1

Calendar Basics

A learned man once asked me regarding the eras used by different nations, and regarding the difference of their roots, that is, the epochs where they begin, and of their branches, that is, the months and years, on which they are based; further regarding the causes which led to such difference, and the famous festivals and commemoration-days for certain times and events, and regarding whatever else one nation practices differently from another. He urged me to give an explanation, the clearest possible, of all this, so as to be easily intelligible to the mind of the reader, and to free him from the necessity of wading through widely scattered books, and of consulting their authors. Now I was quite aware that this was a task difficult to handle, an object not easily to be attained or managed by anyone, who wants to treat it as a matter of logical sequence, regarding which the mind of the student is not agitated by doubt.

Abū-Raiḥān Muḥammad ibn ’Aḥmad al-Bīrūnī:
Al-Āthār al-Bāqiyah ‘an al-Qurūn al-Khāliyah (1000)

Calendrical calculations are ubiquitous. Banks need to calculate interest on a daily basis. Corporations issue paychecks on weekly, biweekly, or monthly schedules. Bills and statements must be generated periodically. Computer operating systems need to switch to and from daylight saving time. Dates of secular and religious holidays must be computed for consideration in planning events. Most of these calculations are not difficult because the rules of our civil calendar (the Gregorian calendar) are straightforward.

Complications begin when we need to know the day of the week on which a given date falls or when various religious holidays based on other calendars occur. These complications lead to difficult programming tasks—not often difficult in an algorithmic sense but difficult because it can be extremely tedious to delve into, for example, the complexities of the Hebrew calendar and its relation to the civil calendar.

The purpose of this book is to present, in a unified, completely algorithmic form, a description of over three dozen calendars and how they relate to one another. Among them are included the present civil calendar (Gregorian); the recent ISO commercial calendar; the old civil calendar (Julian); the ancient Egyptian calendar and its Armenian equivalent; the Coptic and the virtually identical Ethiopic calendars; the Akan (African) calendar, the Islamic (Muslim) calendar (the arithmetical version, one based on calculated lunar observability, and a

Saudi Arabian variant); the modern Persian calendar (both astronomical and arithmetic forms); the Bahá’í calendar, both arithmetic and astronomical forms; the Hebrew (Jewish) calendar, both its present arithmetical form and a speculative observational form; the three Mayan calendars and two virtually identical Aztec calendars; the Pawukon calendar from Bali; the French Revolutionary calendar (both astronomical and arithmetic forms); the Chinese calendar and the virtually identical Japanese, Korean, and Vietnamese calendars; both the old (mean) and new (true) Hindu (Indian) solar and lunisolar calendars; and the Tibetan calendar. Information that is sufficiently detailed to allow computer implementation is difficult to find for most of these calendars because the published material is often inaccessible, ecclesiastically oriented, incomplete, inaccurate, based on extensive tables, overburdened with extraneous material, focused on shortcuts for hand calculation to avoid complicated arithmetic or to check results, or difficult to find in English. Most existing computer programs are proprietary, incomplete, or inaccurate.

The need for such a secular, widely available presentation was made clear to us when we (primarily E.M.R., with contributions by N.D.), in implementing a calendar/diary feature for GNU Emacs [44], found difficulty in gathering and interpreting appropriate source materials that describe the interrelationships among the various calendars and the determination of the dates of holidays. Some of the calendars (Chinese, Japanese, Korean, Vietnamese, Hindu, and Tibetan) had never had full algorithmic descriptions published in English.

The calendar algorithms in this book are presented as mathematical function definitions in standard mathematical format. Appendix A gives the types (ranges and domains) of all functions and constants we use; Appendix B is a cross reference list that gives all dependencies among the functions and constants. In Appendix C we tabulate results of the calendar calculations for 33 sample dates and 44 holidays; this will aid those who develop their own implementations of our calendar functions. To ensure correctness, all calendar functions were automatically typeset[1] directly from the working Common Lisp [46] functions given in Appendix D.[2]

We chose mathematical notation as the vehicle for presentation because of its universality and easy convertibility to any programming language. We have endeavored to simplify the calculations as much as possible without obscuring the intuition. Many of the algorithms we provide are considerably more concise than previously published ones; this is particularly true of the arithmetic Persian, Hebrew, and old Hindu calendars.

We chose Lisp as the vehicle for implementation because it encourages functional programming and has a trivial syntax, nearly self-evident semantics, historical durability, and wide distribution; moreover, Lisp was amenable to translation into ordinary mathematical notation. Except for a few short macros, the code uses

[1] This has meant some sacrifice in the typography of the book; we hope readers sympathize with our decision.

[2] The Lisp code is available through a Cambridge University Press web site `www.cambridge.org/calendricalcalculations` under the terms of the License Agreements and Limited Warranty on page xli. Any errata are available over the World Wide Web at `www.calendarists.com`.

only a very simple, side-effect-free, subset of Lisp. We emphasize that our choice of Lisp should be considered irrelevant to most readers, whom we expect to follow the mathematical notation used in the text, not to delve into the code.

It is not the purpose of this book to give a detailed historical treatment of the material, nor, for that matter, a mathematical one; our goal is to give a logical, thorough, *computational* treatment. Thus, although we give much historical, religious, mathematical, and astronomical data to leaven the discussion, the focus of the presentation is algorithmic. Full historical and religious details as well as the mathematical and astronomical underpinnings of the calendars can be pursued in the references.

In this chapter, we describe the underlying unifying theme of all the calculations along with some useful mathematical facts. The details of specific calendars are presented in subsequent chapters. Historically, the oldest calendars that we consider are the Egyptian (more than 3000 years old) and Babylonian. The Chinese and Mayan calendars also derive from millennia-old calendars. Next are the classical (observation-based) Hebrew, the Julian (the roots of which date back to the ancient Roman empire), the Coptic and Ethiopic (third century), the current Hebrew (fourth century) and the old Hindu (fifth century), followed by the Islamic calendar (seventh century), the newer Hindu calendars (tenth century), the Persian and Tibetan calendars (eleventh century), the Gregorian modification to the Julian calendar (sixteenth century), the French Revolutionary calendar (eighteenth century), and the Bahá'í calendar (nineteenth century). Finally, the International Organization for Standardization's ISO calendar and the arithmetic Persian calendar are of twentieth-century origin.

For expository purposes, however, we present the Gregorian calendar first, in Part I, because it is the most popular calendar currently in use. Because the Julian calendar is so close in substance to the Gregorian, we present it next, followed by the very similar Coptic and Ethiopic calendars. Then we give the ISO calendar and the Icelandic calendar, which are trivial to implement and depend wholly on the Gregorian. The arithmetic Islamic calendar, which because of its simplicity is easy to implement, follows. Next, we present the Hebrew calendar, one of the more complicated and difficult calendars to implement. This is followed by a chapter on the computation of Easter, which is lunisolar like the Hebrew calendar. The ancient Hindu solar and lunisolar calendars are described next; these are simple versions of the modern Hindu solar and lunisolar calendars described in Part II. Next come the Mayan and similar Aztec calendars of historical interest, which have several unique computational aspects. These are followed by the Balinese Pawukon calendar. All the calendars described in Part I are "arithmetical" in that they operate by straightforward integer-based rules. We conclude Part I with a chapter describing the generic arithmetic calendar schemata that apply to many calendars in this part.

In Part II we present calendars that are controlled by irregular astronomical events (or close approximations to them), although these calendars may have an arithmetical component as well. Because the calendars in Part II require some understanding of astronomical events such as solstices, equinoxes, and lunar phases, we begin Part II with a chapter introducing the topics and algorithms that will be needed. We then give the modern Persian calendar in its astronomical and arithmetic forms followed by the Bahá'í calendar, also in two versions: the former Western (arithmetic) version, which depends wholly on the Gregorian, and the new astronomical version. Next we describe the original (astronomical) and modified (arithmetic) forms of the French Revolutionary calendar. All these calendars are computationally simple, provided that certain astronomical values are available. Next we describe some astronomical calendars based on the moon: the Babylonian calendar, a proposed astronomical calculation of Easter, the observational Islamic calendar, and the classical Hebrew calendar. We continue with the Chinese lunisolar calendar and its Japanese, Korean, and Vietnamese versions. We then describe the modern Hindu calendars, which are by far the most complicated of the calendars in this book. We conclude with the Tibetan calendar.

We also provide algorithms for computing holidays based on most of the calendars. In this regard we take the ethnocentric view that our task is to compute the dates of holidays in a given *Gregorian year*; there is clearly little difficulty in finding the dates of, say, Islamic New Year in a given Islamic year! In general we have tried to mention significant holidays on the calendars we cover but have not attempted to be exhaustive and to include all variations. The interested reader can find extensive holiday definitions in [22], [23], and [24].

The selection of calendars we present was chosen with two purposes: to include all common modern calendars and to cover all calendrical techniques. We do not give all variants of the calendars we discuss, but we have given enough details to make any calendar easy to implement.

1.1 Calendar Units and Taxonomy

Teach us to number our days, that we may attain a wise heart.
Psalms 90:12

The sun moves from east to west, and night follows day with predictable regularity. This apparent motion of the sun as viewed by an earthbound observer provided the earliest time-keeping standard for humankind. The day is, accordingly, the basic unit of time underlying all calendars, but various calendars use different conventions to structure days into larger units: weeks, months, years, and cycles of years. Different calendars also begin their day at different times: the French Revolutionary day, for example, begins at true (apparent) midnight; the Islamic, Bahá'í, and Hebrew days begin at sunset; the Hindu day begins at sunrise. The various definitions of *day* are surveyed in Section 14.3.

The purpose of a calendar is to give a name to each day. The mathematically simplest naming convention would be to assign an integer to each day; fixing day 1 would determine the whole calendar. The Babylonians had such a day count (in base 60). Such *diurnal* calendars are used by astronomers (see Section 14.3) and by calendarists (see, for example, Section 10.1); we use a day numbering in this book as an intermediate device for converting from one calendar to another (see the following section). Day-numbering schemes can be complicated by using a mixed-radix system [28] in which the day number is given as a sequence of numbers or names (see Section 1.10). The Mayans, for example, utilized such a method (see Section 11.1).

Calendar day names are generally distinct, but this is not always the case. For example, the day of the week is a calendar, in a trivial sense, with infinitely many days having the same day name (see Section 1.12). A 7-day week is almost universal today. In many cultures, the days of the week were named after the seven "wandering stars" (or after the gods associated with those heavenly bodies), namely, the sun, the moon, and the five planets visible to the naked eye—Mercury, Venus, Mars, Jupiter, and Saturn. In some languages—Arabic, Lithuanian, Portuguese, Ukrainian, and Hebrew are examples—some or all of the days of the week are numbered, not named. In the Armenian calendar, for example, the days of the week are named as follows [22, vol. 3, p. 70]:

Sunday	Kiraki (or Miashabathi)
Monday	Erkoushabathi
Tuesday	Erekhshabathi
Wednesday	Chorekhshabathi
Thursday	Hingshabathi
Friday	Urbath (or Vetsshabathi)
Saturday	Shabath

"Shabath" means "day of rest" (from the Hebrew), "Miashabathi" means the first day following the day of rest, "Erkoushabathi" is the second day following the day of rest, and so on. The Armenian Christian church later renamed "Vetsshabathi" as "Urbath," meaning "to get ready for the day of rest." Subsequently, they declared the first day of the week as "Kiraki" or "the Lord's day."

Other cycles of days have also been used, including 4-day weeks (in the Congo), 5-day weeks (in other parts of Africa, in Bali, and in Russia in 1929), 6-day weeks (Japan), 8-day weeks (in yet other parts of Africa and in the Roman Republic), and 10-day weeks (in ancient Egypt and in France at the end of the eighteenth century; see page 282). The mathematics of cycles of days are described in Section 1.12. Many calendars repeat after one or more years. In one of the Mayan calendars (see Section 11.2), and in many preliterate societies, day names are recycled every year. The Chinese calendar uses a repeating 60-name scheme for days and years, and at one time used it to name months.

An interesting variation in some calendars is the use of two or more cycles running simultaneously. For example, the Mayan tzolkin calendar (Section 11.2) combines a cycle of 13 names with a cycle of 20 numbers. The Chinese cycle of 60 names for years is actually composed of cycles of length 10 and 12 (see Section 19.4). The Balinese calendar takes this idea to an extreme; see Chapter 12. The mathematics of simultaneous cycles is described in Section 1.13.

The notions of "month" and "year," like the day, were originally based on observations of heavenly phenomena, namely the waxing and waning of the moon, and the cycle of seasons, respectively. The lunar cycle formed the basis for the palaeolithic marking of time (see [32] and [13]), and many calendars today begin each month with the new moon, when the crescent moon first becomes visible (as in the Hebrew calendar of classical times and in the religious calendar of the Muslims today—see Sections 14.9 and 18.4); others begin the month at full moon (in northern India, for example)—see page 160. For calendars in which the month begins with the observed new moon, beginning the day at sunset is natural.

