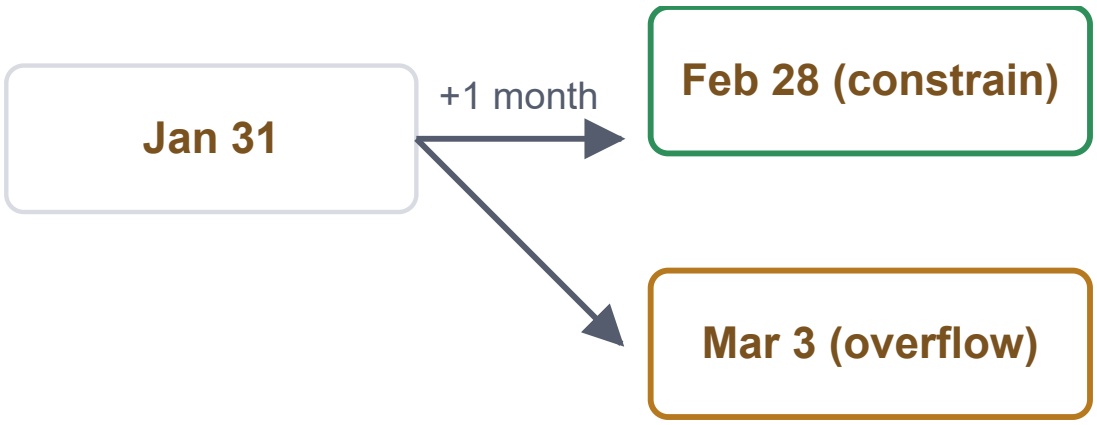
Why Adding a Month to Month-End Dates Breaks

> January 31 plus one month cannot land on February 31, so every date system has to pick a rule for what happens instead, and the rules disagree.

Adding a fixed number of days to a date is unambiguous. Adding "one calendar month" is not, because months do not all have the same length. January 31 plus one month has no exact equivalent in February, so a rule has to fill the gap, either constrain to the last valid day of the target month (landing on February 28 or 29), or overflow into the next month (landing on March 2 or 3). Both are used in real software, and the difference compounds across repeated monthly additions.



Same input, two internally consistent but different outputs -- the rule has to be stated

February has no 31st, so a month-end date-add either constrains to February's last day or overflows into March.

Where the ambiguity actually comes from

Calendar months have 28, 29, 30, or 31 days depending on the month and the leap-year rule, so "the same day number, next month" is not always a real date. The two common resolutions are constrain-to-end-of-month (if the target day does not exist, use the target month's last real day instead) and overflow (spill the excess days into the following month). Every date library, spreadsheet, and calendar app has picked one of these, and they do not all agree.

Why the effect compounds over repeated additions

Adding one month once to January 31 is a single edge case. But a recurrence rule that adds one month repeatedly (monthly billing on "the 31st," for example) hits this edge case every single month that lacks a 31st day, and a constrain-based system will drift toward earlier days in the

month over a year in a way an overflow-based system will not, or vice versa. See the companion resource on recurrence drift for the compounding version of this same problem.

What WhenMath states about this

The Add/Subtract Calendar Time and Month-End Roll Calculator engines both surface which rule applied whenever a month-end edge case is triggered, rather than silently picking one and moving on. If your own downstream system (payroll, billing, a calendar app) uses the opposite rule, expect a one-time or recurring date mismatch at every month-end boundary.

Key takeaway. Adding "a month" to a month-end date has no single universally correct answer. Check which rule (constrain or overflow) the systems on both sides of a date exchange actually use, especially for anything recurring.

Frequently asked

Which rule does WhenMath use by default?

WhenMath's calendar-math engines default to constrain-to-end-of-month and state so directly in the result whenever the input would otherwise land on a nonexistent date, so the assumption is visible rather than silent.

Does this affect adding years the same way?

Yes, in one specific case. Adding a year to February 29 has no February 29 in a non-leap target year, so the same constrain-or-overflow choice applies there too (typically resolving to February 28 or March 1).

Is this the same issue as leap years?

It is related but distinct. Leap years explain why February sometimes has 29 days, while month-end date math explains what happens when an add or subtract operation lands on a day that a given month does not have, leap year or not.

WhenMath is a browser-local calculation tool, not a legal, payroll, or scheduling authority. This guide explains a calculation convention; it is educational, not a determination that any specific contract, employer, or jurisdiction follows it.