HOW to PROVE it

SHAILESH SHIRALI

Elsewhere in this issue, two students report on their search for Happy Numbers. This is an interesting topic in recreational mathematics which throws up questions that are not easy to answer. We describe the essential proof techniques used in the study of such topics.

In our column for this issue, we deal with the notion of Happy Numbers. This has to do with the digital representation of a number, so we specify right at the start that we will be working throughout in base ten.

The SSQ iteration. Start with any positive integer and compute the sum of the squares of its digits. Do the same for the resulting number, then repeat for the new number, and continue the iteration. We call this the SSQ iteration. For example, the number 2375 yields:

$$\begin{aligned} 2375 &\to 87 \to 113 \to 11 \to 2 \to 4 \to 16 \to 37 \to 58 \\ &\to 89 \to 145 \to 42 \to 20 \to 4 \to 16 \to 37 \to 58 \\ &\to \cdots \to \cdots, \end{aligned}$$

and we see that we have got caught in a cycle, (4, 16, 37, 58, 89, 145, 42, 20). As the cycle has 8 numbers, we refer to it as an 8-cycle.

On the other hand, observe what happens if we start with the number 91:

$$91 \to 82 \to 68 \to 100 \to 1 \to 1 \to 1 \to 1 \to \cdots.$$

Having reached the number 1, we never leave it. Once again, we have reached a cycle, but this time the cycle has just one number; it is called a 1-cycle. The number in a **1-cycle** is called a **fixed point** of the iteration under study.

Keywords: Happy number, iteration, cycle, programming, Mathematica, Python

The two cycles we have found may be depicted in a more striking way as follows:

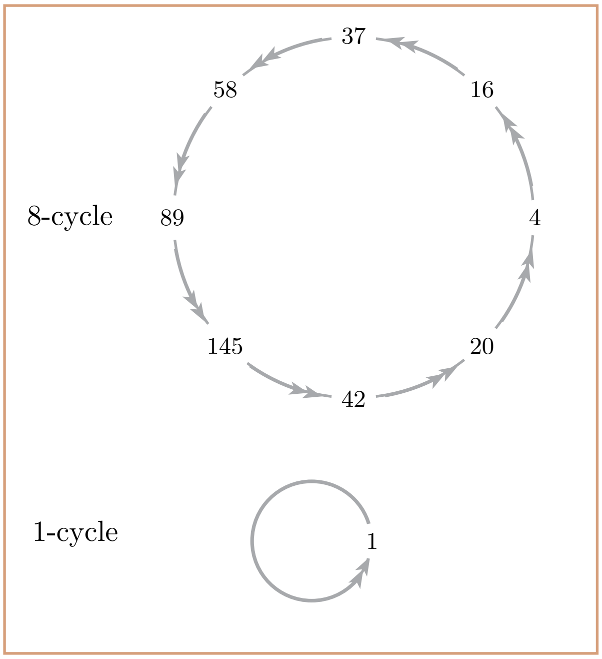


Figure 1

As noted in the companion article, numbers which ultimately map to the fixed point 1 (under the SSQ iteration) are referred to as *Happy Numbers.*

In this note, we shall establish that the SSQ iteration yields just the two cycles listed above. This conclusion is valid regardless of the choice of the initial number.

Proof of the claim that the SSQ iteration yields precisely 2 cycles. Suppose that n is a six-digit number. What is the largest that its SSQ value could be? Clearly, the largest value is attained when n = 999999, i.e., it is composed wholly of 9's. So the largest possible SSQ value for a six-digit number is $6 \times 81 = 486$. Observe that this is very much smaller than the original number.

If n were a seven-digit number, this discrepancy would be even larger. For, the largest possible SSQ value for a seven-digit number is $7 \times 81 = 567$.

In general, if n is a k-digit number, the largest its SSQ value can be is $k \times 9^2 = 81k$.

The smallest possible k-digit number is 10^{k-1}, and it is easy to check that $81k < 10^{k-1}$ for all $k \geq 4$.

We display here the values of $81k$ and 10^{k-1} for a few values of k:

k	1	2	3	4	5	6	. . .
$81k$	81	162	243	324	405	486	. . .
10^{k-1}	1	10	100	1000	10000	100000	. . .

The claim that $81k < 10^{k-1}$ for $k \geq 4$ is easy to prove using induction; we leave the details to the reader.

