WIKIPEDIA

Determination of the day of the week

The **determination of the day of the week** for any date may be performed with a variety of algorithms. In addition, perpetual calendars require no calculation by the user, and are essentially lookup tables. A typical application is to calculate the day of the week on which someone was born or a specific event occurred.

Contents

Concepts

In numerical calculation, the days of the week are represented as weekday numbers. If Monday is the first day of the week, the days may be coded 1 to 7, for Monday through Sunday, as is practiced in ISO 8601. The day designated with 7 may also be counted as *0*, by applying the arithmetic modulo 7, which calculates the remainder of a number after division by 7. Thus, the number 7 is treated as 0, 8 as 1, 9 as 2, 18 as 4 and so on. If Sunday is counted as day 1, then 7 days later (i.e. day 8) is also a Sunday, and day 18 is the same as day 4, which is a Wednesday since this falls three days after Sunday.

Standard	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday	Usage examples
ISO 8601	1	2	3	4	5	6	7	%_ISODOWI%, %@ISODOWI[]% (4DOS);[1] DAYOFWEEK() (HP Prime)[2]
	0	1	2	3	4	5	6	
	2	3	4	5	6	7	1	%NDAY OF WEEK% (NetWare, DR-DOS[3]); %_DOWI%, %@DOWI[]% (4DOS)[1]
	1	2	3	4	5	6	0	HP financial calculators

The basic approach of nearly all of the methods to calculate the day of the week begins by starting from an 'anchor date': a known pair (such as January 1, 1800 as a Wednesday), determining the number of days between the known day and the day that you are trying to determine, and using arithmetic modulo 7 to find a new numerical day of the week.

One standard approach is to look up (or calculate, using a known rule) the value of the first day of the week of a given century, look up (or calculate, using a method of congruence) an adjustment for the month, calculate the number of leap years since the start of the century, and then add these together along with the number of years since the start of the century, and the day number of the month. Eventually, one ends up with a day-count to which one applies modulo 7 to determine the day of the week of the date.[4]

Some methods do all the additions first and then cast out sevens, whereas others cast them out at each step, as in Lewis Carroll's method. Either way is quite viable: the former is easier for calculators and computer programs; the latter for mental calculation (it is quite possible to do all the calculations in one's head with a little practice). None of the methods given here perform range checks, so unreasonable dates will produce erroneous results.

Corresponding days

Every seventh day in a month has the same name as the previous:

Day of the month	*d*
00 07 14 21 28	0
01 08 15 22 29	1
02 09 16 23 30	2
03 10 17 24 31	3
04 11 18 25	4
05 12 19 26	5
06 13 20 27	6

Corresponding months

"Corresponding months" are those months within the calendar year that start on the same day of the week. For example, September and December correspond, because September 1 falls on the same day as December 1. Months can only correspond if the number of days between their first days is divisible by 7, or in other words, if their first days are a whole number of weeks apart. For example, February of a common year corresponds to March because February has 28 days, a number divisible by 7, 28 days being exactly four weeks. In a leap year, January and February correspond to different months than in a common year, since adding February 29 means each subsequent month starts a day later.

The months correspond thus:
For common years:

- January and October.
- February, March and November.
- April and July.
- No month corresponds to August.

For leap years:

- January, April and July.
- February and August.
- March and November.
- No month corresponds to October.

For all years:

- September and December.
- No month corresponds to May or June.

In the months table below, corresponding months have the same number, a fact which follows directly from the definition.

Common years	Leap years	*m*
Jan Oct	Oct	0
May		1
Aug	Feb Aug	2
Feb Mar Nov	Mar Nov	3
Jun		4
Sept Dec		5
Apr July	Jan Apr July	6

Corresponding years

There are seven possible days that a year can start on, and leap years will alter the day of the week after February 29. This means that there are 14 configurations that a year can have. All the configurations can be referenced by a dominical letter, but as February 29 has no letter allocated to it a leap year has two dominical letters, one for January and February and the other (one step back in the alphabetical sequence) for March to December. For example, 2018 is a common year starting on Monday, meaning that 2018 corresponds to the 2007 calendar year and with the last 10 months corresponds to the 2012 calendar year. On the other hand, 2019 is a common year starting on Tuesday, meaning that 2019 corresponds to the 2013 calendar year. 2020 is a leap year starting on Wednesday, meaning that 2020 corresponds to the 1992 calendar year, meaning that the first two months of the year begin on the same day as they do in 2014 (i.e. January 1 is a Wednesday and February 1 is a Saturday) but because of a leap day the last ten months correspond to the last ten months in 2015 (i.e. March 1 is a Sunday to December 31 is a Thursday.). 2021 is a common year starting on Friday, meaning that 2021 corresponds to the 2010 calendar year and with the first 2 months corresponds to the 2016 calendar year. 2022 is a common year starting on Saturday, meaning that 2022 corresponds to the 2011 calendar year and with the last 10 months corresponds to the 2016 calendar year. For details see the table below.

