411-PROGR-suma-cuadrados

November 4, 2017

```
In [5]: def cuadrado(n):
            return n*n
```

```
In [6]: def F(n):
            L = (n).digits()
            return sum(map(cuadrado,L))
```

```
In [7]: F(102)
```

```
Out[7]: 5
```

```
In [8]: def orbita0(n,N):
            L = [n]
            for _ in xsrange(N):
                n = F(n)
                L.append(n)
            return L
```

```
In [13]: [orbita0(n,17) for n in srange(150,170)]
```

```
Out[13]: [[150, 26, 40, 16, 37, 58, 89, 145, 42, 20, 4, 16, 37, 58, 89, 145, 42],
          [151, 27, 53, 34, 25, 29, 85, 89, 145, 42, 20, 4, 16, 37, 58, 89, 145],
          [152, 30, 9, 81, 65, 61, 37, 58, 89, 145, 42, 20, 4, 16, 37, 58, 89],
          [153, 35, 34, 25, 29, 85, 89, 145, 42, 20, 4, 16, 37, 58, 89, 145, 42],
          [154, 42, 20, 4, 16, 37, 58, 89, 145, 42, 20, 4, 16, 37, 58, 89, 145],
          [155, 51, 26, 40, 16, 37, 58, 89, 145, 42, 20, 4, 16, 37, 58, 89, 145],
          [156, 62, 40, 16, 37, 58, 89, 145, 42, 20, 4, 16, 37, 58, 89, 145, 42],
          [157, 75, 74, 65, 61, 37, 58, 89, 145, 42, 20, 4, 16, 37, 58, 89, 145],
          [158, 90, 81, 65, 61, 37, 58, 89, 145, 42, 20, 4, 16, 37, 58, 89, 145],
          [159, 107, 50, 25, 29, 85, 89, 145, 42, 20, 4, 16, 37, 58, 89, 145, 42],
          [160, 37, 58, 89, 145, 42, 20, 4, 16, 37, 58, 89, 145, 42, 20, 4, 16],
          [161, 38, 73, 58, 89, 145, 42, 20, 4, 16, 37, 58, 89, 145, 42, 20, 4],
          [162, 41, 17, 50, 25, 29, 85, 89, 145, 42, 20, 4, 16, 37, 58, 89, 145],
          [163, 46, 52, 29, 85, 89, 145, 42, 20, 4, 16, 37, 58, 89, 145, 42, 20],
          [164, 53, 34, 25, 29, 85, 89, 145, 42, 20, 4, 16, 37, 58, 89, 145, 42],
          [165, 62, 40, 16, 37, 58, 89, 145, 42, 20, 4, 16, 37, 58, 89, 145, 42],
          [166, 73, 58, 89, 145, 42, 20, 4, 16, 37, 58, 89, 145, 42, 20, 4, 16],
          [167, 86, 100, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
          [168, 101, 2, 4, 16, 37, 58, 89, 145, 42, 20, 4, 16, 37, 58, 89, 145],
          [169, 118, 66, 72, 53, 34, 25, 29, 85, 89, 145, 42, 20, 4, 16, 37, 58]]
```

Parece que todas las órbitas llegan al 4 o bien al 1. £Es ésto cierto?

```
In [7]: def orbita(n):
            L = [n]
            while n != 4 and n != 1:
                n = F(n)
                L.append(n)
            return L

In [8]: orbita(99)

Out[8]: [99, 162, 41, 17, 50, 25, 29, 85, 89, 145, 42, 20, 4]

In [9]: time L = [len(orbita(n)) for n in xsrange(1,10^4)]

Out[9]: Time: CPU 0.55 s, Wall: 0.55 s

In [10]: time L = [len(orbita(n)) for n in xsrange(10^4,10^5)]

Out[10]: Time: CPU 5.07 s, Wall: 5.07 s

In [11]: time L = [len(orbita(n)) for n in xsrange(10^5,10^6)]

Out[11]: Time: CPU 51.59 s, Wall: 51.60 s
```

La diferencia fundamental entre este problema y el de Collatz es que aquí la función tiende a crear una sucesión decreciente de iteraciones, es decir, para casi todos los valores de n, se verifica $n > F(n)$. Comprobemos:

```
In [12]: def buscador(N):
             for j in xsrange(5,N):
                 if F(j)>j:
                     print j

In [13]: buscador(10^6)

Out[13]: 5
         6
         7
         8
         9
         14
         15
         16
         17
         18
         19
         25
         26
         27
```

```
28
29
36
37
38
39
46
47
48
49
56
57
58
59
66
67
68
69
76
77
78
79
85
86
87
88
89
94
95
96
97
98
99
```

Parece claro que para $n \geq 100$ se verifica $n > F(n)$, de forma que podemos separar el problema en dos partes:

Si $n \geq 100$ debemos demostrar que $n > F(n)$. Puede encontrase una demostración en el PDF "sumas-cuadrados.pdf" en la carpeta "PDFs/PROGR/".

Para $n \leq 99$ utilizamos el ordenador para ver que todas las órbitas llegan al 1 o al 4 (ésto ya está comprobado porque la función *orbita* no entra en un bucle infinito en el intervalo $[1, 99]$.

In [ ]: