

Square Roots from 1; 24, 51, 10 to Dan Shanks

Ezra Brown

1. The World’s Oldest Algorithm. Arithmetic and numbers have always fascinated me. Even in the first grade, my friend Coleman and I used to race each other to see who could finish our pages of addition and subtraction and get the most answers right. (I usually won, but he’s now president of a very large jewelry store. Oh, well) In third grade, my friend Ron and I decided to write down all the Roman numerals up to 1000, just for the fun of it. And after long division in the fourth grade, I stood atop Arithmetic with no new worlds to conquer.

Or so I thought.

Right at the end of the fifth grade, we were told to read the *Iliad* and the *Odyssey* over the summer, and that in sixth grade math, Mrs. Garrison was going to teach us how to take square roots by hand. “Great!” I said. But what was a square root?

I soon found out. We children who grew up in New Orleans and who rode on truck floats on Mardi Gras Day all knew about perfect squares. A gross was a square, namely 12 times 12. You bought carnival throws and beads by the gross, and every kid knew that a gross was a dozen dozen, or 12 times 12, or 144. We soon learned that a square root was a number, such as 12, that you multiplied by itself to get the number you started with, such as 144. If you began with, say, 49, then the square root of 49 is 7, because $7 \cdot 7 = 49$. Likewise, the square root of 1 is 1 and the square root of 4 is 2.

But how could you take the square root of 2? It didn’t seem possible— $\sqrt{2}$ is not an integer—but I soon found out how to do it. I also found that Mrs. Garrison’s method applied to 2 never stopped; it just went on forever:

$$\sqrt{2} = 1.4142135623730950488016887242096980785696718753769 \dots$$

(That was my introduction to infinity, by the way.)

Eventually, I learned a number of things about square roots we’ll talk about in this paper:

- People have been interested in $\sqrt{2}$, and in square roots in general, for a long time.

- 1; 24, 51, 10 has something to do with $\sqrt{2}$ —if you know how to read it.
- One of the earliest methods of approximating square roots—Heron’s Method—is also one of the fastest.
- There’s a way of approximating square roots that uses things called continued fractions and has something to do with solutions of equations of the form $x^2 - ny^2 = 1$.
- Mrs. Garrison’s method dates back many hundreds of years, it is easy to learn, and it is easy to show that it does what it claims to do—namely, to find the closest decimal approximation to \sqrt{n} to a given number of places.
- Heron’s Method is a special case of a well-known method of approximating the roots of arbitrary functions, called Newton’s Method.
- If you want to calculate, not the real square root of 2, but the square root of 2 considered as an integer modulo, say,

$$p = 360027784083079948259017962255826129,$$

there’s a devilishly clever algorithm due to the late and legendary Dan Shanks that does the trick quickly. We’ll find out what that square root is later. Oddly enough, the Shanks–Tonelli Algorithm—to give it its proper name—doesn’t work mod 360027784083079948259017962255826079. If it did, that would be truly earthshaking. We’ll find out why later.

2. 1; 24, 51, 10 **and** $\frac{577}{408}$. The Old Babylonian Dynasty (from 1900 to 1600 B.C.) was located in the “Cradle of Civilization” in ancient Mesopotamia—the Land Between the Rivers. The flourishing trade economy in this part of the world gave impetus to the development of arithmetic. The ancient Babylonian numbering system was mainly sexagesimal (base sixty) with an admixture of the decimal system. The ancient Babylonians wrote numbers using a positional notation with neither a true zero nor a true “sexagesimal point”—magnitudes were inferred from context [1, p. 10–11]. The four hundred or so numerical tablets from the time that have been read reveal considerable computational facility: their arithmetic extended to summing arithmetic and geometric progressions, calculating squares and cubes—and calculating square roots.

Tablet No. 7289 from the Yale Collection includes the calculation of $\sqrt{2}$ to three sexagesimal places, namely

$$\sqrt{2} = 1 + \frac{24}{60} + \frac{51}{60^2} + \frac{10}{60^3} = 1;24,51,10$$

(using a semicolon as the sexagesimal point and commas as separators). This is very close to $\sqrt{2}$: squaring it yields 1;59,59,59,38,1,40, which differs from 2 by less than .000001696.

How did they find this approximation?

Our best guess is that they used a method of successive approximations based on two observations; namely, (1) if $x < \sqrt{2}$, then $\frac{2}{x} > \sqrt{2}$, and (2) the average $\frac{1}{2}(x + \frac{2}{x})$ is closer to $\sqrt{2}$ than either number. Most historians believe that they began the approximation scheme with $x = \alpha_1 = 3/2$; in the following, $\beta_i = 2/\alpha_i$, and $\alpha_{i+1} = \frac{1}{2}(\alpha_i + \beta_i)$:

i	α_i	β_i
1	$\frac{3}{2}$	$\frac{4}{3}$
2	$\frac{17}{12}$	$\frac{24}{17}$
3	$\frac{577}{408} = 1;24,51,10,35, \dots$	

When they found this approximation, they apparently truncated its sexagesimal representation to three places—rather like our using 1.414 as a three-decimal place approximation to $\sqrt{2}$.