Year of the century mod 28	*y*
00 06 12 17 23	0
01 07 12 18 24	1
02 08 13 19 24	2
03 08 14 20 25	3
04 09 15 20 26	4
04 10 16 21 27	5
05 11 16 22 00	6

Notes:

- Black means the all months of Common Year
- Red means the first 2 months of Leap Year
- Blue means the last 10 months of Leap Year

Corresponding centuries

See the table below.

Julian century mod 700	Gregorian century mod 400	*Day*
400: 1100 1800 ...	300: 1500 1900 ...	*Sun*
300: 1000 1700 ...		*Mon*
200: 0900 1600 ...	200: 1800 2200 ...	*Tue*
100: 0800 1500 ...		*Wed*
000: 1400 2100 ...	100: 1700 2100 ...	*Thu*
600: 1300 2000 ...		*Fri*
500: 1200 1900 ...	000: 1600 2000 ...	*Sat*

The Julian starts on Thursday and the Gregorian on Saturday.

Tabular methods to calculate the day of the week

Complete table: Julian and Gregorian calendars

For Julian dates before 1300 and after 1999 the year in the table which differs by an exact multiple of 700 years should be used. For Gregorian dates after 2299, the year in the table which differs by an exact multiple of 400 years should be used. The values "r0" through "r6" indicate the remainder when the Hundreds value is divided by 7 and 4 respectively, indicating how the series extend in either direction. Both Julian and Gregorian values are shown 1500–1999 for convenience. Bold figures (e.g., **04**) denote leap year. If a year ends in 00 and its hundreds are in bold it is a leap year. Thus 19 indicates that 1900 is not a Gregorian leap year, (but **19** in the Julian column indicates that it *is* a Julian leap year, as are all Julian *x*00 years). **20** indicates that 2000 is a leap year. Use **Jan** and **Feb** only in leap years.

100s of Years		D W	D L	D D	Remaining Year Digits				Month					#
Julian (r ÷ 7)	Gregorian (r ÷ 4)													
r5 19	**16 20 r0**	Sa	A	Tu	00 06 17 23	**28** 34 45 51	**56** 62 73 79	**84** 90	Jan				Oct	0
r4 18	15 19 r3	Su	G	W	01 07 **12** 18	29 35 **40** 46	57 63 **68** 74	85 91 **96**			May			1
r3 17	N/A	M	F	Th	02 13 19 **24**	30 41 47 **52**	58 69 75 **80**	86 97	**Feb**			Aug		2
r2 16	18 22 r2	Tu	E	F	03 **08** 14 25	31 **36** 42 53	59 **64** 70 81	87 **92** 98	Feb	Mar			Nov	3
r1 15	N/A	W	D	Sa	09 15 **20** 26	37 43 **48** 54	65 71 **76** 82	93 99			Jun			4
r0 14	17 21 r1	Th	C	Su	**04** 10 21 27	**32** 38 49 55	**60** 66 77 83	**88** 94				Sep	Dec	5
r6 13	N/A	F	B	M	05 11 **16** 22	33 39 **44** 50	61 67 **72** 78	89 95	**Jan**	Apr	Jul			6

For determination of the day of the week (1 January 2000, Saturday)

- the day of the month: 1 ~ 31 (1)
- the month: (6)
- the year: (0)
- the century mod 4 for the Gregorian calendar and mod 7 for the Julian calendar DW: (Sa)
- adding Sa + 1 + 6 + 0 = Sa + 7. Dividing by 7 leaves a remainder of 0, so the day of the week is Saturday.

For determination of the dominical letter (2000, B**A**)

- the century DL: (A)
- the year: (0)
- subtracting A - 0 = A.

For determination of the doomsday (2000, Tuesday)

- the century DD: (Tu)
- the year: (0)
- adding Tu + 0 = Tuesday.

Revised Julian calendar

Note that the date (and hence the day of the week) in the Revised Julian and Gregorian calendars is the same from 14 October 1923 to 28 February AD 2800 inclusive and that for large years it may be possible to subtract 6300 or a multiple thereof before starting so as to reach a year which is within or closer to the table.

