567901234567901230
0.00000000000000000210000000000000000000
0.00000000000000002345679012345679012300
0.00000000000000002000000000000000000000
0.00000000000000022345679012345679012300
0.00000000000000019000000000000000000000
0.00000000000000021234567901234567901230
0.00000000000000180000000000000000000000
0.00000000000000201234567901234567901230
0.00000000000000170000000000000000000000
0.00000000000001901234567901234567901230
0.00000000000001600000000000000000000000
0.00000000000017901234567901234567901230
0.00000000000015000000000000000000000000
0.00000000000167901234567901234567901230
0.00000000000140000000000000000000000000

0.00000000000156790123456790123456790123
0.00000000001300000000000000000000000000
0.00000000014567901234567901234567901230
0.00000000012000000000000000000000000000
0.00000000134567901234567901234567901230
0.00000001100000000000000000000000000000
0.00000001234567901234567901234567901230
0.00000010000000000000000000000000000000
0.00000011234567901234567901234567901230
0.00000090000000000000000000000000000000
0.00000101234567901234567901234567901230
0.00000800000000000000000000000000000000
0.00000901234567901234567901234567901230
0.00007000000000000000000000000000000000
0.00007901234567901234567901234567901230
0.00006000000000000000000000000000000000
0.00006790123456790123456790123456790123
0.00050000000000000000000000000000000000
0.00056790123456790123456790123456790123
0.00400000000000000000000000000000000000
0.00456790123456790123456790123456790123
0.03000000000000000000000000000000000000
0.03456790123456790123456790123456790123
0.20000000000000000000000000000000000000
0.23456790123456790123456790123456790123
1.00000000000000000000000000000000000000
1.23456790123456790123456790123456790123

1.23456790123456790123456790123456790123

---

standard float value : 1.0203040506070808

Multiprécision :

0.00000000000000000000000000000000000000000000000250
0.00000000000000000000000000000000000000000000000250
0.00000000000000000000000000000000000000000000024000
0.00000000000000000000000000000000000000000000024250
0.00000000000000000000000000000000000000000002300000
0.00000000000000000000000000000000000000000002324250
0.00000000000000000000000000000000000000000220000000
0.00000000000000000000000000000000000000000222324250
0.00000000000000000000000000000000000000021000000000
0.00000000000000000000000000000000000000021222324250
0.00000000000000000000000000000000000002000000000000
0.00000000000000000000000000000000000002021222324250
0.00000000000000000000000000000000000190000000000000
0.00000000000000000000000000000000000192021222324250

```
0.00000000000000000000000000000001800000000000000
0.00000000000000000000000000000001819202122232425 0
0.00000000000000000000000000000017000000000000000
0.00000000000000000000000000000017181920212223242 50
0.00000000000000000000000000001600000000000000000
0.00000000000000000000000000001617181920212223242 50
0.00000000000000000000000000015000000000000000000
0.00000000000000000000000000015161718192021222324 250
0.00000000000000000000000001400000000000000000000
0.00000000000000000000000001415161718192021222324 250
0.00000000000000000000001300000000000000000000000
0.00000000000000000000001314151617181920212223242 50
0.00000000000000000000120000000000000000000000000
0.00000000000000000001213141516171819202122232425 0
0.00000000000000000110000000000000000000000000000
0.00000000000000001112131415161718192021222324250
0.00000000000000010000000000000000000000000000000
0.00000000000000010111213141516171819202122232425 0
0.00000000000000900000000000000000000000000000000
0.00000000000009101112131415161718192021222324250
0.00000000000080000000000000000000000000000000000
0.00000000000809101112131415161718192021222324250
0.00000000007000000000000000000000000000000000000
0.00000000070809101112131415161718192021222324250
0.00000000600000000000000000000000000000000000000
0.00000000607080910111213141516171819202122232425 0
0.00000005000000000000000000000000000000000000000
0.00000050607080910111213141516171819202122232425 0
0.00000400000000000000000000000000000000000000000
0.00000405060708091011121314151617181920212223242 50
0.00030000000000000000000000000000000000000000000
0.00030405060708091011121314151617181920212223242 50
0.02000000000000000000000000000000000000000000000
0.02030405060708091011121314151617181920212223242 50
1.00000000000000000000000000000000000000000000000
1.02030405060708091011121314151617181920212223242 50

1.020304050607080910111213141516171819202122232425 26
```

Références

- https://mrob.com/ → 1/99**2
  - https://mrob.com/pub/math/numbers-14.html
  - https://mrob.com/pub/seq/digits.html
- https://fr.wikipedia.org/wiki/S%C3%A9rie_de_Taylor
- https://socratic.org/questions/what-is-the-taylor-series-expansion-of-f-x-1-x-2-at-a-1

- Multiprécision en Python : http://mpmath.org/
  - Documentation

From:
https://dvillers.umons.ac.be/wiki/ - **Didier Villers, UMONS - wiki**

Permanent link:
https://dvillers.umons.ac.be/wiki/teaching:progappchim:math_nombres?rev=1579006681

Last update: **2020/01/14 13:58**