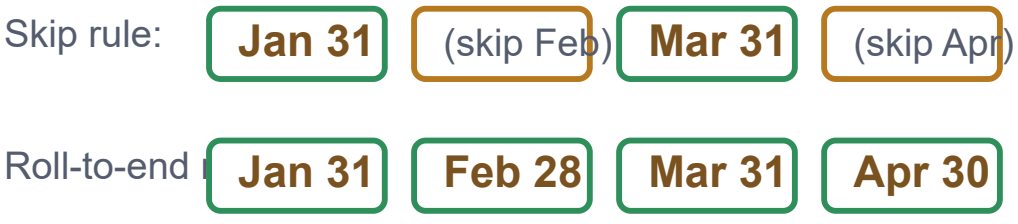
Recurrence Drift: Why "Every Month on the 31st" Breaks

> A recurrence rule anchored to a specific day number silently skips or shifts in any month that day does not exist, and the shift compounds over a year.

A recurrence rule like "on the 31st of every month" runs into the same month-end problem as a single date addition, except it happens repeatedly. Seven months of the year do not have a 31st day at all. Depending on how the recurrence engine resolves that gap (skip the month entirely, roll to the last day of the month, or roll into the next month), the same nominal rule produces materially different real schedules, and the differences accumulate rather than canceling out.

"Every month on the 31st," Jan through Apr



A day-of-month recurrence rule resolves February and April differently depending on the skip-vs-roll policy.

The gap every day-of-month rule has to resolve

Any recurrence rule pinned to a specific day number (the 29th, 30th, or 31st) hits months that do not contain that day. A skip policy simply produces no occurrence that month. A roll-to-end policy substitutes the month's actual last day. A roll-forward policy spills into the next month instead. All three are defensible, but they generate different total occurrence counts across a year and different actual dates for the affected months.

Why "close enough" reasoning fails here

Because the gap only affects specific months (February always, and April, June, September, and November for a 31st-anchored rule), a schedule built under one policy can look identical to a schedule built under another policy for eight months of the year and only diverge in the other four.

That makes the discrepancy easy to miss until the affected month actually arrives and two systems (a billing schedule and a calendar reminder, for instance) disagree about whether an occurrence happened.

How WhenMath's recurrence engines handle it

The Recurring Date Generator and Recurrence Drift Comparator both make the skip/roll-to-end/roll-forward choice an explicit input rather than a hidden default, and the Drift Comparator specifically runs the same nominal rule under more than one policy so the divergent months are visible side by side instead of discovered later.

Key takeaway. A recurrence rule anchored to a day number that not every month has (29th, 30th, 31st) needs an explicit skip-or-roll policy. Check which policy a downstream calendar or billing system actually uses before relying on a generated schedule matching it.

Frequently asked

Does this affect weekly or "every N days" recurrences too?

No. Fixed-interval recurrences (every 7 days, every 14 days) never land on a nonexistent date, since they are pure day-count arithmetic rather than a day-of-month lookup. The gap is specific to day-of-month and nth-weekday-style rules.

What about "last day of the month" as the rule itself?

That is actually the simplest case, since "last day" always resolves to a real date in every month by definition. It is a distinct, unambiguous rule rather than a policy choice layered on top of a fixed day number.

Can two systems both be "correct" and still disagree?

Yes. Skip, roll-to-end, and roll-forward are all internally consistent policies. The disagreement is not a bug in either system, only a mismatch in which policy each one implements.

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.