Over the course of history, many different schemes have been devised for determining the start of the year, usually based on the solar cycle.[3] Some are astronomical, beginning at the autumnal or spring equinox, or at the winter or summer solstice. Solstices are more readily observable; either one can note when the midday shadow of a gnomon is longest (at the winter solstice in the northern hemisphere) or shortest (at the summer solstice) or one can note the point in time when the sun rises or sets the farthest south during the course of the year (which is the start of winter in the northern hemisphere) or the farthest north (the start of summer). The ancient Egyptians began their year with the *heliacal rising* of Sirius—that is, on the day when the Dog Star Sirius (the brightest fixed star in the sky) can first be seen in the morning after a period during which the sun's proximity to Sirius makes the latter invisible to the naked eye. The Pleiades ("Seven Sisters") were used by the Maoris and other peoples for the same purpose. Various other natural phenomena such as harvests or the rutting seasons of certain animals have been used among North American tribes [9] to establish the onset of a new year. And not just humans use such phenomena: the lunar cycle determines life cycle events for certain corals [7], birds [43], and monkeys [30]. It has also been suggested [10] that the pink "skylight" on the crown of the head of leatherback turtles serves to allow them to determine when in late summer the lengths of day and night are equal (taking refraction into account), at which point foraging turtles turn south.

Calendars have, of necessity, an integral number of days in a month and an integral number of months in a year. However, these astronomical periods—day, month, and year—are incommensurate: their periods do not form integral multiples of one another. The lunar month is about $29\frac{1}{2}$ days long, and the solar year is about $365\frac{1}{4}$ days long (Chapter 14 has precise definitions and values). How exactly one coordinates these time periods and the accuracy with which they approximate their astronomical values is what differentiates one calendar from another.

Broadly speaking, solar calendars—including the Egyptian, Armenian, Persian, Gregorian, Julian, Coptic, Ethiopic, ISO, French Revolutionary, and Bahá'í —are based on the yearly solar cycle, whereas lunar and lunisolar calendars—such as the Islamic, Hebrew, Hindu, Tibetan, and Chinese—take the monthly lunar cycle as their basic building block. Most solar calendars are divided into months, but these months are divorced from the lunar events; they are sometimes related to the movement of the sun through the 12 signs of the zodiac, notably in the Hindu solar calendars (see Chapter 20).

Because observational methods suffer from vagaries of weather and chance, they have for the most part been supplanted by calculations. The simplest option

[3] It has been claimed that in equatorial regions, where the tropical year is not of paramount agricultural importance, arbitrary year lengths are more prevalent, such as are found in the 210-day Balinese Pawukon calendar (Chapter 12) and the 260-day Mayan divine year (Section 11.2).

is to approximate the length of the year, of the month, or of both. Originally, the Babylonian solar calendar was based on 12 months of 30 days each (see [26]), overestimating the length of the month and underestimating the year; see Figure 1.1. Such a calendar is easy to calculate, but each month begins at a slightly later lunar phase than the previous, and the seasons move forward slowly through the year. The ancient Egyptian calendar achieved greater accuracy by having 12 months of 30 days plus 5 extra days—Egyptian mythology includes a tale of how the calendar came to have these five extra days [3]. Conversions for this calendar are illustrated in Section 1.11. To achieve better correlation with the motion of the moon, one can instead alternate months of 29 and 30 days. Twelve such months, however, amount to 354 days—more than 11 days short of the solar year.

Almost every calendar in this book and virtually all other calendars incorporate a notion of "leap" year to deal with the cumulative error caused by approximating a year by an integral number of days and months.[4] Solar calendars add a day every few years to keep up with the astronomical year. The calculations are simplest when the leap years are evenly distributed and the numbers involved are small; for instance, the Julian, Coptic, and Ethiopic calendars add 1 day every 4 years. Formulas for the evenly distributed case, such as when one has a leap year every fourth or fifth year, are derived in Section 1.14. The old Hindu solar calendar (Chapter 10) follows such a pattern; the arithmetical Persian calendar almost does (see Chapter 15). The Gregorian calendar, however, uses an uneven distribution of leap years but a relatively easy-to-remember rule (see Chapter 2). The modified French Revolutionary calendar (Chapter 17) included an even more accurate but uneven rule.

Most lunar calendars incorporate the notion of a year. Purely lunar calendars may approximate the solar year with 12 lunar months (as does the Islamic), though this is about 11 days short of the astronomical year. Lunisolar calendars invariably alternate 12- and 13-month years, according either to some fixed rule (as in the Hebrew calendar) or to an astronomically determined pattern (Chinese and modern Hindu). The so-called *Metonic cycle* is based on the observation that 19 solar years contain almost exactly 235 lunar months. This correspondence, named after the Athenian astronomer Meton (who published it in 432 B.C.E.) and known much earlier to ancient Babylonian and Chinese astronomers, makes a relatively simple and accurate fixed solar/lunar calendar feasible. The $235 = 12 \times 12 + 7 \times 13$ months in the cycle are divided into 12 years of 12 months and 7 leap years of 13 months. The Metonic cycle is used in the Hebrew calendar (Chapter 8) and for the calculation of Easter (Chapter 9).

The more precise the mean year, the larger the underlying constants must be. For example the Metonic cycle is currently accurate to within 6.5 minutes a year, but other lunisolar cycles are conceivable: 3 solar years are approximately 37 lunar months with an error of 1 day per year; 8 years are approximately 99 months with an error of 5 hours per year; 11 years are approximately 136 months with

[4] See [6, pp. 677–678] for a discussion of the etymology of the term "leap."

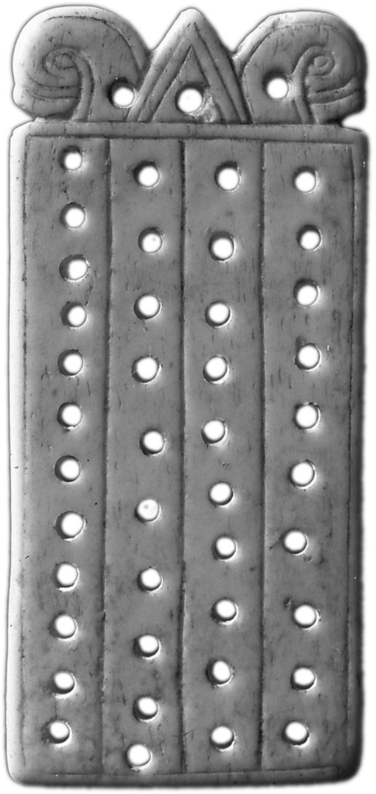


Figure 1.1 A small (6 × 2.7 cm) bone plaque found in Tel ‘Aroer, an Iron Age II (8th–6th century B.C.E.) caravan town in the Negev, Israel. It is conjectured to be a calendar counter: a peg could move daily through the 30 holes in the three right-hand columns of 10 holes each, while another peg moved monthly through the 12 holes in the first column. It could have been used either as a schematic 360-day calendar or as a lunar calendar, in which case some months would end after 29 days [14]. (Reproduced courtesy of the Hebrew Union College, Jerusalem.)

an error of 3 hours per year; and 334 years are 4131 months with an error of 7.27 seconds per year. The old Hindu calendar is even more accurate, comprising 2226389 months in a cycle of 180000 years (see Chapter 10) to which the leap-year formulas of Section 1.14 apply, and errs by less than 8 seconds per year.

The placement of leap years must make a trade-off between two conflicting requirements: small constants defining a simple leap year rule of limited accuracy versus greater accuracy at the expense of larger constants, as the examples in the last paragraph suggest. The choice of the constants is aided by taking the continued fraction (see [27]) of the desired ratio and choosing among the convergents (where to stop in evaluating the fraction). In the case of lunisolar calendars, the solar year is about 365.24244 days, while the lunar month is about 29.53059 days, so we write

$$\frac{365.24244}{29.53059} = 12 + \cfrac{1}{2 + \cfrac{1}{1 + \cfrac{1}{2 + \cfrac{1}{1 + \cfrac{1}{1 + \cfrac{1}{18 + \cfrac{1}{3 + \cdots}}}}}}}.$$

By choosing further and further stopping points, we get better and better approximations to the true ratio. For example,

$$12 + \cfrac{1}{2 + \cfrac{1}{1}} = \frac{37}{3}$$

while

$$12 + \cfrac{1}{2 + \cfrac{1}{1 + \cfrac{1}{2}}} = \frac{99}{8}$$

$$12 + \cfrac{1}{2 + \cfrac{1}{1 + \cfrac{1}{2 + \cfrac{1}{1}}}} = \frac{136}{11}$$

and

$$12 + \cfrac{1}{2 + \cfrac{1}{1 + \cfrac{1}{2 + \cfrac{1}{1 + \cfrac{1}{1}}}}} = \frac{235}{19}$$

These are the ratios of the previous paragraph. Not all approximations must come from continued fractions, however: 84 years are approximately 1039 lunar months with an error of 33 minutes per year, but this is not one of the convergents.

Continued fractions can be used to get approximations to solar calendars too. The number of days per solar year is about 365.242177, which we can write as

$$365.242177 = 365 + \cfrac{1}{4 + \cfrac{1}{7 + \cfrac{1}{1 + \cfrac{1}{2 + \cfrac{1}{1 + \cfrac{1}{5 + \cdots}}}}}}$$

The convergents are $1/4$ (the basis of the Julian, Coptic, and Ethiopic calendars), $7/29$, $8/33$ (possibly used for an ancient Persian calendar), $23/95$, and $31/128$ (used in our implementation of the arithmetical Persian calendar—see Chapter 15).

Table 1.1 gives for comparison the values for the mean length of the year and month as implemented by the various solar, lunar, and lunisolar calendars in this book. The true values change over time, as explained in Chapter 14.

1.2 Fixed Day Numbers

May those who calculate a fixed date … perish.[5]
Morris Braude: *Conscience on Trial: Three Public Religious Disputations between Christians and Jews in the Thirteenth and Fifteenth Centuries* (1952)

Over the centuries, human beings have devised an enormous variety of methods for specifying dates.[6] None are ideal computationally, however, because all have idiosyncrasies resulting from attempts to coordinate a convenient human labeling with lunar and solar phenomena.

For a computer implementation, the easiest way to reckon time is simply to count days. Fix an arbitrary starting point as day 1 and specify a date by giving a day number relative to that starting point; a single 32-bit integer allows the representation of more than 11.7 million years. Such a reckoning of time is, evidently, extremely awkward for human beings and is not in common use, except among astronomers, who use *julian day numbers* to specify dates (see Section 1.5), and calendarists, who use them to facilitate conversion among calendars—see equation

[5] This is a loose translation of a famous dictum from the Babylonian Talmud *Sanhedrin* 97b. The omitted words from Braude's translation (p. 112 of his book) are "for the coming of the Messiah." The exact Talmudic wording is "Blasted be the bones of those who calculate the end." Braude was the uncle of E.M.R.'s mother-in-law, a connection we discovered long after the first edition of this book was published!

[6] The best reference is still Ginzel's monumental three-volume work [16], in German. An exceptional survey can be found in the *Encyclopædia of Religion and Ethics* [22, vol. III, pp. 61–141 and vol. V, pp. 835–894]. Useful modern summaries are [6], [12], [40], and [45]; [6] and [40] have extensive bibliographies. The incomparable tables of Schram [41] are the best available for converting dates by hand, whereas those in Parise [36] are best avoided because of an embarrassingly large numbers of errors.

Table 1.1 Length in days of mean years on solar and lunisolar calendars and length in days of mean lunar months on lunar and lunisolar calendars. The year length is given in italics when the sidereal, rather than the tropical, value is intended. These may be compared with the astronomical values given for various millennial points—*in solar days current at the indicated time*. No values are given here for the Chinese, astronomical Persian, observational Islamic, astronomical Bahá'í, and (original) French Revolutionary calendars because they are self-adjusting. The implicit Mayan values come from other values that they knew; see the footnote on page 171.

	Calendar (or year)	Mean year (days)	Mean month (days)
Calendrical	Egyptian	365	
	Mayan (haab)	365	
	Mayan (implicit)	365.24204	29.530864
	Julian/Coptic/Ethiopic	365.25	
	Hebrew	365.24682	29.530594
	Easter (Orthodox)	365.25	29.530851
	Islamic (Arithmetic)		29.530556
	Hindu (*Arya*)	*365.25868*	29.530582
	Hindu (*Sūrya*)	*365.25876*	29.530588
	Tibetan (*Phugpa*)	*365.27065*	29.530587
	Gregorian	365.2425	
	Easter (Gregorian)	365.2425	29.530587
	French (Arithmetic)	365.24225	
	Persian (Arithmetic)	365.24220	
Astronomical	Year −1000	365.24257	29.530598
	Year 0	365.24244	29.530595
	Year 1000	365.24231	29.530591
	Year 2000	365.24218	29.530588
	Year 3000	365.24204	29.530584

(10.2) for the ancient Indian method and for a more modern example see [41]. The day-count can be augmented by a fractional part to give a specific moment during the day; for example, noon on day i, where i is an integer, would be specified by $i + 0.5$.

We have chosen midnight at the onset of Monday, January 1, 1 (Gregorian) as our fixed date 1, which we abbreviate as R.D. 1,[7] and we count forward day-by-day from there. Of course, this is anachronistic because there was no year 1 on the Gregorian calendar—the Gregorian calendar was devised only in the sixteenth century—thus by January 1, 1 (Gregorian) we mean the day we get if we extrapolate backwards from the present; this day turns out to be Monday, January 3, 1 C.E.[8] (Julian); this too is anachronistic. We call an R.D. that has a fractional part giving the time of day a "moment."