You may be wondering about how much the ancients knew in general about finding square roots. To see a slightly bigger picture, we have to jump ahead almost 2000 years to the world of Alexandrian mathematics.

3. Heron's Square Root Algorithm. It was Heron of Alexandria (first century A.D.) who is credited with seeing this slightly bigger picture. His algorithm combines the α_i 's and β_i 's into one step, and it also works for finding \sqrt{n} for any positive n :

Theorem H (Heron's Method). *Let n and α be positive numbers, let d be a nonnegative integer and set $h(\alpha) = \frac{1}{2}(\alpha + \frac{n}{\alpha})$. Then:*

(a) \sqrt{n} is between α and n/α .

(b) If $\alpha > 0$, then $h(\alpha) \geq \sqrt{n}$.

(c) $|h(\alpha) - \sqrt{n}| = \frac{1}{2\alpha}(\alpha - \sqrt{n})^2$.

(d) If $|\alpha - \sqrt{n}| < \frac{1}{10^d}$, then $|h(\alpha) - \sqrt{n}| < \min\left(\frac{1}{10^{2d}}, \frac{\alpha - \sqrt{n}}{2}\right)$.

What (d) means is that if α approximates \sqrt{n} to d decimal places, then $h(\alpha)$ approximates \sqrt{n} to $2d$ decimals. That is, each time you apply Heron's Method, you double the number of digits of accuracy. The fact that the error in approximation is roughly the square of the error in the preceding step is why the convergence of Heron's Method to \sqrt{n} is said to be quadratic.

Heron's Method, it turns out, is just a special case of Newton's Method. If we let $f(x) = x^2 - n$, then $f'(x) = 2x$ and

$$N(\alpha) = \alpha - \frac{f(\alpha)}{f'(\alpha)} = \alpha - \frac{\alpha}{2} + \frac{n}{2\alpha} = \frac{1}{2}\left(\alpha + \frac{n}{\alpha}\right),$$

which is just Heron's approximation.

4. Continued Fractions. A Diophantine equation is an equation whose solutions are integers. Now $\frac{577}{408}$ has an interesting feature—namely,

$$577^2 - 2 \cdot 408^2 = 1.$$

It is also true that $17^2 - 2 \cdot 12^2 = 1$ and that $3^2 - 2 \cdot 2^2 = 1$. These observations lead us to one of the oldest problems in number theory, namely the existence, characterization, and construction of nonzero solutions to the Diophantine equation

$$x^2 - ny^2 = 1,$$

where n is a positive nonsquare integer. To tackle this problem, we turn to yet another way of representing numbers, different from decimals and sexagesimals, called **simple continued fractions**, or **scf's**. Finite scf's are based on the Euclidean algorithm for finding the greatest common divisor of two integers, so they go back over two thousand years. Infinite scf's first turn up in Europe in the sixteenth and seventeenth centuries in the work of Bombelli (1526-1573) and Huyghens (1629-1695), and it was Lagrange who used them to prove that if n is any positive nonsquare integer, then the Diophantine equation $x^2 - ny^2 = 1$ always has nonzero solutions.

All very well and good, you say, but what do they have to do with $\sqrt{2}$? and what are they, anyway? Fair questions—let’s answer the second one first. A finite simple continued fraction is an expression of the form

$$x = a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \frac{1}{a_3 + \dots + \frac{1}{a_{k-1} + \frac{1}{a_k}}}}},$$

where the a_i ’s are integers and $a_i \geq 1$ for $i \geq 1$. This notation is not easy to use, so we customarily write $x = \langle a_0, a_1, a_2, \dots, a_k \rangle$ to represent the above finite scf. We’ll call the a_i ’s **partial quotients** of x .

Since a_i is a positive integer for $i \geq 1$, it follows that $0 \leq x - a_0 < 1$, and $a_0 = [x]$, the greatest integer $\leq x$; if we put $x_0 = x$, $x_1 = \frac{1}{x_0 - a_0}$, and in general $x_{k+1} = \frac{1}{x_k - a_k}$, it turns out that $a_1 = [x_1]$, $a_2 = [x_2]$, and in general, $a_k = [x_k]$. It’s not too hard to prove the following:

Theorem CF1. (a) *Every positive rational number has exactly two representations as a finite scf, differing only in the last place—if $\rho = \langle a_0, a_1, a_2, \dots, a_k \rangle$ with $a_k > 1$, then the other finite scf representing ρ is $\langle a_0, a_1, a_2, \dots, a_k - 1, 1 \rangle$.*

(b) *Every finite scf represents a rational number.*

Somewhat trickier to prove is the following result (for a proof and a nice general treatment of continued fractions, see [2], Chapter 1):

Theorem CF2. *If a_0 is a real number, a_1, a_2, \dots are real numbers ≥ 1 , and $s_k = \langle a_0, a_1, \dots, a_k \rangle$, then $\lim_{k \rightarrow \infty} s_k$ exists. That is, the sequence $\{s_0, s_1, \dots, s_k, \dots\}$ converges to a real number γ .*

With Theorem CF2 at hand we may define the **infinite simple continued fraction**

$$\langle a_0, a_1, a_2, \dots \rangle = a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \dots}}$$

to be the limit of the sequence $\{s_0, s_1, s_2, s_3, \dots\}$.

