Happy Birthday POW

Problem Statement:

- The goal of this problem is to create a method for finding the day of the week a person was born using only their birthdate.
- The given information included the provided celebrity birthdays, month lengths, leap year rules, and the calendar for October 2025.
- The final result should be an accurate system that finds the weekday for any given date after 1900 without using a calendar.

Process:

Approach 1:

We first assigned numerical values for each day of the week. The week repeats every 7 days; therefore, it is effective to assign each day of the week a number in mod 7. We also chose to ignore the special rules regarding leap years that are multiples of 100 and 400 because those rules do not concern a solution that only has to apply for dates post 1900.

- Sunday: 0
- Monday: 1
- Tuesday: 2
- Wednesday: 3
- Thursday: 4
- Friday: 5
- Saturday: 6

We assumed a day for January 1st, 2025, to figure out a pattern for the start of the month.

We assumed January 1, 2025, corresponded to 0 (Sunday). Then, for each subsequent month, we determined its first day by adding the number of days in the previous month (mod 7) to the value of the previous month's first day.

(Number of days in month) mod 7 = x
(x + first day of last month) mod 7 = first day of current month

Jan	Feb	Mar	April	May	June	July	Aug	Sept	Oct	Nov	Dec
0	3	3	6	1	4	6	2	5	0	3	5

After finding the initial values of the days of the week for the first day of a month assuming that January 1 was a Sunday, we found that it would be difficult to manually compute the accurate value for January 1, 2025 with the given celebrity birthdays and the calendar for the month of October.

Approach 2: Finding the day of the week of January 1, 0 and creating a general formula.

Variable and Components	Meaning of Variable
n	Number of years since 0 (1 BCE)
k	Day of month of a given date
d	Day of the week of a given date

January 1, 0 was assumed to be a Sunday, even though no evidence supported this. We tested Monday October 13, 2025 using the following components:

- (n*365) to convert years to days
- $\lfloor \frac{n}{4} \rfloor$ takes account of extra leap days.

$$(n \times 365) + \lfloor \frac{n}{4} \rfloor + k = d \pmod 7$$

The value of *d* was 3, meaning that the formula showed October 13, 2025 as a Wednesday. However, October 13, 2025 was Monday, so an offset of 5 was added to correct it.

The updated formula now reads:

$$(n \times 365) + \lfloor \frac{n}{4} \rfloor + (k + 5) = d \pmod 7$$

The formula worked correctly for January and October when the offset was 5.
It failed for other months, indicating that each month required a different offset.

Jan	Feb	Mar	April	May	June	July	Aug	Sept	Oct	Nov	Dec
0	3	3	6	1	4	6	2	5	0	3	5

- Each value represents the day of the week for the first day of that month, if January 1 is Sunday.
  - These values come from adding the number of days in each month mod 7.
- For leap years, values after February increase by 1.

January and October have the same offset because the first days of both months are on the same weekday.

A value of 5 was added to the offsets for the first day of each month because January required an offset of 5 to align correctly. If January's offset needed adjustment, the same offset had to be applied to all months.

Month	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
a	5	1	1	4	6	2	4	0	3	5	1	3

$$(n \times 365) + \lfloor \frac{n}{4} \rfloor + (k + a) = d \quad \text{mod}(7)$$

The formula was tested using multiple celebrity birthdays from the provided list, covering various months and including leap years. It was found to provide the correct result for all cases.

<u>Solution:</u>

1. Use the formula $(n \times 365) + \lfloor \frac{n}{4} \rfloor + (k + a) = d$ and save the final number where n is the year of birth, *k* is the day of birth (if the birthday is October 12, *k* = 12), *a* is the offset variable which is different for each month, and *d* is the day of the week, which changes depending on the month of birth. Select *a* from the table below.

Month	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
a	5	1	1	4	6	2	4	0	3	5	1	3

2. After computing, divide the number by 7
3. Subtract the whole number from the result of step 2, and take note of the repeating decimal found.
4. Convert the repeating decimal into a fraction using the values below

Repeating Decimal	Fraction
0	0/7
0.142857	1/7
0.285714	2/7
0.428571	3/7
0.571428	4/7
0.714285	5/7
0.857142	6/7

5. Correspond the numerator of the repeating decimal to a day of the week, this day will be the day of the week of any given date after January 1, 0.

Numerator	Day of the Week
0	Sunday
1	Monday
2	Tuesday
3	Wednesday
4	Thursday
5	Friday
6	Saturday

The formula works because it counts all the days from a fixed starting year, adding extra days for leap years and adjusting for the varying lengths of each month. The formula follows the rules for leap years after 1900. The month is accounted for using the offset value a, and the day of the week is determined by calculating the total number of days that have passed since a starting point and taking that total modulo 7. By combining these factors, the formula consistently produces the correct weekday for any date after 1900, including leap year.

Extensions:

- Genghis Khan: May 31st, 1162: Create a general formula that takes the different rules for leap years before 1900 into account. We chose this extension because it would modify the use of the floor operator and the number of leap days accounted for. The offsets for each month, however, would likely stay the same, but the value of January 1, 0 would need to change.

- Determine the day of the week a person's birthday falls on each year and identify any repeating patterns over time. This helps reveal how the calendar cycles, and it would show how leap years shift dates and how long it takes for the same weekday to recur.

- Find the next year a given birthday will occur on the same weekday. This problem would likely involve creating a step by step method that accounts for the number of days passed between two dates (a new computation), the number of leap days between two dates, and the current day of the week. The problem would likely have to utilize modular arithmetic to show the cyclic nature of calendars and days of the week.