To look up the weekday of any date for any year using the table, subtract 100 from the year, divide the difference by 100, multiply the resulting quotient (omitting fractions) by seven and divide the product by nine. Note the quotient (omitting fractions). Enter the table with the Julian year, and just before the final division add 50 and subtract the quotient noted above.

Example: What is the day of the week of 27 January 8315?

8315-6300=2015, 2015-100=1915, 1915/100=19 remainder 15, 19x7=133, 133/9=14 remainder 7. 2015 is 700 years ahead of 1315, so 1315 is used. From table: for hundreds (13): 6. For remaining digits (15): 4. For month (January): 0. For date (27): 27. 6+4+0+27+50-14=73. 73/7=10 remainder 3. Day of week = Tuesday.

Dominical Letter

To find the Dominical Letter, calculate the day of the week for either 1 January or 1 October. If it is Sunday, the Sunday Letter is A, if Saturday B, and similarly backwards through the week and forwards through the alphabet to Monday, which is G.

Leap years have two Sunday Letters, so for January and February calculate the day of the week for 1 January and for March to December calculate the day of the week for 1 October.

Leap years are all years which divide exactly by four with the following exceptions:

In the Gregorian calendar – all years which divide exactly by 100 (other than those which divide exactly by 400).

In the Revised Julian calendar – all years which divide exactly by 100 (other than those which give remainder 200 or 600 when divided by 900).

Check the result

Use this table for finding the day of the week without any calculations.

Index			Mon A	Tue B	Wed C	Thu D	Fri E	Sat F	Sun G	Perpetual Gregorian and Julian calendar Use Jan and Feb for leap years Date letter in year row for the letter in century row All the C days are doomsdays																		
Julian century	Gregorian century	Date	**01** 08 15 22 29	**02** 09 16 23 30	**03** 10 17 24 31	**04** 11 18 25 --	**05** 12 19 26 --	**06** 13 20 27 --	**07** 14 21 28 --																		
12 19	16 20	Apr Jul Jan	G	A	B	C	D	E	F	01	07	12	18		29	35	40	46		57	63	68	74		85	91	96
13 20		Sep Dec	F	G	A	B	C	D	E	02		13	19	24	30		41	47	52	58		69	75	80	86		97
14 21	17 21	Jun	E	F	G	A	B	C	D	03	08	14		25	31	36	42		53	59	64	70		81	87	92	98
15 22		Feb Mar Nov	D	E	F	G	A	B	C		09	15	20	26		37	43	48	54		65	71	76	82		93	99
16 23	18 22	Aug Feb	C	D	E	F	G	A	B	04	10		21	27	32	38		49	55	60	66		77	83	88	94	
17 24		May	B	C	D	E	F	G	A	05	11	16	22		33	39	44	50		61	67	72	78		89	95	
18 25	19 23	Jan Oct	A	B	C	D	E	F	G	06		17	23	28	34		45	51	56	62		73	79	84	90		00
[Year/100]		**Gregorian century**	20 16		21 17		22 18		23 19	Year mod 100																	
		Julian century	19 12	20 13	21 14	22 15	23 16	24 17	25 18																		

Examples:

- For common method

December 26, 1893 (GD)

December is in row F and 26 is in column E, so the letter for the date is C located in row F and column E. 93 (year mod 100) is in row D (year row) and the letter C in the year row is located in column G. 18 ([year/100] in the Gregorian century column) is in row C (century row) and the letter in the century row and column G is B, so the day of the week is Tuesday.

October 13, 1307 (JD)

October 13 is a F day. The letter F in the year row (07) is located in column G. The letter in the century row (13) and column G is E, so the day of the week is Friday.

January 1, 2000 (GD)

January 1 corresponds to G, G in the year row (00) corresponds to F in the century row (20), and F corresponds to Saturday.

A pithy formula for the method: *"Date letter (G), letter (G) is in year row (00) for the letter (F) in century row (20), and for the day, the letter (F) become weekday (Saturday)"*.

- For modified domimical letter method

1783, September 18 (GD)

Use 17 (in the Gregorian century row, column C) and 83 (in row C) to find the dominical letter that is E. The letter for September 18 is B, so the day of the week is Thursday.

1676, February 23 (JD, non-OS)

Use 16 (in the Julian century row, column E) and 76 (in row D) to find the dominical letter that is A. February 23 is a "D" day, so the day of the week is Wednesday.