Assuming that this definition makes sense, let’s determine the infinite scf expan-

sion of our friend $\sqrt{2}$ according to the method given prior to Theorem CF1.

$$\begin{aligned}x_0 &= \sqrt{2}, a_0 = [\sqrt{2}] = 1; \\x_1 &= \frac{1}{x_0 - a_0} = \frac{1}{\sqrt{2} - 1} = \sqrt{2} + 1, a_1 = [\sqrt{2} + 1] = 2; \\x_2 &= \frac{1}{x_1 - a_1} = \frac{1}{\sqrt{2} + 1 - 2} = \sqrt{2} + 1, a_2 = [\sqrt{2} + 1] = 2; \\x_3 &= \dots\end{aligned}$$

hey, wait a minute— $x_2 = x_1$ and $a_2 = a_1$, so it looks like this repeats! Evidently,

$$\sqrt{2} = \langle 1, 2, 2, 2, 2, \dots \rangle.$$

Such a repeating infinite scf

$$x = \langle a_0, a_1, \dots, a_j, a_{j+1}, \dots, a_{j+k}, a_{j+1}, \dots, a_{j+k}, \dots \rangle$$

is called **periodic**, and the shortest repeating part is called the **period**. As a shorthand, we write $x = \langle a_0, a_1, \dots, a_j, \overline{a_{j+1}, \dots, a_{j+k}} \rangle$, using the bar to indicate the period, much as we do with the periodic part of a repeating decimal. Thus, $\sqrt{2} = \langle 1, \overline{2} \rangle$, $\sqrt{13} = \langle 3, \overline{1, 1, 1, 1, 6} \rangle$ and

$$\sqrt{211} = \langle 14, \overline{1, 1, 9, 5, 1, 2, 2, 1, 1, 4, 3, 1, 13, 1, 3, 4, 1, 1, 2, 2, 1, 5, 9, 1, 1, 28} \rangle.$$

Periodic scf's play an important role in the Big Picture, which we'll see at the end of this section.

Meanwhile, let's compute scf's for some of the Heron approximations to $\sqrt{2}$. We begin with the scf for $\frac{17}{12}$ which we find by a combination of division and inversion:

$$\frac{17}{12} = 1 + \frac{5}{12} = 1 + \frac{1}{\frac{12}{5}} = 1 + \frac{1}{2 + \frac{2}{5}} = 1 + \frac{1}{2 + \frac{1}{\frac{5}{2}}} = 1 + \frac{1}{2 + \frac{1}{2 + \frac{1}{2}}} = \langle 1, 2, 2, 2 \rangle.$$

These are just the first four partial quotients in the scf for $\sqrt{2}$! If we do the same thing with $\frac{577}{408}$, streamlining the process a little bit, we get the first eight partial quotients in the scf for $\sqrt{2}$ —and a surprise as a bonus:

$$\frac{577}{408} = 1 + \frac{169}{408}; \quad \frac{408}{169} = 2 + \frac{70}{169}; \quad \frac{169}{70} = 2 + \frac{29}{70}; \quad \frac{70}{29} = 2 + \frac{12}{29};$$

hence,

$$\frac{577}{408} = \langle 1, 2, 2, 2, \frac{29}{12} \rangle = \langle 1, 2, 2, 2, 1 + \frac{17}{12} \rangle.$$

Now if $x = \langle x_0, \dots, x_k + y \rangle$ and $y = \langle y_0, \dots, y_m \rangle$, then

$$\begin{aligned} x &= x_0 + \frac{1}{a_1 + \dots + \frac{1}{x_k + y}} \\ &= x_0 + \frac{1}{a_1 + \dots + \frac{1}{x_k + y_0 + \frac{1}{y_1 + \dots + \frac{1}{y_m}}}}. \end{aligned}$$

Hence, $x = \langle x_0, \dots, x_k + y_0, \dots, y_m \rangle$. It follows that since $\frac{17}{12} = \langle 1, 2, 2, 2 \rangle$, we have that

$$\frac{577}{408} = \langle 1, 2, 2, 2, 1 + \langle 1, 2, 2, 2 \rangle \rangle = \langle 2, 2, 2, 2, 2, 2, 2 \rangle.$$

Does this pattern persist? Yes, indeed; in fact,

$$\begin{aligned} 1 &= \langle 1 \rangle, \\ \frac{3}{2} &= \langle 1, 2 \rangle, \\ \frac{17}{12} &= \langle 1, 2, 2, 2 \rangle, \\ \frac{577}{408} &= \langle 1, 2, 2, 2, 2, 2, 2 \rangle, \\ \frac{665857}{470832} &= \langle 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2 \rangle, \text{ and} \\ \frac{88673108897}{627013566048} &= \langle 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2 \rangle. \end{aligned}$$

In general, we have the following result:

Theorem CF3. *If $a_1 = 3/2$ is the first Heron approximation to $\sqrt{2}$, then*

$$a_k = \langle 1, \underbrace{2, 2, 2, \dots, 2}_{2^k - 1 \text{ twos in all}} \rangle;$$

that is, each iteration doubles the number of partial quotients.