What the inequality proves is that if n has four or more digits, then its SSQ value is strictly less than n. Therefore, *no matter how large this initial number n is, by applying the* SSQ *operation repeatedly, we eventually obtain numbers with no more than three digits.*

It is possible to make a stronger statement. The largest possible SSQ value for a three-digit number is 243; the largest possible SSQ value for numbers not exceeding 243 is $1 + 9^2 + 9^2 = 163$ (achieved by 199); and the largest possible SSQ value for numbers not exceeding 163 is $9^2 + 9^2 = 162$ (achieved by 99). This chain of statements cannot be further continued, because 99 lies below 162. So we conclude by making the following statement: *No matter how large the initial number is, by applying the SSQ operation repeatedly, we eventually obtain numbers no larger than* 162.

Therefore, to list the cycles corresponding to this iteration, it suffices to study what the SSQ operation does to numbers between 1 and 162. This being a finite set, we are faced with a relatively simple task; all we need to do is to list the SSQ values of all the numbers between 1 and 162 and trace out the various trajectories (also called 'orbits'). For example, 3 leads to the trajectory 3, 9, 81, 65, 61, 37, ... We do not need to continue beyond 37, because we know that 37 is part of the 8-cycle listed earlier; following 37, we simply cycle around that 8-cycle indefinitely. Similarly, 5 leads to the trajectory 5, 25, 29, 85, 89, ... As in the earlier case, we do not need to continue beyond 89, because 89 is part of that same 8-cycle.

Proceeding in this way, we obtain all the cycles corresponding to this iteration. In the end, we find that there are just two cycles — the ones that we had listed earlier. We do not give the details of this listing here; it can be done in a straightforward manner. But it would be a good idea if *you* go ahead and trace out all the possible trajectories! If you do so, you will find that what we have claimed is true.

You may be slightly unhappy at the fact that we have left the last part of the proof incomplete; or rather, we have left the details of the verification to the reader. But you will find that this is true, in general, of all computer-assisted proofs. Most such proofs end with some such statement: "the details of the computation are left to the reader"; or: "the algebraic verification is left to the reader"; and so on. In some cases, the missing details can be filled in by pencil-and-paper computation. But if the setting is more complex, we may need to make use of sophisticated computer algebra software (for example, *Mathematica*) or a general purpose computer programming language such as Python.

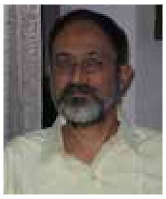
In the case of computer-assisted proofs, questions may arise such as: "How do we know that the program we have written does not contain any logical errors?" Or: "How do we know that the computer algebra software is error-free?" This is not the place to discuss such questions. But they do arise, inevitably, whenever we have a computer-assisted proof. (Such questions have indeed arisen in some cases, for example, the proof of the four-colour theorem, which depends in a critical way on a gigantic amount of machine computation. See [5] and [6] for details.)

A remark on the distribution of happy numbers. Extended computations using high-speed computers reveal some curious facts about the distribution of happy numbers; here are two of these facts (see [2] for details):

- It appears that roughly 1/7 of all the positive integers are happy! What is striking is that this fraction seems to persist as numbers get larger.
- It appears that the longest run of consecutive numbers which are all happy has length 5. Once again, this statistic does not seem to change as the numbers get larger.

References

1. Math forum, "Happy number", http://mathforum.org/library/drmath/view/55856.html
2. Walter Schneider, "Happy Numbers", https://web.archive.org/web/20060204094653/http://www.wschnei.de/digit-related-numbers/happy-numbers.html
3. Shailesh Shirali, *Adventures in Iteration*, Volume I (Universities Press, Hyderabad, India)
4. Wikipedia, "Happy number", https://en.wikipedia.org/wiki/Happy_number
5. Wikipedia, "Four color theorem", https://en.wikipedia.org/wiki/Four_color_theorem
6. Wikipedia, "Four color theorem – Proof by computer", https://en.wikipedia.org/wiki/Four_color_theorem#Proof_by_computer



SHAILESH SHIRALI is Director of Sahyadri School (KFI), Pune, and Head of the Community Mathematics Centre in Rishi Valley School (AP). He has been closely involved with the Math Olympiad movement in India. He is the author of several mathematics books for high school students, and serves as an editor for *At Right Angles*. He may be contacted at shailesh.shirali@gmail.com.