Mathematical algorithms

Gauss's algorithm

In a handwritten note in a collection of astronomical tables, Carl Friedrich Gauss described a method for calculating the day of the week for 1 January in any given year.[5] He never published it. It was finally included in his collected works in 1927.[6]

Gauss' Method was applicable to the Gregorian calendar. He numbered the weekdays from 0 to 6 starting with Sunday. He defined the following operation: The weekday of 1 January in year number A is[5]

$$R(1+5R(A-1,4)+4R(A-1,100)+6R(A-1,400),7)$$

where $R(y,m)$ is the remainder after division of y by m,[6] or y modulo m.

This formula was also converted into graphical and tabular methods for calculating any day of the week by Kraitchik and Schwerdtfeger.[6][7]

Disparate variation

Another variation of the above algorithm likewise works with no lookup tables. A slight disadvantage is the unusual month and year counting convention. The formula is

$$w = \left(d + \lfloor 2.6m - 0.2 \rfloor + y + \left\lfloor \frac{y}{4} \right\rfloor + \left\lfloor \frac{c}{4} \right\rfloor - 2c\right) \bmod 7,$$

where

- Y is the year minus 1 for January or February, and the year for any other month
- y is the last 2 digits of Y
- c is the first 2 digits of Y
- d is the day of the month (1 to 31)
- m is the shifted month (March=1,...,February=12)
- w is the day of week (0=Sunday,...,6=Saturday). If w is negative you have to add 7 to it.

For example, January 1, 2000. (year − 1 for January)

$$\begin{aligned} w &= \left(1 + \lfloor 2.6 \cdot 11 - 0.2 \rfloor + (0-1) + \left\lfloor \frac{0-1}{4} \right\rfloor + \left\lfloor \frac{20}{4} \right\rfloor - 2 \cdot 20\right) \bmod 7 \\ &= (1 + 28 - 1 - 1 + 5 - 40) \bmod 7 \\ &= 6 = \text{Saturday} \end{aligned}$$

$$\begin{aligned} w &= \left(1 + \lfloor 2.6 \cdot 11 - 0.2 \rfloor + (100-1) + \left\lfloor \frac{100-1}{4} \right\rfloor + \left\lfloor \frac{20-1}{4} \right\rfloor - 2 \cdot (20-1)\right) \bmod \\ &= (1 + 28 + 99 + 24 + 4 - 38) \bmod 7 \\ &= 6 = \text{Saturday} \end{aligned}$$

Note: The first is only for a 00 leap year and the second is for any 00 years.

The term $\lfloor 2.6m - 0.2 \rfloor \bmod 7$ gives the values of months: m

Months	m
January	0
February	3
March	2
April	5
May	0
June	3
July	5
August	1
September	4
October	6
November	2
December	4

The term $y + \lfloor y/4 \rfloor \bmod 7$ gives the values of years: y

y mod 28	y
01 07 12 18 --	1
02–13 19 24	2
03 08 14–25	3
-- 09 15 20 26	4
04 10–21 27	5
05 11 16 22 --	6
06–17 23 00	0

The term $\lfloor c/4 \rfloor - 2c \bmod 7$ gives the values of centuries: c

$c \bmod 4$	c
1	5
2	3
3	1
0	0

Now from the general formula: $w = d + m + y + c \bmod 7$; January 1, 2000 can be recalculated as follows:

$w = 1 + 0 + 5 + 0 \bmod 7 = 6 = \text{Saturday}$
$d = 1, m = 0$
$y = 5 (0 - 1 \bmod 28 = 27)$
$c = 0 (20 \bmod 4 = 0)$

$w = 1 + 0 + 4 + 1 \bmod 7 = 6 = \text{Saturday}$
$d = 1, m = 0$
$y = 4 (99 \bmod 28 = 15)$
$c = 1 (20 - 1 \bmod 4 = 3)$

Kraitchik's variation

Kraitchik proposed two methods for calculating the day of the week.[7] One is a graphical method. The other uses a formula that he credits to Gauss on p. 110:

$$w = d + m + c + y \bmod 7,$$

where w is the day of the week (counting upwards from 1 on Sunday instead of 0 in Gauss's version); and d, m, c and y are numbers depending on the day, month, century and year which are tabulated in the "Complete table: Julian and Gregorian calendars" above. Note that the numbers tabulated for y can also be described by the following equation:

$$y = \left(\left\lfloor \frac{s}{4} \right\rfloor + s \right) \bmod 7$$

Where

s is the last two digits of the year (i.e. if the year is 1987, s = 87)
y is ...