Not only that, but the scf for each Heron approximation to $\sqrt{2}$ is obtained by truncating the infinite scf for $\sqrt{2}$ at an appropriate number of places.

All of this business raises an interesting question. The first partial quotient in the scf for \sqrt{n} is $[\sqrt{n}]$, the greatest integer $\leq \sqrt{n}$. That's easy to find if n is small—but what if, say,

$$n = 3600277840830799482590179622558261311759349942860011?$$

Is there a reasonable way to find $[\sqrt{n}]$, even if n is really big?

Well, there is, and the answer brings us back to Mrs. Garrison.

5. Square Roots By Hand. How do we find $[\sqrt{n}]$, so that we can calculate the scf for \sqrt{n} ? One possibility is to take a number with roughly half the number of digits in n , and use Heron's Method on it until the integer part doesn't change from one iteration to the next. For example, $n = 41897492$ has eight digits, so begin with some four-digit number—say, $\alpha_1 = 1000$. Then $\alpha_2 = 21448.746\dots$, $\alpha_3 = 11701.061\dots$, $\alpha_4 = 7640.859\dots$, $\alpha_5 = 6562.103\dots$, $\alpha_6 = 6473.434\dots$, $\alpha_7 = 6472.827\dots$ and $\alpha_8 = 6472.827\dots$, so that $[\sqrt{41897492}] = 6472$.

But there's a way to find square roots **by hand** that Mrs. Garrison taught us in sixth grade, and is based on the following result:

Theorem SRH. *Let a and b be non-negative integers, with $a > 0$ and $b < 100$. If $c = [\sqrt{a}]$ and $d = \max\{j : j \text{ is a non-negative integer and } (20c + j)j \leq 100(a - c^2) + b\}$, then $10c + d = [\sqrt{100a + b}]$.*

Proof. If c and d are as described, then

$$20cd + d^2 \leq 100a + b - 100c^2, \text{ or } (10c + d)^2 \leq 100a + b.$$

But

$$(10c + d + 1)^2 > 100a + b,$$

by the maximality of d . This implies that $10c + d = [\sqrt{100a + b}]$, as claimed.

Example. Let's try this on 41897492. First, group the digits in pairs, beginning at the decimal point:

$$\sqrt{41 \ 89 \ 74 \ 92}.$$

Let $a = 41$ and $b = 89$. Since $36 < a < 49$, we see that $c = [\sqrt{41}] = 6$, and so $a - c^2 = 41 - 36 = 5$. Now we need the largest d for which

$$(20c + d)d \leq 100(a - c^2) + b, \quad \text{i.e.} \quad (120 + d)d \leq 589.$$

Now $124 \cdot 4 = 496 < 589$, so 4 works, and $125 \cdot 5 = 625 > 589$, so 5 is too big. Hence $d = 4$ and so $\lceil \sqrt{4189} \rceil = 64$. For the next step, set $a = 4189$ and $b = 74$; we already know that $c = \lceil \sqrt{4189} \rceil = 64$, so ... but let's lay the whole procedure out. Note that successive values of d are boldfaced:

$$\begin{array}{r}
 \begin{array}{r}
 \sqrt{41} \\
 \hline
 6 \\
 41 \\
 \hline
 36 \\
 \hline
 5 \\
 89 \\
 \hline
 4 \\
 96 \\
 \hline
 93 \\
 74 \\
 \hline
 90 \\
 09 \\
 \hline
 3 \\
 65 \\
 92 \\
 \hline
 2 \\
 58 \\
 84 \\
 \hline
 1 \\
 07 \\
 08
 \end{array}
 \end{array}$$

To obtain the initial digits of the first trial divisor, we double $\lceil \sqrt{41} \rceil = 6$, i.e. $12 = 6 + 6$. To obtain the initial digits of the subsequent trial divisors, add the most recently found digit of the square root to the previous trial divisor. Thus, $128 = 124 + 4$, $1294 = 1287 + 7$, and so forth.

Thus, $41897492 = 6472^2 + 10708$. Can we do better? Certainly; this procedure doesn't really care where the decimal point is, as long as the digits are paired correctly, and the pairing begins at the decimal point. That is, the above procedure shows that $418974.92 = 647.2^2 + 107.08$ and that $41.897492 = 6.472^2 + 0.010708$.

Well, then, why not try it out on 2? If we do, here's what we get:

$$\begin{array}{r}
 \begin{array}{r}
 \sqrt{2} \\
 \hline
 1. \\
 00 \\
 \hline
 1 \\
 00 \\
 \hline
 96 \\
 \hline
 4 \\
 00 \\
 \hline
 2 \\
 81 \\
 \hline
 1 \\
 19 \\
 00 \\
 \hline
 1 \\
 12 \\
 96 \\
 \hline
 6 \\
 04 \\
 00 \\
 \hline
 5 \\
 65 \\
 64 \\
 \hline
 38 \\
 36 \\
 00 \\
 \hline
 28 \\
 28 \\
 41 \\
 \hline
 10 \\
 07 \\
 59 \\
 00 \\
 \hline
 8 \\
 48 \\
 52 \\
 69 \\
 \hline
 1 \\
 59 \\
 06 \\
 31
 \end{array}
 \end{array}$$

Thus, $2 = 1.414213^2 + 0.000001590631$, and $1.414214^2 = 2.000001237796$. It

follows that 1.414213 is the six-decimal place lower approximation to $\sqrt{2}$. We could continue this process as long as we please to get better and better approximations to $\sqrt{2}$.

