Assignment 4, Due 5/13

We're going to recreate the functionality of the Unix `cal` utility, and we're going to do it in two steps.

- First, we want to create a function called `day_of_week` that takes as input three integers giving a month, day, and year, and outputs an integer from 0 to 6 (inclusive), which is the day of the week the date occurred on. Here 0 corresponds to Sunday, 1 to Monday, etc.
- Next, we want to use that to write a utility that will print a nicely formatted calendar, either for a single month or a full year. To make life easier, we'll only do this for years *after* 1752. You can just implement a function that takes either one or two arguments (year or month and year) and prints appropriately. If you're feeling energetic, it'd be nice to make this work directly from the command line, too (like our original Collatz sequence program did).
- Your code should also be fairly robust — check the input data, and if it's invalid, print a usage message and raise an error.

Python already has a facility to do this — the `calendar` module implements all the functionality we're talking about (and probably more), though by different means (I don't think they use the Doomsday Rule). You should implement everything **without** using the Python `calendar` module, and without copy-pasting any code from it.

Some facts about dates

In order to make this all work, you'll need to know a few random facts about the Gregorian calendar.

- Every fourth year is a leap year. However, there are a few exceptions to this rule. Years divisible by 100 are *not* leap years, *unless* they are divisible by 400. So 1900 was not a leap year, 2000 was, but 2100 will not be. (I'm still hoping medical technology will progress enough to make this relevant information for some of us.) This is due to the fact that the Earth's orbit around the sun isn't *exactly* 365.25 days. (In fact, even this rule isn't enough — people have proposed making years divisible by 4000 *not* leap years ... Given that this calendar has only been around ∼250 years, it's probably a little early to start planning for 2000 years from now.)
- Thirty days has September, April, June, and November. All the rest have 31, except for February, which gets the short end of the stick.
- Britain switched to the Gregorian calendar in September of 1752. In order to account for the skew from their previous calendar, they eliminated several days — so September 2, 1752 was immediately followed by September 14, 1752. (Luckily, this was before the invention of computers — otherwise this would have totally overshadowed the whole "Y2K" thing.)

The Doomsday Rule

The Doomsday Rule is a simple algorithm for calculating the day of the week given a date, originally invented by John H. Conway. (There's even a Wikipedia page about it. You may also recognize his name as the inventor of the Game of Life — the cellular automata thing, not the board game with the little cars.) **Doomsday** is the last day of February. (It makes sense to use, since it's the biggest variable in the calendar.) The system is based on the following facts:

- The Gregorian calendar repeats every 400 years.
- All of the following easy-to-remember dates occur on the same day of the week as the last day of February in a given year: 4/4, 6/6, 8/8, 10/10, 12/12, 9/5, 7/11, 11/7, 5/9, and 3/14. (It's maybe helpful to remember something silly about "working 9–5 at the 7–11.")
- If no leap years are involved, Doomsday moves forward one day each year, because 365 days gives 52 weeks with one day left over.
- Whatever day Doomsday is in year `N`, it's the day after that in year `N+12`, unless the interval `[N, N+12]` happens to contain a year divisible by 100 but not 400.

So without further ado, here's the Doomsday Rule. Keep in mind that we want to know the day of the week, so we're going to do all our arithmetic `mod 7`. (Remember, this means that you can throw away any multiples of 7 that occur.) Given a date, you want to calculate five things:

1. **Days ahead** — how many days ahead you are from the nearest Doomsday. So if the date was November 10, you'd be three days off, and if it was November 25, you'd be four days off (because we're counting `mod 7`). **Important note**: order matters here. You need to make sure you do `(current date - doomsday)` instead of the other way around.
2. **Doomsday** — The Doomsday for your century. There's a little table below.
3. **Dozens** — Take the last two digits of your year and divide by 12. This is the quotient.
4. **Remainder** — This is the remainder from the last part.
5. **Leaps in remainder** — Finally, this is the number of times 4 goes into the remainder.

Each of these gives you a small(ish) integer. Add them up `mod 7`, and you've got the day of the week. Here are the Doomsdays for each century:

- 1800: Friday (5)
- 1900: Wednesday (3)
- 2000: Tuesday (2)
- 2100: Sunday (0)

An example

That may not make any sense at first, so let's go through an example. I was born on October 24, 1980, which I happen to know was a Friday. If I wanted to double-check this, though, here's what I'd do:

1. Days off: 10/10 is a Doomsday, which is off by exactly two weeks from 10/24, so it's **0** days off.
2. Doomsday: It's Wednesday for the 20th century, so that gives us a **3**.
3. Dozens: 80 / 12 = 6 with a remainder of 8, so we get a **6** for this one.
4. Remainder: The remainder was 8, which gives us a **1** (remember, `mod 7`).
5. Leaps in remainder: 8/4 has quotient **2**. (There's no remainder, but it's the one place where you throw away the remainder.)

So we add these up, and we get `0 + 3 + 6 + 1 + 2 = 5 mod 7`, confirming that I was indeed born on a Friday.

How it works

It's not too hard to see how this whole system works. Basically, the last four items in the list add up to give you what day Doomsday was in a given year, and then you just figure out how many days off you are from Doomsday. There's one really clever trick in that part, which is the "dozens" thing. As it happens, in a 12-year period not crossing between centuries, moving up 12 years pushes Doomsday up by 12 days from the extra day every year, plus 3 more from the three leap years. This is 15 days, which is just 1 `mod 7`. This means that you never really have to have numbers bigger than 12 that you're doing much with other than the original, making this manageable to do in your head.

You can also use this another way: if you just memorize what Doomsday is in the *current* year, it makes it really easy to determine what day of the week something occurs on. For instance, this year Doomsday is a Sunday — so the Fourth of July will also be a Sunday (7/11 is a Doomsday, so go back one week), and Christmas will be a Saturday (12/12 is a Doomsday, and hence so are 12/19 and 12/26). You can even use it backwards: 9/5 is a Sunday, so Labor Day will be September 6.