So, for example, if you want to find 'y' for the year 1987:

$$\begin{aligned} y &= \left(\left\lfloor \frac{87}{4} \right\rfloor + 87 \right) \bmod 7 \\ &= (21 + 87) \bmod 7 \\ &= 108 \bmod 7 \\ &= 3 \end{aligned}$$

Zeller's algorithm

In Zeller's algorithm, the months are numbered from 3 for March to 14 for February. The year is assumed to begin in March; this means, for example, that January 1995 is to be treated as month 13 of 1994.[8] The formula for the Gregorian calendar is

$$w = \left(d + \left\lfloor \frac{13(m+1)}{5} \right\rfloor + y + \left\lfloor \frac{y}{4} \right\rfloor + \left\lfloor \frac{c}{4} \right\rfloor - 2c \right) \bmod 7,$$

where

- Y is the year minus 1 for January or February, and the year for any other month
- y is the last 2 digits of Y
- c is the first 2 digits of Y
- d is the day of the month (1 to 31)
- m is the shifted month (March=3,...January = 13, February=14)
- w is the day of week (1=Sunday,..0=Saturday)

The only difference is one between Zeller's algorithm (Z) and the Gaussian algorithm (G), that is $Z - G = 1$ = Sunday.

$$\begin{aligned} &(d + \lfloor (m+1)2.6 \rfloor + y + \lfloor y/4 \rfloor + \lfloor c/4 \rfloor - 2c) \bmod 7 - (d + \lfloor 2.6m - 0.2 \rfloor + y + \lfloor y/4 \rfloor + \lfloor c/4 \rfloor - 2c) \bmod 7 \\ &= (\lfloor (m+2+1)2.6 - (2.6m - 0.2) \rfloor) \bmod 7 \text{ (March = 3 in } Z \text{ but March = 1 in } G) \\ &= (\lfloor 2.6m + 7.8 - 2.6m + 0.2 \rfloor) \bmod 7 \\ &= 8 \bmod 7 = 1 \end{aligned}$$

So we can get the values of months from those for the Gaussian algorithm by adding one:

Months	m
January	1
February	4
March	3
April	6
May	1
June	4
July	6
August	2
September	5
October	0
November	3
December	5

Other algorithms

Schwerdtfeger's method

In a partly tabular method by Schwerdtfeger, the year is split into the century and the two digit year within the century. The approach depends on the month. For $m \geq 3$,

$$c = \left\lfloor \frac{y}{100} \right\rfloor \quad \text{and} \quad g = y - 100c,$$

so g is between 0 and 99. For m = 1,2,

$$c = \left\lfloor \frac{y-1}{100} \right\rfloor \quad \text{and} \quad g = y - 1 - 100c.$$

The formula for the day of the week is[6]

$$w = d + e + f + g + \left\lfloor \frac{g}{4} \right\rfloor \mod 7,$$

where the positive modulus is chosen.[6]

The value of e is obtained from the following table:

m	1	2	3	4	5	6	7	8	9	10	11	12
e	0	3	2	5	0	3	5	1	4	6	2	4

The value of f is obtained from the following table, which depends on the calendar. For the Gregorian calendar,[6]

c mod 4	0	1	2	3
f	**0**	**5**	**3**	**1**

For the Julian calendar,[6]

c mod 7	0	1	2	3	4	5	6
f	**5**	**4**	**3**	**2**	**1**	**0**	**6**

Lewis Carroll's method

Charles Lutwidge Dodgson (Lewis Carroll) devised a method resembling a puzzle, yet partly tabular in using the same index numbers for the months as in the "Complete table: Julian and Gregorian calendars" above. He lists the same three adjustments for the first three months of non-leap years, one 7 higher for the last, and gives cryptic instructions for finding the rest; his adjustments for centuries are to be determined using formulas similar to those for the centuries table. Although explicit in asserting that his method also works for Old Style dates, his example reproduced below to determine that "1676, February 23" is a Wednesday only works on a Julian calendar which starts the year on January 1, instead of March 25 as on the "Old Style" Julian calendar.

Algorithm:[9]

> Take the given date in 4 portions, viz. the number of centuries, the number of years over, the month, the day of the month. Compute the following 4 items, adding each, when found, to the total of the previous items. When an item or total exceeds 7, divide by 7, and keep the remainder only.
>
> Century-item: For 'Old Style' (which ended 2 September 1752) subtract from 18. For 'New Style' (which began 14 September 1752) divide by 4, take overplus from 3, multiply remainder by 2.
>
> Year-item: Add together the number of dozens, the overplus, and the number of 4s in the overplus.