How does this method compare with Heron's Method? No contest here! Heron's Method is much faster: it converges quadratically to \sqrt{n} , whereas the by-hand method gives you just one significant digit per iteration.

Then why bother with the by-hand method at all?

Good question. The reason is that it dates from a time when there were no calculators or computers, and in those days it was almost the only way to calculate \sqrt{n} at all!

And now, for something significantly (but not completely) different.

5. Square Roots (mod p). If arithmetic and numbers have been a fascination for me, then they were a downright passion for Daniel Shanks (1917–1996). Dan was an absolute master at devising and modifying algorithms for computing using quadratic forms, number fields, modular arithmetic and ordinary arithmetic. For example, he devised three separate algorithms (one of which is known by the picturesque name of SQUFOF) for factoring large numbers using the arithmetic of binary quadratic forms—polynomials of the form $ax^2 + bxy + cy^2$. But one of his cleverest pieces of work was his modification of an old procedure, due to A. Tonelli, for finding modular square roots.

Now, modular arithmetic is the familiar arithmetic of remainders. Just for the sake of review, recall that we fix some number m , called the **modulus**, and we consider two numbers a and b equivalent if they have the same remainder on division by m . If this is so, we say that a is **congruent** to b modulo m and write $a \equiv b \pmod{m}$. Thus, $55 \equiv 16 \pmod{13}$, but $55 \not\equiv 16 \pmod{11}$, since 39 is a multiple of 13 but not of 11. To see what we mean by modular square roots, we notice that $25^2 = 625 = 7 \cdot 89 + 2 \equiv 2 \pmod{89}$. That is, 25 is a square root of 2 in the world of mod 89 arithmetic. With this example in mind, we say that x is a **square root of $a \pmod{m}$** if $x^2 \equiv a \pmod{m}$ is true.

These modular square roots are worth knowing about and worth being able to compute. To give two examples, they show up in a scheme that emulates a fair coin toss over the telephone (see [7] and [4, p. 340–341]), as well as in a scheme that

computers use to verify a user's identity. It's easy to find them for small moduli—trial and error will do it—but for even moderately large ones, trial and error is out of the question. Worse than that, not every number a has a square root mod every modulus m . (For example, 2 has a square root mod 597035519, but not mod 597035539. Don't even try.)

Before we get to the Shanks–Tonelli Algorithm, we need several definitions (which may be familiar) and a theorem we're not going to prove. If a and $m \neq 0$ are integers with no common factors except 1, we define $\text{ord}_m(a)$, the **order of a (mod m)**, to be the least positive integer j for which $a^j \equiv 1 \pmod{m}$. For example, $\text{ord}_7(2) = 3$, since $2^3 \equiv 1 \pmod{7}$, and 3 is the smallest positive integer for which this is true. If there exists an integer d for which $a = md$, then we say that m **divides**, or is a **divisor** of, a and write $m|a$. That is, $m|a$ means that a is an integral multiple of m ; note that $13|39$ but $11 \nmid 39$. Finally, a and b are said to be **relatively prime** if they have no divisors in common except ± 1 .

The result we'll assume is the following:

Theorem FEL (Fermat–Euler–Lagrange). *Let a and m be relatively prime integers, with $m \neq 0$, let p be an odd prime, and let a and p be relatively prime. Then*

- (1) $\text{ord}_m(a)$ always exists.
- (2) (Fermat's Little Theorem) $a^{p-1} \equiv 1 \pmod{p}$; that is, $\text{ord}_p(a)|p-1$.
- (3) If $j = \text{ord}_m(a)$ and k is an integer, then $\text{ord}_m(a^k)|j$.
- (4) If $\text{ord}_p(a) = j$ and if $b \equiv a^{p-1-r} \pmod{p}$, then $ab \equiv 1 \pmod{p}$, and we call b an **inverse** of $a \pmod{p}$.
- (5) (Euler's Criterion) $a^{(p-1)/2} \equiv 1$ or $-1 \pmod{p}$ according as a does or does not have a square root \pmod{p} .

Suppose Euler's Criterion tells us that a has a square root \pmod{p} . Now p is an odd prime, so $p-1 = s \cdot 2^e$ with s odd and e positive. Then $x = a^{(s+1)/2}$ is almost the square root of $a \pmod{p}$, because

$$x^2 \equiv a^{s+1} \equiv a^s \cdot a \pmod{p}$$

and if $a^s \equiv 1 \pmod{p}$, then x is the square root. Dan Shanks points out that this is true in two-thirds of all cases (Turner [6] gives a neat proof of this fact). For

example, if $p = 23$ and $a = 3$, then $p - 1 = 11 \cdot 2$ and so $s = 11$. Sure enough,

$$(3^{(s+1)/2})^2 = (3^6)^2 = 3^{12} = 531441 = 3 + 23106 \cdot 23 \equiv 3 \pmod{23}.$$

What this means is that in the cases when $a^{(s+1)/2}$ is not the square root, it is only off by a fudge factor—and the Shanks–Tonelli algorithm keeps updating the fudge factor until it gets the correct answer.

