JOHN D. COOK
CONSULTING

Navigation

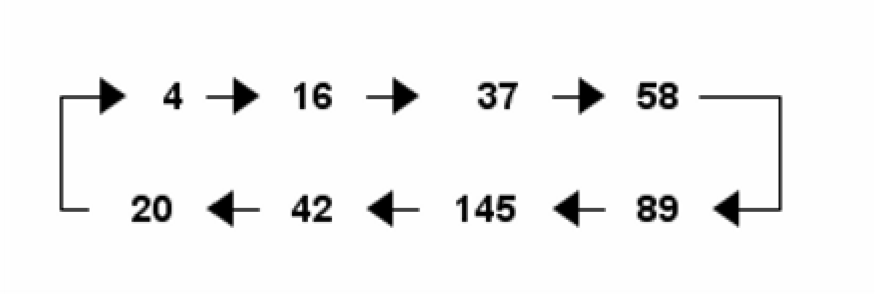
Squared digit sum

Posted on **24 March 2018** by **John**

Take any positive integer *n* and sum the squares of its digits. If you repeat this operation, eventually you'll either end at 1 or cycle between the eight values 4, 16, 37, 58, 89, 145, 42, and 20.

For example, pick *n* = 389. Then $3^2 + 8^2 + 9^2 = 9 + 64 + 81 = 154$.

Next, $1^2 + 5^2 + 4^2 = 1 + 25 + 16 = 42$, and now we're at one of the eight special values. You can easily verify that once you land on one of these values you keep cycling.



The following code shows that the claim above is true for numbers up to 1,000.

```
def square_digits(n):
    return sum( int(c)**2 for c in str(n) )

attractors = [1, 4, 16, 37, 58, 89, 145, 42, 20]

for n in range(1, 1000):
    m = n
    while m not in attractors:
        m = square_digits(m)

print("Done")
```

For a proof that the claim is true in general, see “A Set of Eight Numbers” by Arthur Porges, The American Mathematical Monthly, Vol. 52, No. 7, pp. 379-382.

***

By the way, the cycle image above was created with Emacs org-mode using the following code.

```
#+BEGIN_SRC ditaa :file square_digit_sum_cycle.png

+-> 4 -> 16 ->  37 -> 58---+
|                          |
|                          |
+- 20 <- 42 <- 145 <- 89 <-+

#+END_SRC
```

It’s not the prettiest output, but it was quick to produce. More on producing ASCII art graphs here.

Categories : Math Python

Bookmark the permalink

Previous Post
Asymptotic solution to ODE

Next Post
Sums of palindromes

14 thoughts on “Squared digit sum”

Eli Rose

24 March 2018 at 10:01

Unless I misunderstand, it looks like the proof in the article is at its core:

* Observe that, if you iterate this operation, you will eventually end up below some k. (The article uses k = 162).
* By computation, establish that the property holds for all n <= k.

The author asks for a more elegant proof. "The writer is aware of the inelegance of such a proof, and would like very much to see a simple indirect one."

Does anyone know of a more elegant proof?

NC

24 March 2018 at 10:43

interesting… I wrote a quick excel spreadsheet to produce this effect. I notice that there is a difference in how many iterations occur before the pattern sets in (3, 4, 5). There are also a few numbers that can be chosen that produce an 85 (usually when this number falls at the beginning of the repeating sequence of 8), which the algorithm “sees” as a 58 within this pattern. I was going to dig deeper to find the pattern of 1’s appearing. Thanks.

Om Joshi

24 March 2018 at 11:39

Just want to add that numbers that terminate are called happy numbers: https://en.m.wikipedia.org/wiki/Happy_number

viliml

24 March 2018 at 13:39

It's easy to prove that no three-digit number or greater will have the sum of its digits be larger than or equal to itself (109 comes closest, with a decrease of 27), so you need only check the numbers up to 99 (with intermediate results up to 162), which can be done by hand.

A. Reader

24 March 2018 at 15:17

Hi John, love your blog.
You may try http://viz-js.com/
syntax:
digraph G {
4 -> 16 -> 37 -> 58 -> 89 -> 145 -> 42 -> 20 -> 4;
}
use "neato" engine, output png.

JohnH

24 March 2018 at 15:21

I wondered if this held true for other bases, and it looks like in base 16 you either go to 1 or hit a 6-cycle of (d, a9, b5, 92, 55, 32).

JohnH

24 March 2018 at 16:00

Octal has more cycles and has two fixed points of 1 and 24. The three cycles are (4, 20), (15,32) and (5, 31, 12)

Not sure why I took of on this tangent, but it is interesting.

Peter Norvig

24 March 2018 at 18:32

The following slight variation is about 3 times faster:

```
square_digit = {str(i): i**2 for i in range(10)}.get

attractors = {1, 4, 16, 37, 58, 89, 145, 42, 20}

def is_attracted(n):
while n not in attractors:
n = sum(map(square_digit, str(n)))
return True

print(all(is_attracted(n) for n in range(1, 10**6)))
```

And here we can see a “histogram” of cycle lengths:

```
def cycle_length(n):
“The length of the square-digit cycle before reaching an attractor.”
count = 1
while n not in attractors:
```

```
n = sum(map(square_digit, str(n)))
count += 1
return count

Counter(cycle_length(n) for n in range(1, 10**6))

Counter({
1: 9,
2: 10071,
3: 55904,
4: 87555,
5: 214984,
6: 116275,
7: 173836,
8: 158606,
9: 86968,
10: 60293,
11: 19058,
12: 13980,
13: 2460})
```

Mike Scirocco

24 March 2018 at 21:43

Fun, I hope you will consider creating an RSS feed so I can follow your work in my RSS reader (Thunderbird).

John

24 March 2018 at 22:36

The RSS feed is https://www.johndcook.com/blog/feed

Mike Scirocco

24 March 2018 at 23:27

Great, thank you very much! Looking fwd to future installments.

JAM

28 March 2018 at 07:30

Many years ago (about 17 I think) I looked at this in lots of different bases and with different powers using FORTRAN . I found that whether you had (non-trivial) cycles or not depended on the base, and also (IIRC) that you could prove that certain things were true depending on the form of (e.g. b=3k+1), or in terms of the base. I don't think I ever wrote any of it up, but others might find it fun to look at.

Unsal Gokdag

29 March 2018 at 16:15

@Eli Rose,

Think about like this:

What is the biggest sum you can have in 3 digits: 999 -> 243.
So all 3 digit numbers will fall below that number.

What is the greatest number you can have until number 243: 199 -> 163
If you do that again what you will get is: 99 -> 162.

Maybe thats why you get most of the numbers under that.

Paul J Campbell

21 March 2020 at 11:14

I prefer the following version for reducing from 163: Thus, if we iterate over and over—say 164 times—starting with any number between 1 and 163, we must eventually repeat a number, since there are only 163 potentially different results. And once a number repeats, we have a cycle. Thus, iterating any 3-digit number eventually produces a cycle, and enumeration (plus keeping track and symmetry: e.g., 86, 68 give same cycle) resolves the question.

Leave a Reply

Your email address will not be published. Required fields are marked *

Comment

Name *

Email *

Website

Post Comment

Search …

Search

John D. Cook, PhD, President

My colleagues and I have decades of consulting experience helping companies solve complex problems involving data privacy, math, statistics, and computing.

Let’s talk. We look forward to exploring the opportunity to help your company too.

Email address

Your company's project …

SEND

JOHN D. COOK

Search …

SEARCH