The date Monday, January 1, 1 (Gregorian), though arbitrarily chosen as our starting point, has a desirable characteristic: It is early enough that almost all dates

[7] *Rata Die*, or fixed date. We are indebted to Howard Jacobson for this coinage.

[8] Common Era, or A.D.

of interest are represented by positive integers of moderate size. We have been careful to write our functions in such a way that all dependencies on this choice of starting point are explicit. To change the origin of the calculations we have provided a function

$$\mathbf{rd}\,(t) \overset{\text{def}}{=} t - \mathit{epoch} \tag{1.1}$$

where

$$\mathit{epoch} = 0$$

which defines the origin, *epoch*. Changing this definition to *epoch* = 710347, for example, would make Monday, November 12, 1945 (Gregorian) the starting point.

We should thus think of the passage of time in terms of a sequence of days numbered . . . , −2, −1, 0, 1, 2, 3, . . . , which the various human-oriented calendars label differently. For example, R.D. 710347 is called

- Monday, November 12, 1945, on the Gregorian calendar.
- October 30, 1945 C.E., on the Julian calendar, which would be called *ante diem III Kalendas Novembris* in the Roman nomenclature.
- Julian day number 2431772 (at noon).
- Modified julian day number 31771.
- Month 7, day 10, 2694, on the ancient Egyptian calendar.
- Trē 5, 1395, on the Armenian calendar.
- Fodwo on the Akan calendar.
- Day 1 of week 46 of year 1945, on the ISO calendar.
- Mánudagur of week 3 of winter of year 1945, on the Icelandic calendar.
- Athōr 3, 1662, Era of the Martyrs, on the Coptic calendar (until sunset).
- H̲edār 3, 1938, on the Ethiopic calendar (until sunset).
- Dhu al-Ḥijja 6, 1364, on the arithmetic and observational Islamic calendars (until sunset).
- Kislev 7, 5706, on the Hebrew calendar, but Kislev 6, 5706, on the observational Hebrew calendar (until sunset in both cases).
- 12.16.11.16.9 in the Mayan long count.
- 7 Zac on the Mayan haab calendar.
- 11 Muluc on the Mayan tzolkin calendar.
- Panquetzaliztli 1 on the Aztec xihuitl calendar.
- 11 Atl on the Aztec tonalpohualli calendar.

- Luang, Pepet, Pasah, Sri, Pon, Tungleh, Coma of Gumbreg, Ludra, Urungan, Pati on the Balinese Pawukon calendar.
- Tulā 29, 5046, Kali Yuga Era (elapsed) on the old Hindu solar calendar (after sunrise).
- Day 8 in the bright half of Kārtika, 5046, Kali Yuga Era (elapsed) on the old Hindu lunisolar calendar (after sunrise).
- Abān 21, 1324, on the modern Persian arithmetic and astronomical calendars.
- The day of Asmā', of the month of Qudrat, of the year Abad, of the sixth Vahid, of the first Kull-i-Shay on the Bahá'í calendar (until sunset).
- Décade III, Primidi de Brumaire de l'Année 154 de la République on the arithmetical and astronomical French Revolutionary calendars.
- Day 8 of the tenth month in the year Yĭyŏu on the Chinese calendar.
- Kārtika 27, 1867, Śaka Era (elapsed) on the modern and astronomical Hindu solar calendars (after sunrise).
- Day 7 in the bright half of Kārtika, 2002, Vikrama Era (elapsed) on the modern and astronomical Hindu lunisolar calendars (after sunrise).
- Arakhsamna 6, 2256 on the Babylonian calendar.
- Day 7 of the tenth month, 2072 on the Tibetan calendar.

All that is required for calendrical conversion is to be able to convert each calendar to and from this fixed-date R.D. calendar. Because some calendars begin their day at midnight and others at sunrise or sunset,

$$\boxed{\text{We fix the time of day at which conversions are performed to be noon.}}$$

Figure 1.2 shows the relationships of various calendar's times for the beginning and ending of days.

In subsequent chapters we give functions to do the conversions for the various calendars. For each calendar x, we write a function **fixed-from-*x***(*x-date*) to convert a given date *x-date* on that calendar to the corresponding R.D. date, and a function ***x*-from-fixed**(*date*) to do the inverse operation, taking the R.D. *date* and computing its representation in calendar x. One direction is often much simpler to calculate than the other, and occasionally we resort to considering a range of possible dates on calendar x, searching for the one that converts to the given R.D. date (see Section 1.8). To convert from calendar x to calendar y, one need only compose these two functions:

$$\textbf{\textit{y}-from-\textit{x}}(\textit{x-date}) \overset{\text{def}}{=}$$
$$\qquad \textbf{\textit{y}-from-fixed}(\textbf{fixed-from-\textit{x}}(\textit{x-date}))$$

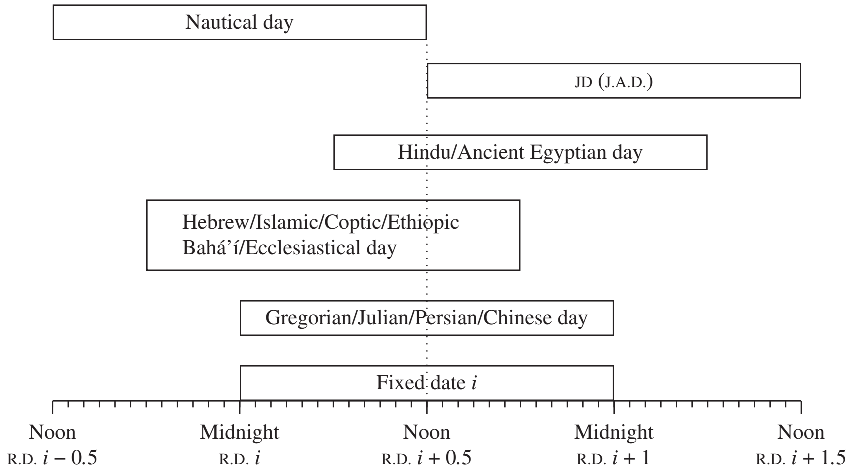


Figure 1.2 Meaning of a "day" in various calendars. Conversion from a date on a calendar to an R.D. date is done as of noon; the rectangles indicate the day of a calendar that gets converted to R.D. i. For example, the Hebrew date corresponding to fixed date i is the Hebrew day that begins at sunset of the evening of fixed date $i - 1$ and ends at sunset of the evening of fixed date i. Similarly, the Hindu date corresponding to fixed date i is the Hindu day that begins at sunrise in the morning of fixed i and ends at sunrise of the morning of fixed date $i + 1$. The JD corresponding to fixed date i begins at noon of fixed date i and ends at noon of fixed date $i + 1$.

Each calendar has an *epoch*, the first day of the first year of that calendar (see Section 1.4). We assign an integer R.D. date to an epoch, even if the calendar in question begins its days at a time other than midnight. Such assignment is done as per Figure 1.2. All the algorithms given in this book give mathematically sensible results for dates prior to a calendar's epoch.

1.3 Negative Years

Quis enim potest intelligere dies et tempora et annos, nisi per numerum? [Who can understand days and seasons and years, save by number?]

Attributed to the Venerable Bede: *De Computo Dialogus*

We cannot avoid dealing with dates before the common era. For example, the Hebrew calendar begins at sunset on Sunday, September 6, −3760 (Gregorian); scholarly literature is replete with such statements. Thus, to aid the reader, we now explain how years before the common era are conventionally handled. This convention is often a source of confusion, even among professional historians.

It is computationally convenient, and mathematically sensible, to label years with the sequence of integers $\ldots, -3, -2, -1, 0, 1, 2, 3, \ldots$, so that year 0 precedes year 1; we do this when extrapolating backward on the Gregorian calendar, so the same leap-year rule based on divisibility by 4, 100, and 400 will apply (see Chapter 2). However, on the Julian calendar it is customary to refer to the year preceding 1 C.E. as 1 B.C.E.,[9] counting it as a leap year in accordance with the every-fourth-year leap-year rule of the Julian calendar. Thus, the beginning of the Hebrew calendar can alternatively be referred to as sunset on October 6, 3761 B.C.E. (Julian). To highlight this asymmetry, in the rest of this book we append "B.C.E." *only* to Julian calendar years, reserving the minus sign for Gregorian calendar years.[10] Care must therefore be taken when doing arithmetic with year numbers. For $n \geqslant 0$, the rough present-day alignment of the Julian and Gregorian calendars gives

$$\text{year } -n \text{ (Gregorian)} \approx \text{year } (n+1) \text{ B.C.E. (Julian)}$$

and, for $n \geqslant 1$,

$$\text{year } n \text{ (Gregorian)} \approx \text{year } n \text{ C.E. (Julian)}$$

1.4 Epochs

My son, take occasional lessons on calendrical calculations from R. Aaron for it is a necessary wisdom.

Judah ibn Tibbon: *Ethical Will* (circa 1180)

Every calendar has an *epoch* or starting date. This date is virtually never the date the calendar was adopted but rather a hypothetical starting point for the first day. For example, the Gregorian calendar was devised and adopted in the sixteenth century,

[9] Before the Common Era, or B.C.

[10] Historically scholars have mixed the notations, using negative years for the Julian calendar and the B.C.E./C.E. (B.C./A.D.) notation for Gregorian years, so one must be cautious in interpreting what a particular author means. The ambiguity has led to confusion and errors.

but its epoch is January 1, 1. Because days begin at different hours on different calendars, we follow the convention that a calendar's epoch is the onset of the civil day (the mean solar day, beginning at midnight) containing the first noon (see Figure 1.2). For example, we take midnight at the onset of September 7, −3760 (Gregorian) as the epoch of the Hebrew calendar, which was codified in the fourth century, though the first Hebrew day began at sunset the preceding evening. For calendars like the Akan or Balinese Pawukon, in which cycles are unnumbered, the choice of epoch is arbitrary; the first day of any cycle can be used.

Table 1.2 gives the epochs of the calendars discussed in this book. With the exception of the Julian day number, we express the epochs of all the calendars as integer R.D. dates, that is, the integer R.D. day number at *noon* of the first day of the calendar (again, see Figure 1.2). Thus, the epoch for the Gregorian calendar is R.D. 1, and that for the Hebrew calendar is R.D. −1373427. Using this form of calendar epochs is convenient because

$$\text{R.D. } d = (d - \text{calendar epoch}) \text{ days since the start of that calendar}$$

For example,

$$\begin{aligned} 710347 - (\text{Hebrew calendar epoch}) &= 710347 - (-1373427) \\ &= 2083774 \end{aligned}$$

and hence

$$\text{R.D. } 710347 = 2083774 \text{ days since the start of the Hebrew calendar}$$

Because, for the most part, our formulas depend on the number of days elapsed on some calendar, we often use the expression (d − calendar epoch) in our calendar formulas.

For many calendars, including the Gregorian, the same calendar rules were used with different eras and different month names at different times and in different places. In Taiwan, for instance, the Gregorian calendar is used with an era beginning with the founding of the republic in 1912. An often-encountered era from the second century B.C.E. until recent times—used with many calendars—was the Era of Alexander, or the Seleucid Era, in which year 1 corresponds to 312 B.C.E. In general, we will avoid describing the details of trivial variants of calendars.

1.5 Julian Day Numbers

Iulianam vocauimus: quia ad annum Iulianum dumtaxat accommodata est. [*I have called this the Julian period because it is fitted to the Julian year.*]

Joseph Justus Scaliger: *De Emendatione Temporum,*
end of introduction to Book V (1583)

Astronomers in recent centuries have avoided the confusing situation of date references on different calendars, each with its idiosyncrasies, by specifying moments in time by giving them in "julian days" or JD (sometimes "julian astronomical days" or J.A.D.). The "Julian period," published in 1583 by Joseph Justus Scaliger, was

Table 1.2 Epochs for various calendars.