Before we present the algorithm, here is a key lemma (not a Key Lime) that we will prove.

Key Lemma. *If p is a prime and $y^2 \equiv 1 \pmod{p}$, then $y \equiv \pm 1 \pmod{p}$.*

Proof. For then $p|y^2 - 1$ and so $p|(y-1)(y+1)$. Now if a prime divides a product, it must divide one of the factors. Hence, either $p|y - 1$, in which case $y \equiv 1 \pmod{p}$, or $p|y + 1$, in which case $y \equiv -1 \pmod{p}$.

Note that the lemma is not true for composite moduli— $y = \pm 1$ and $y = \pm 103$ all satisfy $y^2 \equiv 1 \pmod{221}$.

With that, we now list the steps of the **Shanks–Tonelli Algorithm**:

1. BEGIN with an integer a and a prime $p > 2$, relatively prime to a . Calculate $a^{(p-1)/2} \pmod{p}$. Since by Fermat’s Little Theorem, $a^{p-1} \equiv 1 \pmod{p}$, by the Key Lemma it follows that $a^{(p-1)/2} \equiv 1$ or $-1 \pmod{p}$.

2. IF $a^{(p-1)/2} \equiv -1 \pmod{p}$, then by Euler’s Criterion, a has no square root \pmod{p} . **EXIT** quietly.

3. IF $a^{(p-1)/2} \equiv 1 \pmod{p}$, then we’re in business. Write $p - 1 = s \cdot 2^e$ with s odd and e positive.

4. FIND a number n such that $n^{(p-1)/2} \equiv 1 \pmod{p}$ —that’s right, some number that does not have a square root \pmod{p} . You can test numbers sequentially, beginning with $n = 2 \dots$ it doesn’t usually take very long.

5. INITIALIZE the following variables (all congruences are \pmod{p}):

$$x \equiv a^{(s+1)/2} \text{ (first guess at the square root)}$$

$$b \equiv a^s \text{ (first guess at the fudge factor)}$$

$$g \equiv n^s \text{ (powers of } g \text{ will update both } x \text{ and } b)$$

$$r = e \text{ (exponent will decrease with each update of the algorithm).}$$

Note that $x^2 \equiv ba \pmod{p}$.

6. Now $b^{2^{r-1}} = a^{s \cdot 2^{r-1}} = a^{s \cdot 2^r / r} = a^{(p-1)/2} \equiv 1 \pmod{p}$, so by part 3 of Theorem FEL, there is a least integer m such that $0 \leq m \leq r-1$ and $b^{2^m} \equiv 1 \pmod{p}$. **FIND** that m —this you do by successive squarings and reductions \pmod{p} . That is, find m such that $\text{ord}_p(b) = 2^m$.

7. IF $m = 0$, we're done. **RETURN** the value of x and **EXIT** triumphantly.

8. IF $m > 0$, **UPDATE** the variables:

replace x by $x \cdot g^{2^{r-m-1}}$

replace b by $b \cdot g^{2^{r-m}}$

replace g by $g^{2^{r-m}}$

replace r by m .

Shanks observed that the algorithm will terminate, as the old value of b has order 2^m , but the new value has order at most 2^{m-1} . The reasoning is as follows. Set $y = b^{2^{m-1}}$. Now $y \equiv 1 \pmod{p}$, by the definition of order, but $y^2 \not\equiv 1 \pmod{p}$. Hence, by the Key Lemma, $y \equiv -1 \pmod{p}$. In the same way, $g^{2^{r-1}} \equiv -1 \pmod{p}$. Hence

$$(b \cdot g^{2^{r-m}})^{2^{m-1}} \equiv b^{2^{m-1}} g^{2^{r-m+m-1}} \equiv b^{2^{m-1}} g^{2^{r-1}} \equiv -1 \cdot -1 \equiv 1 \pmod{p}.$$

Hence, $\text{ord}_p(b \cdot g^{2^{r-m}}) \leq 2^{m-1}$, as claimed. It follows that the value of m decreases with each iteration of the algorithm.

9. GO BACK to Step 6 with the new value of r , which is the old value of m . As we have shown, r (and hence, m) decreases with each iteration of the algorithm. Eventually, m must equal zero, and when it does, Step 7 tells us to stop—we've got the answer.

What about the new value of x ? Note that the old value of x satisfies $x^2 \equiv ba \pmod{p}$; multiplying the old x by $g^{2^{r-m-1}}$ leads to the congruences

$$(xg^{2^{r-m-1}})^2 = x^2 g^{2^{r-m}} \equiv bag^{2^{r-m}} \equiv bg^{2^{r-m}} a \pmod{p}.$$

Hence, in multiplying x by $g^{2^{r-m-1}}$, we replace the old value of b in $x^2 \equiv ba \pmod{p}$ by the new value of b . Since this new value has order strictly less than that of the old value, we are making progress.

