1 Table of Contents

1 Obfuscated code or piece of art?
  1.1 Mandelbrot set
  1.2 Penrose patterns
  1.3 Bitcoin address & private key generator

2 Obfuscated code or piece of art?

This short notebook shows a few examples of Python code, designed to draw something, and shaped as what they will draw...

2.1 Mandelbrot set

This nice little code will write a visualization of the Mandelbrot set, on the domain $[-3,3] \times [-3i,3i]$, for 1500 $\times$ 1500 points, as a Bitmap (written manually in binary).

In [2]: %time
   
   b_ = (255, 
   lambda V ,B,c :c and Y(V*V+B,B, c 
   -1)if(abs(V)<6)else 
   (2+c-4*abs(V)**-0.4)/i 
   ) ;v, x=1500,1000;C=range(v*x 
   );import struct;P=struct.pack;M,\ 
   j = '<QIIHHHH',open('art/M.bmp','wb').write 
   for X in j('BM'+P(M,v**x*3+26,26,12,v,x,1,24))or C: 
   i ,Y=_;j(P('BBB',*(lambda T:(T*80+T**9 
   *i-950*T **99,T*70-880*T**18+701* 
   T **99),T*i**((1-T**45*2))))(sum( 
   [Y(0,(A%3/3.+X%v+(X/v+ 
   A/3.-x/2)/1j)*2.5 
   /x -2.7,i)**2 for \
   A in C

Out[2]: 10.9999454048 second(s)
2.2 Penrose patterns

This second nice little code will write a visualization of a Penrose tiling (infinite pattern) to a PNG image, of resolution 2000 × 2000.

In [3]: %time

```python
_ =

U1 "'"if!
1:"e,V=200
0,(0j-1)**-.2;
v,S=.5/ V.real,
[(0,0,4 e,4*e
V)];w=1 -v"def!
```

CPU times: user 3min 43s, sys: 296 ms, total: 3min 44s
Wall time: 3min 43s
2.3 Bitcoin address & private key generator

This is the most concise (and the most sexy!) implementation of the Bitcoin protocol to generate a new address and private key!

In [26]: %time

CPU times: user 4.17 s, sys: 20 ms, total: 4.19 s
Wall time: 4.18 s
Address: 1PMc1Xt6AMaBhDkX4De6nEPMA8XtDDLJhL
Privkey: 5Jrof4v1UX7Zzj724rr6sviw6RhpzcqZEcV8a3MJu1u5eBEb
CPU times: user 64 ms, sys: 24 ms, total: 88 ms
Wall time: 72.9 ms

Disclaimer: I am not the author of these small examples!

That's it for today!