

by Terence Kelly (tpkelly@acm.org)[illegible]

The C program in Figure 1 demonstrates wayward fixed-width binary floating-point arithmetic. It naïvely computes $x = 0.1^n \times 10^n$ for n ranging from 1 to 47. Mathematically, of course, $0.1^n \times 10^n = (0.1 \times 10)^n = 1^n = 1$, but the output shows that `float` ops aren’t mathematical.

The calculation in Figure 1 is admittedly contrived, but it illustrates nasty surprises that can arise naturally in practical computations such as compound interest calculations and polynomial evaluation. Numerical gotchas can also plague seemingly mundane summations over large streams of widely varying values. Is there an easy way to avoid such hazards?