Let's run through a couple of examples to make some kind of sense out of this welter of symbols and terms.

Example 1. Find a square root of 2 modulo the prime 113.

SET UP: $a = 2, p = 113, p - 1 = 7 \cdot 2^4, e = 4, s = 7, (p - 1)/2 = 56, (s + 1)/4 = 4$

BEGIN: $2^{56} \equiv 1 \pmod{113}$; we're in business.

FIND n : $3^{56} \equiv -1 \pmod{113}$, so $n = 3$.

INITIALIZE: $x = a^{(s+1)/2} = 2^4 \equiv 16 \pmod{113}$; $b = a^s = 2^7 \equiv 15 \pmod{113}$;
 $g = n^s = 3^7 \equiv 40 \pmod{113}$; $r = e = 4$.

FIND $\text{ord}_p(b) = 2^m$: $b^2 = 225 \equiv -1, b^4 \equiv 1 \pmod{113}$. Hence $b^{2^2} \equiv 1 \pmod{113}$, and so $m = 2$.

$m \neq 0$, so **UPDATE:**

$$x = xg^{2^{r-m-1}} = 16 \cdot 40^{2^{4-2-1}} = 16 \cdot 1600 \equiv 16 \cdot 18 \equiv 62 \pmod{113};$$

$$b = bg^{2^{r-m}} = 15 \cdot 40^4 \equiv 15 \cdot (-15) \equiv 1 \pmod{113};$$

$$g = g^{2^{r-m}} \equiv -15 \pmod{113};$$

$$r = m = 2.$$

Since $b = 1$, $\text{ord}_p(b) = 1 = 2^0$; hence $m = 0$ and we're done: **RETURN** the current value of x , namely 62. Sure enough, $62^2 = 3844 = 2 + 34 \cdot 113 \equiv 2 \pmod{113}$, and so 62 is a square root of 2 mod 113. As it is with real numbers, an integer either has exactly two square roots modulo a prime p , or no square roots (mod p)—this is a consequence, by the way, of the Key Lemma. The other square root of 2 mod 113 is $51 \equiv -62 \pmod{113}$.

That first example didn't go through too many iterations, so here's one that does. We'll list the steps in a table.

Example 2. Find a square root of 5 modulo the prime 40961.

SET UP:

$$a = 5, p = 40961 = 1 + 5 \cdot 2^{13}; 5^{20480} \equiv 1 \pmod{p};$$

$$3^{20480} \equiv -1 \pmod{p}; n = 3;$$

$$x = a^{(s+1)/2} = 5^3 = 125;$$

$$b = a^s = 5^5 = 3125;$$

$$g = n^s = 3^5 = 243.$$

$$\text{ord}_p(b) \quad m \quad x = x \cdot g^{2^{r-m-1}} \quad b = b \cdot g^{2^{r-m}} \quad g = g^{2^{r-m}} \quad r$$

2^{11}	11	$8145 \equiv 125 \cdot 18088$	$37802 \equiv 3125 \cdot 20237$	20237	11
2^9	9	$35907 \equiv 8145 \cdot 8091$	$21227 \equiv 37802 \cdot 8603$	8603	9
2^7	7	$33206 \equiv 35907 \cdot 36043$	$26432 \equiv 21227 \cdot 19734$	19734	7
2^6	6	$34087 \equiv 33206 \cdot 19734$	$21153 \equiv 26432 \cdot 14529$	14529	6
2^5	5	$31533 \equiv 34087 \cdot 14529$	$8555 \equiv 21153 \cdot 19808$	19808	5
2^4	4	$32336 \equiv 31533 \cdot 19808$	$9282 \equiv 8555 \cdot 32406$	32406	4
2^3	3	$16114 \equiv 32336 \cdot 32406$	$26420 \equiv 9282 \cdot 31679$	31679	3
2^2	2	$19424 \equiv 16114 \cdot 31679$	$1 \equiv 26420 \cdot 14541$	14541	2

After eight iterations, we have $b = 1$, so we're done: $x^2 = 19424^2 \equiv 5 \pmod{40961}$. We found 19424 by successively multiplying the initial value of x by various powers of the initial value of g :

$$19424 \equiv 125 \cdot 18088 \cdot 8091 \cdot 36043 \cdot 19734 \cdot 14529 \cdot 19808 \cdot 32406 \cdot 31679 \pmod{40961}.$$

And now, as mentioned in the introduction, we can use the Shanks–Tonelli Algorithm to find a square root of $2 \pmod{p = 360027784083079948259017962255826129}$. It doesn't take long:

$$\begin{aligned} p &= 360027784083079948259017962255826129 \\ p - 1 &= 22501736505192496766188622640989133 \cdot 2^4 = s \cdot 2^e \\ a &= 2, a^{(p-1)/2} \equiv 1 \pmod{p} \\ n &= 23, g = n^s \equiv 125498114661614417168928780905935147 \pmod{p} \\ x &= a^{(s+1)/2} \equiv 14950440904776671273718554989608293 \pmod{p} \\ b &= a^s \equiv 345077343178303276985299407266217835 \pmod{p}. \end{aligned}$$