Calendar	Epoch (R.D.)	Equivalents
Julian day number	−1721424.5	Noon, November 24, −4713 (Gregorian)
		Noon, January 1, 4713 B.C.E. (Julian)
Hebrew	−1373427	September 7, −3760 (Gregorian)
		October 7, 3761 B.C.E. (Julian)
Mayan	−1137142	August 11, −3113 (Gregorian)
		September 6, 3114 B.C.E. (Julian)
Hindu (Kali Yuga)	−1132959	January 23, −3101 (Gregorian)
		February 18, 3102 B.C.E. (Julian)
Chinese	−963099	February 15, −2636 (Gregorian)
		March 8, 2637 B.C.E. (Julian)
Samaritan	−598573	March 3, −1638 (Gregorian)
		March 15, 1639 B.C.E. (Julian)
Egyptian	−272787	February 18, −746 (Gregorian)
		February 26, 747 B.C.E. (Julian)
Babylonian	−113502	March 29, −310 (Gregorian)
		April 3, 311 B.C.E. (Julian)
Tibetan	−46410	December 7, −127 (Gregorian)
		December 10, 128 B.C.E. (Julian)
Julian	−1	December 30, 0 (Gregorian)
		January 1, 1 C.E. (Julian)
Gregorian	1	January 1, 1 (Gregorian)
		January 3, 1 C.E. (Julian)
ISO	1	January 1, 1 (Gregorian)
		January 3, 1 C.E. (Julian)
Akan	37	February 6, 1 (Gregorian)
		February 8, 1 C.E. (Julian)
Ethiopic	2796	August 27, 8 (Gregorian)
		August 29, 8 C.E. (Julian)
Coptic	103605	August 29, 284 (Gregorian)
		August 29, 284 C.E. (Julian)
Armenian	201443	July 13, 552 (Gregorian)
		July 11, 552 C.E. (Julian)
Persian	226896	March 22, 622 (Gregorian)
		March 19, 622 C.E. (Julian)
Islamic	227015	July 19, 622 (Gregorian)
		July 16, 622 C.E. (Julian)
Zoroastrian	230638	June 19, 632 (Gregorian)
		June 16, 632 C.E. (Julian)
French Revolutionary	654415	September 22, 1792 (Gregorian)
		September 11, 1792 C.E. (Julian)
Bahá'í	673222	March 21, 1844 (Gregorian)
		March 9, 1844 C.E. (Julian)
Modified julian day number	678576	November 17, 1858 (Gregorian)
		November 5, 1858 C.E. (Julian)
Unix	719163	January 1, 1970 (Gregorian)
		December 19, 1969 C.E. (Julian)

originally a counting of *years* in a repeating pattern 7980 years long, starting from 4713 B.C.E. (Julian). It is often claimed ([1, page 431], for example) that Scaliger named the period after his father, the Renaissance physician Julius Cæsar Scaliger, but this claim is not borne out by examination of Scaliger's great work, *De Emendatione Temporum*, from which the section quote above is taken. Grafton [17] gives a full history of *De Emendatione Temporum*. The details of the derivation for the value 7980 are given in [39]; the roots of the 7980-year cycle are much earlier than Scaliger, however, dating back to the twelfth century [38]. In the mid-nineteenth century, Herschel [25, page 532] adapted the system into a strict counting of *days* backward and forward from

$$\begin{aligned}\text{JD } 0 &= \text{noon on Monday, January 1, 4713 B.C.E. (Julian)}\\ &= \text{noon on Monday, November 24, } -4713 \text{ (Gregorian)}\end{aligned}$$

A fractional part of a julian[11] date gives the fraction of a day beyond noon; switching dates at noon makes sense for astronomers who work through the night. In this system, for example, sunset on the first day of the Hebrew calendar occurred at about JD 347997.25 (local time), which is 1/4 of a day after noon. The literature on the Mayan calendar commonly specifies the beginning of the calendar in julian days. Because noon of R.D. 0 is JD 1721425, it follows that

$$\text{JD } n = \text{Noon on R.D. } (n - 1721425)$$

In other words,

$$\text{Midnight at the onset of R.D. } d = \text{JD } (d + 1721424.5) \tag{1.2}$$

We do not use julian days directly, as suggested in [21], because we want our days to begin at civil midnight. *We also use fractional days when we need to calculate with time, but we begin each day at midnight.*

To distinguish clearly between the Julian calendar and julian days in our functions, we use the abbreviation "jd" instead of "julian." We have

$$\textbf{jd-epoch} \overset{\text{def}}{=} \text{R.D. } -1721424.5 \tag{1.3}$$

$$\textbf{moment-from-jd}\,(jd) \overset{\text{def}}{=} jd + \textbf{jd-epoch} \tag{1.4}$$

$$\textbf{jd-from-moment}\,(t) \overset{\text{def}}{=} t - \textbf{jd-epoch} \tag{1.5}$$

where *jd* can be a fraction representing time as well as date. As used by historians, julian day numbers are defined as *jd* + 0.5 (see [33, vol. 3, p. 1064], for example). Thus our function **fixed-from-jd** gives the R.D. date intended by historians when they refer to julian dates.[12]

[11] We use lowercase here to avoid any confusion between a julian day number and a date on the Julian calendar.

[12] "Note that Julian date is sometimes used as a synonym for day of year, but this is not correct usage. Day of year ranges from 1 to 365 (366 for leap years) whereas Julian dates are a continuous count of days" [2].

For dates near the present, the julian day number is inconvenient because at least 7-digit accuracy is needed. Astronomers occasionally use *modified julian day numbers*, or MJD, defined as

Modified julian day number = julian day number − 2400000.5

which counts days from midnight, Wednesday, November 17, 1858 (Gregorian). This is equivalent to defining

$$\textbf{mjd-epoch} \overset{\text{def}}{=} \text{R.D. } 678576 \tag{1.6}$$

$$\textbf{fixed-from-mjd}\,(mjd) \overset{\text{def}}{=} mjd + \textbf{mjd-epoch} \tag{1.7}$$

$$\textbf{mjd-from-fixed}\,(date) \overset{\text{def}}{=} date - \textbf{mjd-epoch} \tag{1.8}$$

We do not use modified julian days directly because we want positive numbers for dates within recent history.

1.6 Unix Time Representation

Unix is simple and coherent, but it takes a genius (or at any rate, a programmer) to understand and appreciate the simplicity.

Dennis Ritchie: "Unix: A Dialectic," *The Australian UNIX systems User Group Newsletter* (1989)

In the Unix operating system, and its derivatives, time is measured in seconds after midnight Universal Time (see Section 14.2) on January 1, 1970, ignoring leap seconds. Unix time simply counts 60 seconds per minute, 60 minutes per hour, 24 hours per day. Hence we define

$$\textbf{unix-epoch} \overset{\text{def}}{=} \text{R.D. } 719163 \tag{1.9}$$

The equivalent R.D. moment of a Unix time is easily computed,

$$\textbf{moment-from-unix}\,(s) \overset{\text{def}}{=} \textbf{unix-epoch} + \frac{s}{24 \times 60 \times 60} \tag{1.10}$$

as is the Unix time of an R.D. moment,

$$\textbf{unix-from-moment}\,(t) \overset{\text{def}}{=} 24 \times 60^2 \times (t - \textbf{unix-epoch}) \tag{1.11}$$

On computers that represent integers with signed 32-bit words, only moments from 20:45:53 December 13, 1908 until 3:14:07 on January 19, 2038 can be represented. With 64 bits, the range is greater than ±292 billion years from the present.

1.7 Mathematical Notation

The best notation is no notation.
Paul Halmos: *How to Write Mathematics (1970)*

We use the following mathematical notation (see [18]) when describing the calendar calculations: The *floor function*, $\lfloor x \rfloor$, gives the largest integer less than or equal to x. For example, $\lfloor \pi \rfloor = 3$. The similar *ceiling function*, $\lceil x \rceil$, gives the smallest integer greater than or equal to x. For example, $\lceil \pi \rceil = 4$ and $\lceil -\pi \rceil = -3$. In general, $\lceil x \rceil = -\lfloor -x \rfloor$, so for example $\lfloor -\pi \rfloor = -4$. For integers n, $\lfloor n \rfloor = \lceil n \rceil = n$. Using the floor function, we can convert a moment into an R.D. date by

$$\textbf{fixed-from-moment}\,(t) \overset{\text{def}}{=} \lfloor t \rfloor \tag{1.12}$$

Similarly, we can convert a moment given in julian days to an R.D. date, with no fractional part by

$$\textbf{fixed-from-jd}\,(jd) \overset{\text{def}}{=} \lfloor \textbf{moment-from-jd}\,(jd) \rfloor \tag{1.13}$$

The inverse is simply the same as **jd-from-moment**:

$$\textbf{jd-from-fixed}\,(\mathit{date}) \overset{\text{def}}{=} \textbf{jd-from-moment}\,(\mathit{date}) \tag{1.14}$$

Occasionally we need to *round* values to the nearest integer. We can express this using the floor function as

$$\text{round}(x) \overset{\text{def}}{=} \lfloor x + 0.5 \rfloor \tag{1.15}$$

which is either $\lfloor x \rfloor$ or $\lceil x \rceil$.

We use a single large left-hand brace to indicate a conditional expression, one whose value depends on two or more conditions. For example, in

$$x \overset{\text{def}}{=} \begin{cases} \textbf{value}_1 & \textbf{condition}_1 \\ \textbf{value}_2 & \textbf{condition}_2 \\ \textbf{value}_3 & \textbf{otherwise} \end{cases}$$

the conditions are examined in order, from top down. Thus the value of x is $\textbf{value}_1$ if $\textbf{condition}_1$ is true, $\textbf{value}_2$ if $\textbf{condition}_1$ is false but $\textbf{condition}_2$ is true, and $\textbf{value}_3$ if both conditions are false. Note that if both conditions are true, the sequential evaluation of them means that the value of the expression is $\textbf{value}_1$. As a simple example of such a conditional expression we define the sign function

$$\text{sign}\,(y) \overset{\text{def}}{=} \begin{cases} -1 & \textbf{if } y < 0 \\ 1 & \textbf{if } y > 0 \\ 0 & \textbf{otherwise} \end{cases} \tag{1.16}$$

The *remainder*, or *modulus, function*, $x \bmod y$, is defined for $y \neq 0$ as

$$x \bmod y \overset{\text{def}}{=} x - y\lfloor x/y \rfloor \tag{1.17}$$

which is the remainder when x is divided by y (x and y need not be integers). For example, $9 \bmod 5 = 4$, $-9 \bmod 5 = 1$, $9 \bmod -5 = -1$, and $-9 \bmod -5 = -4$. Definition (1.17) makes sense for any nonzero value of y; for example, $5/3 \bmod 3/4 = 1/6$. In particular, when $y = 1$, $x \bmod 1$ is the *fractional part* of x, allowing us to obtain the time of day as a fraction from a moment by

$$\textbf{time-from-moment}\,(t) \overset{\text{def}}{=} t \bmod 1 \tag{1.18}$$

In programming languages (including C, C++, and Pascal) without a built-in remainder function that works for nonintegers, the definition given in (1.17) must be used instead.

There are five important consequences of definition (1.17). First,

$$\text{if } y > 0 \text{ then } x \bmod y \geqslant 0$$

for all x, even for negative values of x; we use this property throughout our calculations. Care must thus be exercised in implementing our algorithms in computer languages such as C and C++ in which the mod operator `%` may have (x `%` y) < 0 for $x < 0$, $y > 0$. It follows from (1.17) that

$$(-x) \bmod y = y - (x \bmod y)$$

for $y > 0$ and $x \not\equiv 0 \pmod{y}$. The third consequence is that the definition of the mod function implies that for $y \neq 0$ and $z \neq 0$,

$$a = (x \bmod y) \quad \text{if and only if} \quad az = (xz \bmod yz) \tag{1.19}$$

Setting $z = -1$ then gives

$$(-x) \bmod (-y) = -(x \bmod y)$$

as a special case. Fourth,

$$x - (x \bmod y) \text{ is always an integer multiple of } y \tag{1.20}$$

Finally, the fifth consequence is a generalization of the first consequence: for $y \neq 0$,

$$0 \leqslant \text{sign}(y) \times (x \bmod y) < |y| \tag{1.21}$$

The mod function allows us to define two other important functions, the *greatest common divisor* and the *least common multiple*. The greatest common divisor of two positive integers, x and y is defined as

$$\gcd(x, y) \overset{\text{def}}{=} \begin{cases} x & \textbf{if } y = 0 \\ \gcd(y, x \bmod y) & \textbf{otherwise} \end{cases} \tag{1.22}$$

and their least common multiple as

$$\text{lcm}(x, y) \overset{\text{def}}{=} \frac{xy}{\gcd(x, y)} \tag{1.23}$$

We make extensive use of an extension of the standard modulus notation of [11] which takes an *interval* as the modulus, rather than a divisor; we use the "double-dot" notation ". ." for interval ranges, as suggested by C. A. R. Hoare and L. Ramshaw; see [18, p. 73]). Then we can write the *interval modulus* as

$$x \bmod [a \,..\, b)$$

which shifts a real-valued x into the half-open (meaning one end point, indicated by the right-hand parenthesis, is not included) real interval $[a \,..\, b)$ by adding a multiple of the length $b - a$. We define

$$x \bmod [a \,..\, b) \overset{\text{def}}{=} \begin{cases} x & \textbf{if } a = b \\ a + (x - a) \bmod (b - a) & \textbf{otherwise} \end{cases} \tag{1.24}$$

or, equivalently,

$$x \bmod [a \,..\, b) \overset{\text{def}}{=} \begin{cases} x & \textbf{if } a = b \\ x - (b - a)\left\lfloor \dfrac{x - a}{b - a} \right\rfloor & \textbf{otherwise} \end{cases}$$

This definition works perfectly well when the interval is given backward, that is, $a > b$, yielding a modulus in the half-open interval $(b \,..\, a]$. It follows that $a \leqslant x \bmod [a \,..\, b) < b$ if $a < b$, but $b < x \bmod [a \,..\, b) \leqslant a$ when $a > b$. This notation conveniently supports addition and multiplication:

$$c + (x \bmod [a \,..\, b)) = (c + x) \bmod [c + a \,..\, c + b) \tag{1.25}$$

$$c \times (x \bmod [a \,..\, b)) = (c \times x) \bmod [c \times a \,..\, c \times b) \tag{1.26}$$

On a few occasions (especially in Chapter 20) we will want to recenter the remainder; this is easy with the interval modulus notation. For example, to convert an angle α to the range $[-180° \,..\, 180°)$, we just write

$$\alpha \bmod [-180 \,..\, 180) \tag{1.27}$$

We frequently need a special case of the interval modulus in order to shift an integer into the range $[1 \,..\, b + 1) = [1 \,..\, b]$, where b is also an integer. We call this the *adjusted remainder function*, $x \bmod [1 \,..\, b]$, and it can be defined for $b \neq 0$ as

$$x \bmod [1 \,..\, b] \overset{\text{def}}{=} \begin{cases} b & \textbf{if } x \bmod b = 0 \\ x \bmod b & \textbf{otherwise} \end{cases} \tag{1.28}$$

This function is equivalently defined by

$$x \bmod [1 \,..\, b] = b + x \bmod (-b) \tag{1.29}$$

$$x \bmod [1 \,..\, b] = [(x - 1) \bmod b] + 1$$

$$x \bmod [1 \,..\, b] = x - b(\lceil x/b \rceil - 1)$$

Though this definition works equally well for real numbers, we will need this adjusted remainder only for integers x and b.