> Month-item: If it begins or ends with a vowel, subtract the number, denoting its place in the year, from 10. This, plus its number of days, gives the item for the following month. The item for January is "0"; for February or March, "3"; for December, "12".
>
> Day-item: The total, thus reached, must be corrected, by deducting "1" (first adding 7, if the total be "0"), if the date be January or February in a leap year, remembering that every year, divisible by 4, is a Leap Year, excepting only the century-years, in 'New Style', when the number of centuries is not so divisible (e.g. 1800).
>
> The final result gives the day of the week, "0" meaning Sunday, "1" Monday, and so on.

Examples:[9]

1783, September 18

17, divided by 4, leaves "1" over; 1 from 3 gives "2"; twice 2 is "4". 83 is 6 dozen and 11, giving 17; plus 2 gives 19, i.e. (dividing by 7) "5". Total 9, i.e. "2" The item for August is "8 from 10", i.e. "2"; so, for September, it is "2 plus 31", i.e. "5" Total 7, i.e. "0", which goes out. 18 gives "4". Answer, "Thursday".

1676, February 23

16 from 18 gives "2" 76 is 6 dozen and 4, giving 10; plus 1 gives 11, i.e. "4". Total "6" The item for February is "3". Total 9, i.e. "2" 23 gives "2". Total "4" Correction for Leap Year gives "3". Answer, "Wednesday".

Since 23 February 1676 (counting February as the second month) is, for Carroll, the same day as Gregorian 4 March 1676. Had he not assumed the year to begin on 1 January there would have been a difference in year number – just like George Washington's birthday, which differs between the two calendars.

It is noteworthy that those who have republished Carroll's method have failed to point out his error, most notably Martin Gardner.[10]

In 1752, the British Empire abandoned its use of the Old Style Julian calendar upon adopting the Gregorian calendar, which has become today's standard in most countries of the world. For more background, see Old Style and New Style dates.

Implementation-dependent methods

In the C language expressions below, `y`, `m` and `d` are, respectively, integer variables representing the year (e.g., 1988), month (1-12) and day of the month (1-31).

```
(d+=m<3?y--:y-2,23*m/9+d+4+y/4-y/100+y/400)%7
```

In 1990, Michael Keith and Tom Craver published the foregoing expression that seeks to minimise the number of keystrokes needed to enter a self-contained function for converting a Gregorian date into a numerical day of the week.[11] It preserves neither `y` nor `d`, and returns `0` = Sunday, `1` = Monday, etc.

Shortly afterwards, Hans Lachman streamlined their algorithm for ease of use on low-end devices. As designed originally for four-function calculators, his method needs fewer keypad entries by limiting its range either to A.D. 1905-2099, or to historical Julian dates. It was later modified to convert any Gregorian date, even on an abacus. On Motorola 68000-based devices, there is similarly less need of either processor registers or opcodes, depending on the intended design objective.[12]

Sakamoto's methods

The tabular forerunner to To/ndering's algorithm is embodied in the following K&R C function.[13] Posted by Tomohiko Sakamoto on the comp.lang.c Usenet newsgroup in 1992, it is accurate for any Gregorian date.[14][15]

```
dayofweek(y, m, d)  /* 1 <= m <= 12,  y > 1752 (in the U.K.) */
{
    static int t[] = {0, 3, 2, 5, 0, 3, 5, 1, 4, 6, 2, 4};
    y -= m < 3;
    return (y + y/4 - y/100 + y/400 + t[m-1] + d) % 7;
}
```

The function does not always preserve `y`, and returns `0` = Sunday, `1` = Monday, etc. In contrast, the following expression

```
dow(m,d,y) { y-=m<3; return(y+y/4-y/100+y/400+"-bed=pen+mad."[m]+d)%7; }
```

posted simultaneously by Sakamoto is not only not easily adaptable to other languages, but may even fail if compiled on a computer that encodes characters using other than standard ASCII values (e.g. EBCDIC), or on C compilers that enforce ANSI C compliance (even on code that is still compliant with the original K&R C specification, where omitted type declarations are assumed to be integer). For the latter consideration alone, Sakamoto's more-verbose version might be considered non-portable, as might also that of Keith and Craver.