Now $\text{ord}_p(b) = 2^2$, so that we update x by multiplying by $g^{2^{4-2-1}}$; it turns out that

$$\begin{aligned} g^2 &\equiv 9928839158183314543463366096574742 \pmod{p}, \text{ and so} \\ x &\equiv 14950440904776671273718554989608293 \cdot g^2 \pmod{p} \\ &\equiv 162244492740221711333411667492080568 \pmod{p}. \end{aligned}$$

Finally, the new value of b is 1, so we're finished; that is,

$$162244492740221711333411667492080568^2 \equiv 2 \pmod{p}.$$

You can implement Shanks–Tonelli yourself: either use some sort of computer algebra system—one that has “powermod” functions built in to calculate $a^b \pmod{m}$ —or write a free-standing program to do it. (For more information on this algorithm, see [2], pp. 213-230.)

As previously mentioned, the Shanks–Tonelli Algorithm doesn’t work for $q = 360027784083079948259017962255826079$ —even though 2 *has* a square root mod q —because that number is composite. (Don’t believe me? Well, it just so happens that

$$240879636515128888541937896009793966^2 \equiv 2 \pmod{q}.$$

Try it yourself!) Except for an exhaustive search, there is no known algorithm for calculating square roots to composite moduli. The reason is that calculating square roots to composite moduli requires knowing the prime factorization of the modulus. Since the integer factorization problem is very hard in general, that is why it would be truly earthshaking if the Shanks–Tonelli Algorithm worked with composite moduli.

6. Ever so many questions. Naturally, you have them:

- *Are there ways to find square roots that do better than Heron—that is, whose convergence is better than quadratic?* Yes, but they’re a bit complicated to explain. Maybe some other time.

- *Who invented the $\sqrt{\quad}$ sign for the square root, and why was it chosen?* It was Christoff Rudolff who first used the radical sign in his 1525 book called *Die Coss* (see [3]). Notice that $\sqrt{\quad}$ is a kind of elongated lower-case “r”, the first letter in *radix*, which is Latin for “root”.

- *Is there a way to find cube roots, similar to Heron’s method? to Mrs. Garrison’s method?* Sure. The analog to Heron’s algorithm for cube roots is the more general Newton’s Method. Whereas Heron’s update of an approximation α for \sqrt{n} is $h(\alpha) = \frac{1}{2}(\alpha + \frac{n}{\alpha})$, Newton’s update $k(\beta)$ of an approximation β for $\sqrt[3]{n}$ —that is, a number c such that $c^3 = n$ —is $k(\beta) = \frac{1}{3}(2\beta + \frac{n}{\beta^2})$. As for cube root computation by hand, there is such an algorithm. Naturally, it’s a bit more complicated than the square root algorithm. If you want to figure it out for yourself, the result analogous to Theorem SRH is the following:

Theorem CRH. *Let a and b be non-negative integers, with $a > 0$ and $b < 1000$. If $c = \lceil \sqrt[3]{a} \rceil$ and $d = \max\{j : j \text{ is a non-negative integer and } (300c^2 + 30cj + j^2)j \leq 1000(a - c^3) + b\}$, then $10c + d = \lceil \sqrt[3]{1000a + b} \rceil$.*

- *How'd you find that square root of 2 mod that big composite number q ? I found out that q is a product of four primes $p_1 p_2 p_3 p_4$, such that 2 has a square root t_i mod each p_i . Then I used Shanks–Tonelli to find t_i for each i . Finally, I combined the t_i to find the answer you saw, by means of something called the Chinese Remainder Theorem. And no, I didn't cheat!*

- *You've told us about real square roots and square roots (mod p); are there other kinds of square roots? Absolutely. For any set S with a binary operation \circ , you can define square roots: just say that x is a square root of y if $y = x \circ x$. It doesn't matter whether \circ represents multiplication (mod m), matrix multiplication, functional composition, composition of symmetries or composition of points on an elliptic curve—where there's an operation, there's a square root. Oh, yes. What's an elliptic curve? Well, *that's* another story!*

REFERENCES

- [1] Ronald Calinger (ed.), *Classics of Mathematics*. Prentice–Hall, 1995.
- [2] R. Kumanduri and C. Romero, *Number Theory with Computer Applications*. Prentice–Hall, 1998.
- [3] Jeff Miller, Website. *Earliest uses of various mathematical symbols*, <http://members.aol.com/jeff570/mathsym.html>.
- [4] Kenneth H. Rosen, *Elementary Number Theory and its Applications*, 3rd edition. Addison–Wesley, 1993.
- [5] Daniel Shanks, Five number-theoretic algorithms. Proceedings of the Second Manitoba Conference on Numerical Mathematics (Univ. Manitoba, Winnipeg, Man., 1972), pp. 51–70. *Congressus Numerantium*, No. VII, *Utilitas Math.*, Winnipeg, Man., 1973.
- [6] Stephen M. Turner, Square roots mod p , *Amer. Math. Monthly* **101** (1994), 443–449.

- [7] Charles Vanden Eyden, Flipping a coin over the telephone, *Mathematics Magazine* **62** (1989), 167–171.