Finally, we use a special summation operator,

$$\sum_{i \geq k}^{p(i)} f(i) = f(k) + f(k+1) + \cdots$$

whose value is that obtained when $f(i)$ is summed for all $i = k, k+1, \ldots$, continuing only as long as the condition $p(i)$ holds. This operator can be defined recursively as follows:

$$\sum_{i \geq k}^{p(i)} f(i) \stackrel{\text{def}}{=} \begin{cases} f(k) + \sum_{i \geq k+1}^{p(i)} f(i) & \textbf{if } p(k) \\ 0 & \textbf{otherwise} \end{cases} \tag{1.30}$$

Thus, the sum is 0 when $p(k)$ is false. The analogous product operator

$$\prod_{i \geq k}^{p(i)} f(i) = f(k) \times f(k+1) \times \cdots$$

can be defined as follows:

$$\prod_{i \geq k}^{p(i)} f(i) \stackrel{\text{def}}{=} \begin{cases} f(k) \times \prod_{i \geq k+1}^{p(i)} f(i) & \textbf{if } p(k) \\ 1 & \textbf{otherwise} \end{cases} \tag{1.31}$$

1.8 Search

... as two grains of wheat hid in two bushels of chaff: you shall seek all day ere you find them, and when you have them, they are not worth the search.

William Shakespeare: *Merchant of Venice,* Act I, scene i (1600)

In many calendar computations, it is easy to compute an approximate date and easy to check whether a date in question is correct, but difficult to compute the correct date directly. In such cases, we compute a lower bound d_0 on the possible date and then perform a linear search, day by day, until the correct date d is reached. For that purpose we use the operator

$$\underset{d \geqslant d_0}{\textbf{MIN}} \{\psi(d)\}$$

which searches for the smallest d in the sequence d_0, $d_0 + 1$, $d_0 + 2$, ... such that the condition ψ holds true for d. In other words, using the symbol "¬" for logical negation, we have $\neg\psi(d_0)$, $\neg\psi(d_0 + 1)$, $\neg\psi(d_0 + 2)$, ..., $\neg\psi(d - 1)$, but $\psi(d)$. The operator **MIN** is defined formally as

$$\underset{d \geqslant k}{\textbf{MIN}} \{\psi(d)\} \stackrel{\text{def}}{=} \begin{cases} k & \textbf{if } \psi(k) \\ \underset{d \geqslant k+1}{\textbf{MIN}} \{\psi(d)\} & \textbf{otherwise} \end{cases} \tag{1.32}$$

It is undefined and does not terminate if the predicate $\psi(d)$ does not become true eventually, so care must be taken in its use.

Occasionally, we search the sequence for the day prior to the first d' such that $\neg\psi(d')$ and use instead

$$\underset{d \geqslant d_0}{\mathbf{MAX}} \{\psi(d)\}$$

With this operator, we have $\psi(d_0)$, $\psi(d_0 + 1)$, $\psi(d_0 + 2), \ldots, \psi(d)$, but $\neg\psi(d + 1)$. **MAX** is undefined and its computation does not terminate if the predicate $\psi(d)$ does not eventually become false. This **MAX** operator is defined formally as

$$\underset{d \geqslant k}{\mathbf{MAX}} \{\psi(d)\} \overset{\text{def}}{=} \begin{cases} k - 1 & \textbf{if } \neg\psi(k) \\ \underset{d \geqslant k+1}{\mathbf{MAX}} \{\psi(d)\} & \textbf{otherwise} \end{cases} \tag{1.33}$$

When $\psi(d_0)$ is already false, $\underset{d \geqslant d_0}{\mathbf{MAX}} \{\psi(d)\} = d_0 - 1$.

In the absence of explicit methods for calculating the inverse of a function f, that is, for calculating a value x such that $f(x) = y$ given the value of y, we will need to search an interval $[a \mathinner{..} b]$ for the point x, $a \leqslant x \leqslant b$, when $f(x) = y$. To express such a calculation, we write

$$f^{-1}(y, [a \mathinner{..} b])$$

In other words, $f^{-1}(y, [a \mathinner{..} b])$ is a value x_0 such that there is some $x \in [a \mathinner{..} b]$ for which $f(x) = y$, precisely, and $|x - x_0| < \varepsilon$, where $\varepsilon > 0$ is some small tolerance within which the result is acceptable.

When the function in question f is *increasing*, binary search [42, Section 3.2] can be an effective means of inverting f. Hence

$$\boxed{\underset{\xi \in [\mu..\nu]}{\overset{\varphi(l,u)}{\mathbf{MIN}}} \{\psi(\xi)\} \approx y \quad \text{means} \quad \begin{array}{c} \mu \leqslant l < y < u \leqslant \nu, \\ \varphi(l, u), \neg\psi(l), \text{ and } \psi(u) \end{array}} \tag{1.34}$$

That is, we search for a y satisfying the definiens (defining expression) under the assumption that the region $[\mu \mathinner{..} \nu]$ can be split into two intervals $[\mu \mathinner{..} x)$, up to but not including x, and $[x \mathinner{..} \nu]$, such that ψ is false throughout the former and true in the latter. Then y must be close enough to x that it lies in an interval $[l \mathinner{..} u]$, sandwiching x, small enough to satisfy the test $\varphi(l, u)$. If ψ is true of the midpoint ξ, then we go left and let the new upper bound ν be ξ. On the other hand, if ψ is false, then we go right and let the new lower bound μ be ξ. This process continues until the interval $[\mu \mathinner{..} \nu]$ is small enough that φ is true, at which point the midpoint is returned. At each stage of the search, $\psi(\mu)$ is false and $\psi(\nu)$ is true. We implement the definition using a straightforward binary search of the interval $[\mu \mathinner{..} \nu]$ that behaves as follows:

$$\underset{\xi \in [\mu..\nu]}{\overset{\varphi}{\mathbf{MIN}}} \{\psi(\xi)\} \overset{\text{def}}{=} \begin{cases} x & \textbf{if } \varphi\,(\mu, \nu) \\ \underset{\xi \in [\mu..x]}{\overset{\varphi}{\mathbf{MIN}}} \{\psi(\xi)\} & \textbf{if } \psi\,(x) \\ \underset{\xi \in [x..\nu]}{\overset{\varphi}{\mathbf{MIN}}} \{\psi(\xi)\} & \textbf{otherwise} \end{cases} \tag{1.35}$$

where

$$x = \frac{\mu + \nu}{2}$$

To determine the time of astronomical events, for example equinoxes or solstices, we need to invert astronomical functions, such as the celestial longitude of the sun in Section 14.4. These astronomical functions f take moments and return values in the range $[0° \,..\, 360°)$. We use binary search to invert such functions. That is, given a desired value y, here in degrees, we search for a moment x, within some given range, such that $f(x) = y$. Since the search interval will always be relatively small, we sidestep the discontinuity at $360° = 0°$ by searching for the first moment at which $f(x) - y$ modulo 360 becomes tiny. Using binary search, this is achieved as follows:

$$\boxed{f^{-1}(y, [a \,..\, b]) \overset{\text{def}}{=} \underset{x \in [a..b]}{\overset{u-l<10^{-5}}{\mathbf{MIN}}} \{(f(x) - y) \bmod 360 < 180°\}} \tag{1.36}$$

The process terminates when the moment is ascertained within one ten-thousandth of a day, which is less than one second.

1.9 Dates and Lists

The list could surely go on, and there is nothing more wonderful than a list, instrument of wondrous hypotyposis.

Umberto Eco: *The Name of the Rose* (1983)

We represent calendar dates by fixed-length records with fields (components)—in descending order of significance—which we draw as a sequence of boxes, usually having the form

year	*month*	*day*

in which *year*, *month*, and *day* are all integers. We use boldface subscripts to select fields; for example, if

$d =$

1945	11	12

then $d_{\mathbf{day}} = 12$. The fields of dates differ for some calendars; we explain those particular forms in the individual discussions and use analogously named indices for extracting individual components.

We also have occasion to use lists of dates or of other items. Our use of lists requires manipulations such as forming lists, selecting elements from a list, or concatenating lists. We use the following notation for lists in our calendar functions.

- Angle brackets indicate list construction, that is, the formation of a list from individual components. For example, $\langle 1945, 11, 12 \rangle$ is a list of the three components 1945, 11, and 12, respectively.
- Subscripts in square brackets indicate list element selection, with the indices of the elements 0-based. Thus if $b = \langle 1945, 11, 12 \rangle$ then $b_{[0]}$ is 1945, $b_{[1]}$ is 11, and $b_{[2]}$ is 12.

- Empty angle brackets, $\langle\,\rangle$, indicate the list with no elements.
- Double bars indicate the concatenation of lists, and thus

  $$\langle 1945\rangle \parallel \langle 11, 12\rangle = \langle 1945, 11, 12\rangle$$

  The identity under concatenation is $\langle\,\rangle$; that is, the concatenation of $\langle\,\rangle$ with any list leaves the list unchanged.

A recursive traversal of a list can be used, for example, to convert each element of a list of moments into a fixed date, using the earlier function **fixed-from-moment** (1.12):

$$\textbf{list-of-fixed-from-moments}\,(\ell) \overset{\text{def}}{=} \tag{1.37}$$

$$\begin{cases} \langle\,\rangle & \textbf{if } \ell = \langle\,\rangle \\ \left\langle \textbf{fixed-from-moment}\left(\ell_{[0]}\right)\right\rangle \parallel \textbf{list-of-fixed-from-moments}\left(\ell_{[1..]}\right) \\ \qquad \textbf{otherwise} \end{cases}$$

The empty list $\langle\,\rangle$ gives an empty list, while a nonempty list is composed of its first element, $\ell_{[0]}$, converted to a fixed date, followed by the converted remainder of the list.

We use the notation $[a\ ..\ b]$ to represent an interval of time beginning at moment a and ending at moment b (inclusive). To indicate that a given R.D. moment t is within a range $[a\ ..\ b]$, in other words, that $a \leqslant t \leqslant b$, we write

$$t \in [a\ ..\ b]$$

The same notation is used for an occurrence within half-open intervals:

$$t \in [a\ ..\ b) \overset{\text{def}}{=} a \leqslant t < b \tag{1.38}$$

The determination of non-Gregorian holidays occurring in a given Gregorian year can require generating a list of the R.D. dates of those holidays over some longer interval and then scanning the list to filter out those not in the given Gregorian year. To cull a list of dates for those that occur in a given (half-open) range, we use the following recursive process:

$$\ell \cap [a\ ..\ b) \overset{\text{def}}{=} \tag{1.39}$$

$$\begin{cases} \langle\,\rangle & \textbf{if } \ell = \langle\,\rangle \\ \left\langle \ell_{[0]} \right\rangle \parallel r & \textbf{if } \ell_{[0]} \in [a\ ..\ b] \\ r & \textbf{otherwise} \end{cases}$$

where

$$r = \ell_{[1..]} \cap [a\ ..\ b)$$

To collect all occurrences of events, such as holidays, in an interval of time, like a Gregorian year, we write a generic function to find the first occurrence on or after a given moment of the pth moment in a c-day cycle, $0 \leqslant p < c$, and then recursively find the remaining occurrences:

$$\textbf{positions-in-range}\,(p, c, \Delta, [a \,..\, b)) \overset{\text{def}}{=} \tag{1.40}$$
$$\begin{cases} \langle\,\rangle & \textbf{if } \mathit{date} \geqslant b \\ \langle \mathit{date} \rangle \parallel \textbf{positions-in-range}\,(p, c, \Delta, [a + c \,..\, b)) & \textbf{otherwise} \end{cases}$$

where

$$\mathit{date} = (p - \Delta) \bmod [a \,..\, a + c)$$

Here Δ is congruent modulo c to the position of R.D. moment 0 in the repeating cycle, and *date* is the first occurrence of position p in the interval *range*, computed using the interval modulus function (1.24).