Rata Die

IBM's Rata Die method requires that one knows the "key day" of the proleptic Gregorian calendar i.e. the day of the week of January 1, AD 1 (its first date). This has to be done to establish the remainder number based on which the day of the week is determined for the latter part of the analysis. By using a given day August 13, 2009 which was a Thursday as a reference, with *Base* and *n* being the number of days and weeks it has been since 01/01/0001 to the given day, respectively and *k* the day into the given week which must be less than 7, *Base* is expressed as

$$Base = 7n + k \qquad \text{(i)}$$

Knowing that a year divisible by 4 or 400 is a leap year while a year divisible by 100 and not 400 is not a leap year, a software program can be written to find the number of days. The following is a translation into C of IBM's method for its REXX programming language.

```
int daystotal (int y, int m, int d)
{
```

```
        static char daytab[2][13] =
        {
                {0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31},
                {0, 31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31}
        };
        int daystotal = d;
        for (int year = 1 ; year <= y ; year++)
        {
                int max_month = ( year < y ? 12 : m-1 );
                int leap = (year%4 == 0);
                if (year%100 == 0 && year%400 != 0)
                        leap = 0;
                for (int month = 1 ; month <= max_month ; month++)
                {
                        daystotal += daytab[leap][month];
                }
        }
        return daystotal;
}
```

It is found that `daystotal` is 733632 from the base date January 1, AD 1. This total number of days can be verified with a simple calculation: There are already 2008 full years since 01/01/0001. The total number of days in 2008 years not counting the leap days is 365 *2008 = 732920 days. Assume that all years divisible by 4 are leap years. Add 2008/4 = 502 to the total; then subtract the 15 leap days because the years which are exactly divisible by 100 but not 400 are not leap. Continue by adding to the new total the number of days in the first seven months of 2009 that have already passed which are 31 + 28 + 31 + 30 + 31 + 30 + 31 = 212 days and the 13 days of August to get *Base* = 732920 + 502 - 20 + 5 + 212 + 13 = 733632.

What this means is that it has been 733632 days since the base date. Substitute the value of *Base* into the above equation (i) to get 733632 = 7 *104804 + 4, n = 104804 and k = 4 which implies that August 13, 2009 is the fourth day into the 104805th week since 01/01/0001. 13 August 2009 is Thursday; therefore, the first day of the week must be Monday, and it is concluded that the first day 01/01/0001 of the calendar is *Monday*. Based on this, the remainder of the ratio *Base/7*, defined above as k, decides what day of the week it is. If k = 0, it's Monday, k = 1, it's Tuesday, etc.[16]

See also

- Doomsday rule
- Julian day#Calculation
- Mental Calculation World Cup (Has a calendar calculation contest)
- Perpetual calendar
- Buddhist calendar