1.10 Mixed-Radix Notations

It is interesting to note that nearly all MIX *programs can be expressed without knowing whether binary or decimal notation is being used—even when we are doing calculations involving multiple-precision arithmetic. Thus we find that the choice of radix does not significantly influence computer programming.*

Donald E. Knuth, *The Art of Computer Programming, vol. 2, Seminumerical Algorithms* (1998)

Mixed-radix notation, in which the radix can differ from position to position, is a generalization of ordinary positional notation, such as decimal, binary, and sexagesimal (base 60). We represent numbers in mixed-radix notation as lists, following the notation in [28, Section 4.1]: for example, 4 weeks, 1 day, 12 hours, 44 minutes, and 2.88 seconds is written $\langle 4, 1, 12, 44, 2.88 \rangle$ in base $\langle 7; 24, 60, 60 \rangle$, where the semicolon separates the integer part of the base from the fractional part. Each (rational) number b_i in the base determines the range $[0 \,..\, b_i)$ of values that may appear in position i. There will always be one more element in the number than in the base, for the most significant position; this position has no maximal value. The most significant position of a date given in mixed-radix notation can be negative. Also, the least significant position need not be a whole number, as in the above example, $\langle 4, 1, 12, 44, 2.88 \rangle$.

To evaluate the mixed-radix number

$$a = \langle a_0, a_1, \ldots, a_n \rangle$$

written in base

$$b = \langle b_1, \ldots, b_k; b_{k+1}, \ldots, b_n \rangle$$

we define the symbol $a \overset{\text{rad}}{\longleftarrow} b$ by

$$a \overset{\text{rad}}{\longleftarrow} b \overset{\text{def}}{=} \sum_{i=0}^{k} \left(a_i \times \prod_{j=i+1}^{k} b_j \right) + \sum_{i=k+1}^{n} \left(a_i \Big/ \prod_{j=k+1}^{i} b_j \right) \tag{1.41}$$

In the opposite direction, to convert a number x into base $b = \langle b_1, \ldots, b_k; b_{k+1}, \ldots, b_n \rangle$ we define the symbol $x \xrightarrow{\text{rad}} b$ by

$$x \xrightarrow{\text{rad}} b \overset{\text{def}}{=} \langle a_0, a_1, \ldots, a_n \rangle \tag{1.42}$$

where

$$a_0 = \left\lfloor x \middle/ \prod_{j=1}^{k} b_j \right\rfloor$$

$$a_i = \left\lfloor x \middle/ \prod_{j=i+1}^{k} b_j \right\rfloor \bmod b_i, \quad i = 1, \ldots, k$$

$$a_i = \left\lfloor x \times \prod_{j=k+1}^{i} \right\rfloor \bmod b_i, \quad i = k+1, \ldots, n-1$$

$$a_n = \left(x \times \prod_{j=k+1}^{n} \right) \bmod b_n$$

We use mixed-radix numbers for the Gregorian (Section 2.3), Icelandic (Chapter 6), Hebrew (Section 8.3), and Mayan (Section 11.1) calendars.

When referring to durations of time, we use base $\langle ; 24, 60, 60 \rangle$ and indicate positions with superscripts; for example,

$$i^{\text{d}} j^{\text{h}} k^{\text{m}} l^{\text{s}} = \langle i, j, k, l \rangle = i + \frac{j + \frac{k + \frac{l}{60}}{60}}{24} \text{ days}$$

is i days, j hours, k minutes, and l seconds. Thus, a fifth of a day is $0^{\text{d}}4^{\text{h}}48^{\text{m}}0^{\text{s}}$, or $4^{\text{h}}48^{\text{m}}$, for short, and $10^{\text{d}} - 4^{\text{h}}48^{\text{m}} = 9.8^{\text{d}} = 9^{\text{d}}19^{\text{h}}12^{\text{m}}$ is represented as the list $\langle 9, 19, 12 \rangle$.

To convert the usual clock time—also expressed in base $\langle ; 24, 60, 60 \rangle$ but without any whole days—into a fraction of a day, we use

$$\textbf{time-from-clock}\,(hms) \overset{\text{def}}{=} \frac{1}{24} \times hms \xleftarrow{\text{rad}} \langle ; 24, 60, 60 \rangle \tag{1.43}$$

The following function converts the fractional part of R.D. moment t into hours, minutes, and seconds on a 24-hour clock, taking midnight as $\langle 0, 0, 0 \rangle$, that is, 0:00:00 hours:

$$\textbf{clock-from-moment}\,(t) \overset{\text{def}}{=} \left(t \xrightarrow{\text{rad}} \langle ; 24, 60, 60 \rangle \right)_{[1..]} \tag{1.44}$$

The first component of $t \xrightarrow{\text{rad}} \langle ;\, 24, 60, 60\rangle$, which contains whole days, is removed. To round to the nearest second, apply **clock-from-moment** to

$$\frac{\textbf{round}(t \times 24 \times 60 \times 60)}{24 \times 60 \times 60}$$

instead of t.

Angles can be described in terms of a list of degrees, arc minutes, and arc seconds in base $\langle ;\, 60, 60\rangle$:

$$\langle d, m, s\rangle = d^\circ m' s'' = d + \frac{m + \frac{s}{60}}{60} \text{ degrees}$$

Given angle α as a real number of degrees, we can convert it to a list of degrees, arc minutes, and arc seconds with

$$\textbf{angle-from-degrees}\,(\alpha) \overset{\text{def}}{=} \tag{1.45}$$

$$\begin{cases} dms & \textbf{if } \alpha \geqslant 0 \\ \left\langle -dms_{[0]},\ -dms_{[1]},\ -dms_{[2]} \right\rangle & \textbf{otherwise} \end{cases}$$

where

$$dms = |\alpha| \xrightarrow{\text{rad}} \langle ;\, 60, 60\rangle$$

Negative angles (such as southern latitudes) are given as a list of negative numbers of degrees, arc minutes, and arc seconds. The function $|\alpha|$ returns the absolute value of the angle α.

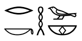
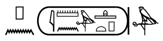
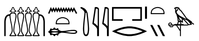
1.11 A Simple Calendar

This calendar is, indeed, the only intelligent calendar which ever existed in human history.

Otto Neugebauer: *The Exact Sciences in Antiquity* (1952)

A simple solar calendar with 365 days each year and no leap-year rule was in use in Egypt for millennia before the adoption of the Julian calendar in the third century C.E. and was also used in Babylon and Persia.[13] The development of this calendar is discussed in [37] ([8] has extensive source documents); it served as the canonical calendar for astronomers until the sixteenth century, and it is to this calendar that Neugebauer refers in the preceding quotation. Each month had 30 days, except for the last 5 days of the year, called *epagomenæ*, which were considered an unlucky period and which we can treat as a short thirteenth month. The month names with their hieroglyphs, according to [4], were:

[13] The ancient Egyptians are also believed to have used a lunar calendar with months beginning the first morning of invisibility of the old moon [37].

	Middle Kingdom	New Kingdom	
(1) Thoth			30 days
(2) Phaophi			30 days
(3) Athyr			30 days
(4) Choiak			30 days
(5) Tybi			30 days
(6) Mechir			30 days
(7) Phamenoth			30 days
(8) Pharmuthi			30 days
(9) Pachon			30 days
(10) Payni			30 days
(11) Epiphi			30 days
(12) Mesori			30 days
(13) (Unnamed)			5 days

Variants of these month names are still used in the Coptic calendar (see Section 4.1). Days began at dawn. We use this calendar as a simple example of calendar conversion functions. Our calendar functions always use numbers to represent months; we provide tables of names, when known, for each calendar.

The epoch chosen by the famous Alexandrian astronomer Ptolemy, author of the *Almagest*, for this calendar, and called the *Nabonassar Era* after the Chaldean king Nabonassar, is given by [33] as JD 1448638:

$$\textbf{egyptian-epoch} \stackrel{\text{def}}{=} \textbf{fixed-from-jd}\,(1448638) \tag{1.46}$$

which corresponds to R.D. −272787, or February 26, 747 B.C.E. (Julian). Because all years have fixed length, converting Egyptian dates into fixed dates is trivial:

$$\textbf{fixed-from-egyptian}\left(\begin{array}{|c|c|c|}\hline \textit{year} & \textit{month} & \textit{day} \\ \hline\end{array}\right) \stackrel{\text{def}}{=} \tag{1.47}$$

$$\textbf{egyptian-epoch} + 365 \times (\textit{year} - 1) + 30 \times (\textit{month} - 1) + \textit{day} - 1$$

The short last month causes no problem, because we only count the number of days in prior months.

In the astronomical code and elsewhere we use a vector notation within summations. For example,

$$\textbf{alt-fixed-from-egyptian}\left(\widetilde{e\text{-}date}\right) \overset{\text{def}}{=} \qquad (1.48)$$

$$\textbf{egyptian-epoch} + \sum \widetilde{a} \times \left(\widetilde{e\text{-}date} - 1\right)$$

where

$$\widetilde{a} = \langle 365, 30, 1 \rangle$$

performs the same calculation as **fixed-from-egyptian**. Each of the three components of the Egyptian date $\widetilde{e\text{-}date}$ is decremented by 1 and then multiplied by the corresponding element of $\widetilde{a} = \langle 365, 30, 1 \rangle$ to give the total number of elapsed days since the epoch.

For the inverse, converting fixed dates to Egyptian dates, we make straightforward use of the floor and mod functions:

$$\textbf{egyptian-from-fixed}\,(date) \overset{\text{def}}{=} \boxed{\,year \mid month \mid day\,} \qquad (1.49)$$

where

$$\begin{aligned}
days &= date - \textbf{egyptian-epoch}\\
year &= \left\lfloor \frac{days}{365} \right\rfloor + 1\\
month &= \left\lfloor \frac{1}{30} \times (days \bmod 365) \right\rfloor + 1\\
day &= days - 365 \times (year - 1) - 30 \times (month - 1) + 1
\end{aligned}$$

The rules of the Armenian calendar were identical to those of the Egyptian; the only difference is the epoch (see [31]):

$$\textbf{armenian-epoch} \overset{\text{def}}{=} \text{R.D. } 201443 \qquad (1.50)$$

which corresponds to July 11, 552 C.E. (Julian). To convert R.D. dates to and from the Armenian calendar, we simply adjust by the difference in epochs:

$$\textbf{fixed-from-armenian}\left(\boxed{\,year \mid month \mid day\,}\right) \overset{\text{def}}{=} \qquad (1.51)$$

$$\textbf{armenian-epoch} + \textbf{fixed-from-egyptian}\left(\boxed{\,year \mid month \mid day\,}\right)$$
$$- \textbf{egyptian-epoch}$$

In the other direction we have

$$\textbf{armenian-from-fixed}\,(date) \overset{\text{def}}{=} \qquad (1.52)$$

$$\textbf{egyptian-from-fixed}\,(date + \textbf{egyptian-epoch} - \textbf{armenian-epoch})$$

The 12 Armenian months were called

(1) Nawasardi	(7) Mehekani
(2) Hor̄i	(8) Areg
(3) Sahmi	(9) Ahekani
(4) Trē	(10) Mareri
(5) K‘aloch	(11) Margach
(6) Arach	(12) Hrotich

and the epagomenæ were called *aweleach*.

The Zoroastrian calendar has an identical structure to that of the ancient Egyptian calendar, but with a different epoch (see Table 1.2) and different month names. In the past, the Persians used individual names for each of the days of the month; these Persian names were

(1) Hormuz	هرمز	(16) Mehr	مهر
(2) Bahman	بهمن	(17) Sorūsh	سروش
(3) Ordībehesht	اردیبهشت	(18) Rashn	رشن
(4) Shahrīvar	شهریور	(19) Farvardīn	فروردین
(5) Esfandārmud	اسفندارمذ	(20) Bahrām	بهرام
(6) Xordād	خرداذ	(21) Rām	رام
(7) Mordād	مرداذ	(22) Bād	باد
(8) Diy be Āzar	دی باذر	(23) Diy be Dīn	دی بدین
(9) Āzar	آزر	(24) Dīn	دین
(10) Ābān	آبان	(25) Ard	ارد
(11) Xor	خور	(26) Ashtād	اشتاذ
(12) Māh	ماه	(27) Asmān	اسمان
(13) Tīr	تیر	(28) Zāmyād	زامیاد
(14) Goosh	گوش	(29) Māresfand	مارسفند
(15) Diy be Mehr	دی بمهر	(30) Anīrān	انیران

and the epagomenæ were sometimes named:

(1) Ahnad	اهند
(2) Ashnad	اشند
(3) Esfandārmud	اسفندارمد
(4) Axshatar	اخشتر
(5) Behesht	بهشت

The Mandean calendar also follows the same structure, but its epagomenæ lie between the eighth and ninth months [47].

1.12 Cycles of Days

And day by day I'll do this heavy task.
Shakespeare: *Titus Andronicus*, Act V, scene ii (1594)

Because R.D. 1 is a Monday, determining the day of the week amounts to taking the R.D. date modulo 7: 0 is Sunday, 1 is Monday, and so forth. We define the seven constants

$$\textbf{sunday} \overset{\text{def}}{=} 0 \tag{1.53}$$

$$\textbf{monday} \overset{\text{def}}{=} 1 \tag{1.54}$$

$$\textbf{tuesday} \overset{\text{def}}{=} 2 \tag{1.55}$$

$$\textbf{wednesday} \overset{\text{def}}{=} 3 \tag{1.56}$$

$$\textbf{thursday} \overset{\text{def}}{=} 4 \tag{1.57}$$

$$\textbf{friday} \overset{\text{def}}{=} 5 \tag{1.58}$$

$$\textbf{saturday} \overset{\text{def}}{=} 6 \tag{1.59}$$

and determine the day of the week with

$$\textbf{day-of-week-from-fixed}\,(\textit{date}) \overset{\text{def}}{=} \tag{1.60}$$
$$(\textit{date} - \text{R.D. } 0 - \textbf{sunday}) \bmod 7$$

We include the superfluous terms −R.D. 0 − **sunday** = 0 to make this function independent of any particular choice of epoch for fixed dates, and for that reason we use this function in the formulas that follow.