References

1. Brothers, Hardin; Rawson, Tom; Conn, Rex C.; Paul, Matthias; Dye, Charles E.; Georgiev, Luchezar I. (2002-02-27). *4DOS 8.00 online help*.
2. "HP Prime – Portal: Firmware update" (http://hp-prime.de/de/id/61-firmware-update) (in German). Moravia Education. 2015-05-15. Archived (https://web.archive.org/web/20161105001326/http://hp-prime.de/de/id/61-firmware-update) from the original on 2016-11-05. Retrieved 2015-08-28.
3. Paul, Matthias (1997-07-30). *NWDOS-TIPs — Tips & Tricks rund um Novell DOS 7, mit Blick auf undokumentierte Details, Bugs und Workarounds* (http://www.antonis.de/dos/dos-tuts/mpdostip/html/nwdostip.htm) (e-book). *MPDOSTIP* (in German) (3, release 157 ed.). Archived (https://web.archive.org/web/20161104235829/http://www.antonis.de/dos/dos-tuts/mpdostip/html/nwdostip.htm) from the original on 2016-11-04. Retrieved 2014-08-06. NWDOSTIP.TXT is a comprehensive work on Novell DOS 7 and OpenDOS 7.01, including the description of many undocumented features and internals. It is part of the author's yet larger `MPDOSTIP.ZIP` collection maintained up to 2001 and distributed on many sites at the time. The provided link points to a HTML-converted older version of the `NWDOSTIP.TXT` file.
4. Richards, E. G. (1999). *Mapping Time: The Calendar and Its History*. Oxford University Press.
5. Gauss, Carl F. (1981). "Den Wochentag des 1. Januar eines Jahres zu finden. Güldene Zahl. Epakte. Ostergrenze.". *Werke. herausgegeben von der Königlichen Gesellschaft der Wissenschaften zu Göttingen* (2. Nachdruckaufl. ed.). Hildesheim: Georg Olms Verlag. pp. 206–207. ISBN 9783487046433.
6. Schwerdtfeger, Berndt E. (May 7, 2010). "Gauss' calendar formula for the day of the week" (http://berndt-schwerdtfeger.de/wp-content/uploads/pdf/cal.pdf) (pdf) (1.4.26 ed.). Retrieved 23 December 2012.
7. Kraitchik, Maurice (1942). "Chapter five: The calendar". *Mathematical recreations* (2nd rev. [Dover] ed.). Mineola: Dover Publications. pp. 109–116. ISBN 9780486453583.
8. J. R. Stockton (19 March 2010). "The Calendrical Works of Rektor Chr. Zeller : The Day-of-Week and Easter Formulae" (http://www.merlyn.demon.co.uk/zeller-c.htm#Alg). *Merlyn*. Retrieved 19 December 2012.
9. Dodgson, C.L.(Lewis Carroll). (1887). "To find the day of the week for any given date". *Nature*, 31&bbsp;March 1887. Reprinted in *Mapping Time*, pp. 299-301.
10. Martin Gardner. (1996). *The Universe in a Handkerchief: Lewis Carroll's Mathematical Recreations, Games, Puzzles, and Word Plays*, pages 24-26. Springer-Verlag.
11. Michael Keith and Tom Craver. (1990). *The ultimate perpetual calendar?* Journal of Recreational Mathematics, 22:4, pp.280-282.
12. The 4-function Calculator; The Assembly of Motorola 68000 Orphans; The Abacus. gopher://sdf.org/1/users/retroburrowers/TemporalRetrology
13. "Day-of-week algorithm NEEDED!" news:1993Apr20.075917.16920@sm.sony.co.jp
14. Date -> Day of week conversion. Newsgroups: comp.lang.c. https://groups.google.com/d/msg/comp.lang.c/GPA5wwrVnVw/hi2wB0TXGkAJ
15. DOW algorithm. Newsgroups: comp.lang.c. https://groups.google.com/d/msg/comp.lang.c/IvROHSayPEM/fQ7B2J5wn10J (1994)
16. REXX/400 Reference manual (https://www.ibm.com/support/knowledgecenter/ssw_i5_54/books/sc415729.pdf?sc=_latest) page 87 (1997).

- Gauss, Carl F. (1981). "Den Wochentag des 1. Januar eines Jahres zu finden. Güldene Zahl. Epakte. Ostergrenze.". *Werke. herausgegeben von der Königlichen Gesellschaft der Wissenschaften zu Göttingen* (2. Nachdruckaufl. ed.). Hildesheim: Georg Olms Verlag. pp. 206–207. ISBN 9783487046433.
- Hale-Evans, Ron (2006). "Hack #43: Calculate any weekday". *Mind performance hacks* (1st ed.). Beijing: O'Reilly. pp. 164–169. ISBN 9780596101534.
- Thioux, Marc; Stark, David E.; Klaiman, Cheryl; Schultz, Robert T. (2006). "The day of the week when you were born in 700 ms: Calendar computation in an autistic savant". *Journal of Experimental Psychology: Human Perception and Performance*. **32** (5): 1155–1168. doi:10.1037/0096-1523.32.5.1155 (https://doi.org/10.1037%2F0096-1523.32.5.1155).
- Treffert, Darold A. "Why calendar calculating?". *Islands of genius : the bountiful mind of the autistic, acquired, and sudden savant* (1. publ., [repr.]. ed.). London: Jessica Kingsley. pp. 63–66. ISBN 9781849058735.

External links

- Tøndering's algorithm for both Gregorian and Julian calendars (http://www.tondering.dk/claus/cal/chrweek.php#calcdow)
- "Key Day" method used so as to reduce computation & memorization (https://web.archive.org/web/20160321132458/http://www.angelfire.com/my/zelime/calendar.html)
- Compact tabular method for memorisation, also for the Julian calendar (http://katzentier.de/_misc/perpetual_calendar.htm)

- When countries changed from the Julian calendar (http://www.tondering.dk/claus/cal/gregorian.php#country)
- World records for mentally calculating the day of the week in the Gregorian Calendar (http://www.recordholders.org/en/records/dates.html)
- National records for finding Calendar Dates (http://www.recordholders.org/en/list/mental-calculation-rankings.html)
- World Ranking of Memoriad Mental Calendar Dates (http://www.memoriad.com/index.asp?s=kategoriler&b=kategori-detay&kategoriid=dcacea11f97125360e50694fd11c2ae4&lang=EN) (all competitions combined)
- Year searching calendar (http://www5a.biglobe.ne.jp/%257eaccent/calendar/retro.htm)