Many holidays are on the *n*th occurrence of a given day of the week, counted forward or backward from some date. For example, Thanksgiving in the United States is the fourth Thursday in November, that is, the fourth Thursday on or after November 1. We handle such specifications by writing a function that encapsulates the formula

$$k \bmod [\textit{date} \,..\, \textit{date} - 7) \tag{1.61}$$

to find the kth day of the week ($k = 0$ for Sunday, and so on) that falls in the 7-day period ending on R.D. *date*. Using equations (1.25) and (1.26) we find that (1.61) is equivalent to

$$\begin{aligned} k \bmod [\textit{date} \,..\, \textit{date} - 7) &= \textit{date} + (k - \textit{date}) \bmod [0 \,..\, -7) \\ &= \textit{date} - (\textit{date} - k) \bmod [0 \,..\, 7) \\ &= \textit{date} - (\textit{date} - k) \bmod 7 \end{aligned}$$

To incorporate the possibility that day 0 of the day count is other than a Sunday, we use the function **day-of-week-from-fixed** (1.60):

$$\textbf{kday-on-or-before}\,(k, \mathit{date}) \stackrel{\text{def}}{=} \mathit{date} - \textbf{day-of-week-from-fixed}\,(\mathit{date} - k) \tag{1.62}$$

We generally use the parameter *date* for R.D. dates.

Formula (1.61) is an instance of a more general principle for finding the occurrence of the kth day of a repeating m-day cycle that is closest to but not past day number d, where day number 0 is day Δ of the m-day cycle:

$$\boxed{(k - \Delta) \bmod [d \mathinner{..} d - m)} \tag{1.63}$$

This formula works equally well for negative and nonintegral dates d (that is, for a time of day) and for nonintegral positions k, shifts Δ, and periods m. We use such computations extensively for the Hindu calendars (Chapter 10), the Mayan calendars (Chapter 11), and the Balinese Pawukon calendar (Chapter 12).

Note that formula (1.63) for the last k-label day on or before day number d remains correct even if the cycle of labels is $a, a + 1, \ldots, a + m - 1$ (that is, based at a instead of 0). We use this in the Chinese calendar (Chapter 19) for $a = 1$, that is, for 1-based cycles of labels, and also for Balinese dates (Chapter 12).

Similarly, the first k-labeled moment at or after moment d is

$$(k - \Delta) \bmod [d \mathinner{..} d + m) \tag{1.64}$$

We can write a function **kday-on-or-after** R.D. d by applying **kday-on-or-before** to $d + 6$. Similarly, applying it to $d + 3$ gives the **kday-nearest** to R.D. d, applying it to $d - 1$ gives the **kday-before** R.D. d, and applying it to $d + 7$ gives the **kday-after** R.D. d:

$$\textbf{kday-on-or-after}\,(k, \mathit{date}) \stackrel{\text{def}}{=} \textbf{kday-on-or-before}\,(k, \mathit{date} + 6) \tag{1.65}$$

$$\textbf{kday-nearest}\,(k, \mathit{date}) \stackrel{\text{def}}{=} \textbf{kday-on-or-before}\,(k, \mathit{date} + 3) \tag{1.66}$$

$$\textbf{kday-before}\,(k, \mathit{date}) \stackrel{\text{def}}{=} \textbf{kday-on-or-before}\,(k, \mathit{date} - 1) \tag{1.67}$$

$$\textbf{kday-after}\,(k, \mathit{date}) \stackrel{\text{def}}{=} \textbf{kday-on-or-before}\,(k, \mathit{date} + 7) \tag{1.68}$$

Equations (1.62) and (1.65)–(1.68) are specific instances of more general calculations that occur in cyclical calendars such as the Mayan, Aztec, and Balinese. The general form can be expressed as

$$(k - \Delta) \bmod [\delta(d) \mathinner{..} \delta(d) - m) \tag{1.69}$$

where the length of the repeating cycle is m, k is the desired position in the cycle, R.D. 0 is at position Δ in the cycle, and the function $\delta(d)$ is chosen according to Table 1.3.

Table 1.3 Functions $\delta(d)$ for use in formula (1.69).

Relation to date d	$\delta(d)$
before d	$d-1$
on or before d	d
after d	$d+m$
on or after d	$d+m-1$
nearest to d	$d+\lfloor m/2 \rfloor$

1.13 Simultaneous Cycles

In the year 4-House of the eighth sheaf of years of the Mexican era the Emperor Moteçuçuma the Younger had a great fright. We know this year as 1509. The Mexicans counted their time in "sheafs" of fifty-two years, and in order to designate them without error or ambiguity, a system had been adopted which can be best understood by reference to a pack of cards: as if we were to call our years one of spades, two of hearts, three of diamonds, four of clubs, five of spades, six of hearts, seven of diamonds, eight of clubs, etc. It is clear that the series or "sheaf" would begin again every fifty-two years. The Mexican calendar divided the fifty-two years of a "sheaf" into four sets or "colours" of thirteen years, i.e., rabbits, reeds, flints and houses.

Salvador de Madariaga: *Hernán Cortés: Conqueror of Mexico* (1942)

Some calendars employ two cycles running simultaneously. Each day is labeled by a pair of numbers $\langle a, b\rangle$, beginning with $\langle 0, 0\rangle$, followed by $\langle 1, 1\rangle$, $\langle 2, 2\rangle$, and so on. Suppose the first component repeats after c days and the second after d days, with $c < d < 2c$; then after day $\langle c-1, c-1\rangle$ come days $\langle 0, c\rangle$, $\langle 1, c+1\rangle$, and so on until $\langle d-c-1, d-1\rangle$, which is followed by $\langle d-c, 0\rangle$. If day 0 of the calendar is labeled $\langle 0, 0\rangle$ then day n is $\langle n \bmod c, n \bmod d\rangle$. The Chinese use such pairs to identify years (see Section 19.4), with cycles of length $c = 10$ and $d = 12$ but, because the first component ranges from 1 to 10, inclusive, and the second from 1 to 12, we would use the adjusted remainder function: $\langle n \bmod [1\,..\,10], n \bmod [1\,..\,12]\rangle$

More generally, for arbitrary positive integers c and d, if the label of day 0 is $\langle \Gamma, \Delta\rangle$ then day n is labeled

$$\boxed{\langle (n+\Gamma) \bmod c, (n+\Delta) \bmod d\rangle} \tag{1.70}$$

For the Mayan tzolkin calendar, with $c = 13$, $d = 20$, $\Gamma = 3$, $\Delta = 19$, and beginning the cycles with 1 instead of 0, this is $\langle (n+3) \bmod [1\,..\,13], (n+19) \bmod [1\,..\,20]\rangle$. It follows that day 1 of the Mayan calendar is labeled $\langle 4, 20\rangle$ (see Section 11.2).

How many distinct day names does such a scheme provide? If m is the least common multiple (lcm) of c and d, then such a calendar repeats after m days. If the cycle lengths c and d are relatively prime (that is, no integer greater than 1 divides both c and d without remainder), then it repeats after $m = c \times d$ days. Thus, for the Mayan tzolkin calendar, with $c = 13$ and $d = 20$, m is 260. For the Chinese year names, $\mathrm{lcm}(10, 12) = 60$ yielding a sexagesimal cycle.

Inverting this representation is harder. Suppose first that $\Gamma = \Delta = 0$. Given a pair $\langle a, b\rangle$, where a is an integer in the range $0 \,..\, c-1$ and b is an integer in the range $0 \,..\, d-1$, we are looking for an n, $0 \leqslant n < m$, such that $a = n \bmod c$ and $b = n \bmod d$. This requires the solution to a pair of simultaneous linear congruences (this is an instance of the Chinese Remainder Theorem; see, for example, [28]):

$$n \equiv a \pmod{c}$$

$$n \equiv b \pmod{d}$$

The first congruence means that

$$n = a + ic \tag{1.71}$$

for some integer i. Substituting this for n in the second congruence and transposing, we get

$$ic \equiv b - a \pmod{d}$$

Let g be the greatest common divisor (gcd) of c and d and let $u = c/g$ and $v = d/g$, so that u and v are relatively prime. Now let k be the multiplicative inverse of u modulo v; that is,

$$ku \bmod v = 1$$

We can use the Fermat-Euler Theorem [19, Theorem 72, p. 63], because u and v are relatively prime:

$$k = u^{\varphi(v)-1} \bmod v \tag{1.72}$$

where the totient function $\varphi(v)$ counts the number of integers i, $1 \leqslant i \leqslant v$, that are relatively prime to v. (The multiplicative inverse k can also be determined using the Euclidean algorithm; see [34] for details.) Now,

$$k\frac{c}{g} \bmod \frac{d}{g} = 1$$

Then

$$i \equiv ik\frac{c}{g} \equiv k\frac{b-a}{g} \left(\bmod\ \frac{d}{g}\right)$$

Using this value of i in equation (1.71), we get day number

$$a + c\left[k\frac{b-a}{g} \bmod \frac{d}{g}\right] = a + \left[\frac{ck}{g}(b-a) \bmod \frac{cd}{g}\right]$$

When day 0 is labeled $\langle\Gamma, \Delta\rangle$, we must subtract Γ from a and Δ from b. To make sure that n is in the range $0 \,..\, m-1$, we use

$$\boxed{n = \left(a - \Gamma + \frac{ck[b - a + \Gamma - \Delta]}{\gcd(c, d)}\right) \bmod \operatorname{lcm}(c, d)} \tag{1.73}$$

For example, if $c = 10$ and $d = 12$, as in the Chinese calendar, then $\gcd(10, 12) = 2$, $\text{lcm}(10, 12) = 60$, and $k = 5$ because $(5 \times 5) \bmod 6 = 1$. Using $\Gamma = \Delta = 0$, but counting from 1 instead of 0, we find that Chinese year name $\langle a, b\rangle$ corresponds to year number

$$(a + 25(b - a)) \bmod [1 \,..\, 60] \tag{1.74}$$

of the sexagesimal cycle; we use this formula in Section 19.4. We use other derivations of this sort for the Hebrew calendar in Section 8.3, for the Mayan calendars in Section 11.2, and for the Balinese calendar in Chapter 12.

Note that some combinations $\langle a, b\rangle$ are impossible. In general, there is no solution (1.73) unless

$$\boxed{\gcd(c, d) \text{ divides } (b - a + \Gamma - \Delta)} \tag{1.75}$$

or, equivalently,

$$(b - a + \Gamma - \Delta) \bmod \gcd(c, d) = 0$$

For example, with the Chinese scheme, the odd-even parity of the two components must be the same because c and d are both even, and only 60 of the 120 conceivable pairs are possible.

Equation (1.73) is all we need to implement the calendar of the Akan people of Ghana [5], [29], [35], which is based on a 42-day cycle of day names formed by two simultaneous cycles of six prefixes and seven stems. It is similar to the Mayan tzolkin calendar (Section 11.2) and the Chinese sexagesimal names (Section 19.4). The prefixes of the Akan calendar are

(1) Nwona (care, wellness, surpass, innocence)
(2) Nkyi (passing, no restrictions)
(3) Kuru (sacred, complete)
(4) Kwa (ordinary, empty, freedom)
(5) Mono (fresh, new)
(6) Fo (generous, calm, love to another)

and the stems are

(1) Wukuo (cleansing, advocate, mean-spirited)
(2) Yaw (pain, suffering, bravery)
(3) Fie (depart from, come forth, travel)
(4) Memene (digest, satiety, creation, ancient)
(5) Kwasi (freedom, purify, smoke)
(6) Dwo (peaceful, cool, calm)
(7) Bene (well-cooked)

Together these prefixes and suffixes form a sequence of 42 day names, Nwonawukuo, Nkyiyaw, . . . , Fobene. Representing Akan day names as pairs of positive integers,

$$\boxed{\textit{prefix}}\boxed{\textit{stem}}$$

where *prefix* and *stem* are integers in the ranges 1 to 6 and 1 to 7, respectively, the nth Akan day name is given by

$$\textbf{akan-day-name}\,(n) \stackrel{\text{def}}{=} \boxed{n \bmod [1\,..\,6]}\boxed{n \bmod [1\,..\,7]} \tag{1.76}$$

Applying formula (1.73) with $c = 6$, $d = 7$, $\Gamma = \Delta = 0$ but counting from 1 instead of 0, we find that the Akan name $\boxed{a}\boxed{b}$ corresponds to name number

$$(a + 36(b - a)) \bmod [1\,..\,42]$$

Determination of the number of names between given Akan names is thus given by

$$\textbf{akan-name-difference} \tag{1.77}$$
$$\left(\boxed{\textit{prefix}_1}\boxed{\textit{stem}_1}, \boxed{\textit{prefix}_2}\boxed{\textit{stem}_2}\right) \stackrel{\text{def}}{=}$$
$$\left(\textit{prefix-difference} + 36 \times (\textit{stem-difference} - \textit{prefix-difference})\right)$$
$$\bmod [1\,..\,42]$$

where

$$\textit{prefix-difference} = \textit{prefix}_2 - \textit{prefix}_1$$
$$\textit{stem-difference} = \textit{stem}_2 - \textit{stem}_1$$

Computing backwards from known dates in the present, we find that a cycle began on R.D. 37, so that we have

$$\textbf{akan-day-name-epoch} \stackrel{\text{def}}{=} \text{R.D. } 37 \tag{1.78}$$

which allows us to write

$$\textbf{akan-name-from-fixed}\,(\textit{date}) \stackrel{\text{def}}{=} \tag{1.79}$$
$$\textbf{akan-day-name}\,(\textit{date} - \textbf{akan-day-name-epoch})$$

Now we can apply formula (1.63) to compute the R.D. date of the last date with a given Akan name before a given R.D. date:

$$\textbf{akan-day-name-on-or-before}\,(\textit{name}, \textit{date}) \stackrel{\text{def}}{=} \tag{1.80}$$
$$\textbf{akan-name-difference}\,(\textbf{akan-name-from-fixed}\,(0), \textit{name})$$
$$\bmod [\textit{date}\,..\,\textit{date} - 42)$$

1.14 Cycles of Years

An ordinary person cannot count each day, and say this is so many and so many days. Instead, the count uses a significant unit, that is, years.

T. Schvarcz: *Zichron Menachem*, 5673 A.M. (= 1913–14); from a talk given in June 1907 to honor the fortieth anniversary of Franz Josef I's rule of Hungary

We now derive some general formulas that are useful in calendar conversions for the Julian, Islamic, Coptic, Hebrew, arithmetic Persian, and old Hindu lunisolar calendars (although not in the same way for the Gregorian calendar, unfortunately), as well as serving as the basis for the generic solar and lunisolar calendars in Chapter 13. All of these calendars have in common that they follow a simple type of leap-year rule in which leap years are spread as evenly as possible over a cycle of years; the particular constants that define these leap-year rules are given in Table 1.4. The formulas in this section are closely related to Bresenham's "midpoint line algorithm" for drawing lines in two dimensions on a discrete raster graphics image [20], [47].

Suppose we have a sequence of years $\dots, -2, -1, 0, 1, 2, \dots$, and we want to place l leap years in a cycle of c years, with year 0 as the first year of the cycle. How can we spread the leap years evenly over the cycle? If l is a divisor of c, our problem is easy: Let year numbers that are multiples of c/l be leap years. If l is not a divisor of c, however, the best we can do is to let year numbers that are *roughly* multiples of c/l be leap years—specifically, we have a leap year whenever the year number has reached, or just passed, a multiple of c/l. Let y be a year number; then it is a leap year if

$$y - 1 < k\frac{c}{l} \leqslant y$$

for some integer k. Rearranging this inequality, we get

$$k\frac{c}{l} \leqslant y < k\frac{c}{l} + 1 \tag{1.81}$$

which is the same as saying that

$$0 \leqslant \left(y \bmod \frac{c}{l}\right) < 1$$

Multiplying by l and using equation (1.19), we obtain

$$0 \leqslant (yl \bmod c) < l$$

Because our cycles always have length $c > 0$, the definition of the mod function guarantees that $(yl \bmod c) \geqslant 0$, so we can drop that part of the inequality to get

$$(yl \bmod c) < l \tag{1.82}$$

For example, on the Julian calendar for years C.E. (see Chapter 3) we want $l = 1$ leap year in the cycle of $c = 4$ years; then year $y > 0$ is a leap year if

$$(y \bmod 4) < 1$$

Table 1.4 Constants describing the simple leap-year structure of various calendars; c is the length of the leap-year cycle, l is the number of leap years in that cycle of c years, Δ is the position in the cycle of year 0, L is the length of an ordinary year (hence $L+1$ is the length of a leap year), $\bar{L} = (cL+l)/c$ is the average length of a year, and $\delta = (\Delta l)/c \bmod 1$ is the time of day or month (as a fraction of the day or month, respectively) when mean year 0 begins. This cyclic pattern also applies to Islamic months, and approximately to the Gregorian/Julian months.

	Calendar	Section	c	l	Δ	L	$\bar{L} = \frac{cL+l}{c}$	$\delta = \frac{\Delta l}{c} \bmod 1$
Years	Julian C.E.	3.1	4	1	0	365 days	$\frac{1461}{4}$ days	0 days
	Julian B.C.E.	3.1	4	1	1	365 days	$\frac{1461}{4}$ days	$\frac{1}{4}$ day
	Coptic	4.1	4	1	1	365 days	$\frac{1461}{4}$ days	$\frac{1}{4}$ day
	Islamic	7.1	30	11	4	354 days	$\frac{10631}{30}$ days	$\frac{7}{15}$ day
	Islamic (variant)	7.1	30	11	15	354 days	$\frac{10631}{30}$ days	$\frac{1}{2}$ day
	Hebrew	8.1	19	7	11	12 months	$\frac{235}{19}$ months	$\frac{1}{19}$ month
	Ecclesiastical	9.1	19	7	13	12 months	$\frac{235}{19}$ months	$\frac{1}{19}$ month
	Old Hindu lunisolar	10.3				12 months	$\frac{2226389}{180000}$ months	$\frac{2093611}{2160000}$ month
	Persian	15.3	128	31	38	365 days	$\frac{46751}{128}$ days	$\frac{13}{64}$ day
Months	Gregorian/Julian (approximate)	2.2	12	7	11	30 days	$\frac{367}{12}$ days	$\frac{5}{12}$ day
	Gregorian/Julian (approximate)	2.3	7	4	6	30 days	$\frac{214}{7}$ days	$\frac{3}{7}$ day
	Gregorian/Julian (March–March)	2.3	5	3	4	30 days	$\frac{153}{5}$ days	$\frac{2}{5}$ day
	Islamic (ordinary year)	7.1	12	6	1	29 days	$\frac{59}{2}$ days	$\frac{1}{2}$ day
	Islamic	7.1	11	6	10	29 days	$\frac{325}{11}$ days	$\frac{5}{11}$ day

or, in other words, if

$$(y \bmod 4) = 0$$

We can complicate the leap-year situation by insisting that year 0 be in position Δ in the cycle of c years. In this case, we have the same analysis but pretend that the cycle begins at year 0 and ask about year $y + \Delta$. Inequality (1.82) becomes

$$\boxed{[(y + \Delta)l \bmod c] < l} \tag{1.83}$$

For example, the Julian calendar for years B.C.E. (Chapter 3) and the Coptic calendar (Chapter 4) have a cycle of $c = 4$ years containing $l = 1$ leap years with $\Delta = 1$. Inequality (1.83) becomes

$$[(y + 1) \bmod 4] < 1$$

this is equivalent to

$$(y \bmod 4) = 3$$

The Islamic calendar (Chapter 7) has a cycle of $c = 30$ years containing $l = 11$ leap years with $\Delta = 4$ (some Muslims have a different leap-year structure, which corresponds to $\Delta = 15$; see page 107), so the test for an Islamic leap year is

$$[(11y + 14) \bmod 30] < 11$$

Spreading 11 leap years evenly over 30 years implies gaps of 2 or 3 years between leap years. Because $\frac{30}{11} = 2\frac{8}{11}$, 3 of the 11 leap years each occur after a gap of only 2 years. If we associate each leap year with the gap preceding it and number the gaps 0, 1, ... 10, these three short gaps are numbers 2, 6, and 9, to which formula (1.83) could also be applied (with $c = 11$, $l = 3$, and $\Delta = 2$).

If $\Delta = 0$, inequality (1.81) implies that

$$k = \left\lfloor \frac{y}{c/l} \right\rfloor \tag{1.84}$$

is the number of leap years in the range of years $1 \,..\, y$. When $\Delta \neq 0$, we again pretend that the cycle begins at year 0 and ask about year $y + \Delta$ instead of year y. Thus, the number of leap years in the range $1 \,..\, y - 1$ for $\Delta \neq 0$ is the same as the number of leap years in the unshifted range of years $\Delta + 1 \,..\, y + \Delta - 1$ (whether y is positive or negative), namely,

$$\boxed{\left\lfloor \frac{y + \Delta - 1}{c/l} \right\rfloor - \left\lfloor \frac{\Delta}{c/l} \right\rfloor = \left\lfloor \frac{ly - l + (\Delta l \bmod c)}{c} \right\rfloor} \tag{1.85}$$

the number of years in the unshifted range $1 \,..\, y + \Delta - 1$ minus the number in the unshifted range $1 \,..\, \Delta$. For example, $\lfloor (y - 1)/4 \rfloor$ is the number of leap years before year y on the Julian calendar (counting from the Julian epoch), $\lfloor (11y + 3)/30 \rfloor$ is the number of leap years prior to year y on the Islamic calendar, and $\lfloor y/4 \rfloor$ is the number of leap years prior to year y on the Coptic calendar.

Using formula (1.85), we immediately get the following formula for the number of days in the years before year y—that is, the number of days in the years 1, 2, 3, ..., $y-1$, assuming there are L days in an ordinary year and $L+1$ days in a leap year:

$$\boxed{n = \left\lfloor \frac{ly - l + (\Delta l \bmod c)}{c} \right\rfloor + L(y-1)} \tag{1.86}$$

For example, for the Julian calendar this yields $\lfloor (y-1)/4 \rfloor + 365(y-1)$, for the Coptic calendar this yields $\lfloor y/4 \rfloor + 365(y-1)$, and for the Islamic calendar it yields $\lfloor (11y+3)/30 \rfloor + 354(y-1)$. Because the Hebrew calendar (and lunisolar calendars in general) adds leap months, formula (1.86) does not apply to days, but it does apply to *months*: The number of months prior to year y on the Hebrew calendar is $\lfloor (7y-6)/19 \rfloor + 12(y-1)$.

Formula (1.86) works for $y \leqslant 0$. In this case it computes the number of days in years $y \,..\, 0$ as a negative number.

Finally, we can derive an inverse to formula (1.86) to find the year at day n, counting day $n = 0$ as the first day of year 1 (the epoch). Because there are L days in an ordinary year and $L+1$ days in a leap year, the average year length is

$$\bar{L} = \frac{cL + l}{c}$$

In the simple case that $\Delta = 0$, year y begins on day

$$\begin{aligned} n &= (y-1)L + (\text{number of leap years in } 1 \,..\, y-1) \\ &= (y-1)L + \left\lfloor \frac{y-1}{c/l} \right\rfloor \\ &= \lfloor (y-1)\bar{L} \rfloor \end{aligned} \tag{1.87}$$

by using formula (1.84) and simplifying. Day n is in year y provided that it is on or after the first day of year y and before the first day of year $y+1$; that is,

$$\lfloor (y-1)\bar{L} \rfloor \leqslant n < \lfloor y\bar{L} \rfloor \tag{1.88}$$

The sequence $\lfloor \bar{L} \rfloor, \lfloor 2\bar{L} \rfloor, \lfloor 3\bar{L} \rfloor, \ldots$ is called the *spectrum* of $\bar{L}$ (see [18, sec. 3.2]); in our case, they are the initial day numbers of successive years. Inequality (1.88) is equivalent to

$$(y-1)\bar{L} - 1 < n \leqslant y\bar{L} - 1$$

from which it follows that

$$y = \left\lceil \frac{n+1}{\bar{L}} \right\rceil \tag{1.89}$$

In general, when $\Delta \neq 0$, we must shift Δ years backward; that is, shift the first day of year 1 to the first day of year $-\Delta + 1$. The number of days in the shifted years $-\Delta + 1 \,..\, 0$ is the same as the number of days in the unshifted years $1 \,..\, \Delta$, which is computed by adding the L ordinary days in each of those Δ years, plus the $\lfloor \Delta/(c/l) \rfloor$ leap days in those years as given by (1.84). The shift of Δ years thus corresponds to a shift of $\Delta L + \lfloor \Delta/(c/l) \rfloor$ days. So the shifted form of (1.89) is

$$y + \Delta = \left\lceil \frac{n + 1 + \Delta L + \left\lfloor \frac{\Delta}{c/l} \right\rfloor}{\bar{L}} \right\rceil$$

which is the same as

$$\boxed{\begin{aligned} y &= \left\lceil \frac{cn + c - (l\Delta \bmod c)}{cL + l} \right\rceil \\ &= \left\lfloor \frac{cn + cL + l - 1 + c - (l\Delta \bmod c)}{cL + l} \right\rfloor \end{aligned}} \tag{1.90}$$

We usually prefer the latter form because the floor function is more readily available than the ceiling function in computer languages.

For the Julian calendar, formula (1.90) gives day n occurring in year

$$\left\lceil \frac{4n + 4}{1461} \right\rceil = \left\lfloor \frac{4n + 1464}{1461} \right\rfloor$$

for the Coptic calendar it gives year

$$\left\lceil \frac{4n + 3}{1461} \right\rceil = \left\lfloor \frac{4n + 1463}{1461} \right\rfloor$$

and for the Islamic calendar it gives year

$$\left\lceil \frac{30n + 16}{10631} \right\rceil = \left\lfloor \frac{30n + 10646}{10631} \right\rfloor$$

Formula (1.90) does not apply to days on the Hebrew calendar but rather to months, giving the formula

$$\left\lceil \frac{19n + 18}{235} \right\rceil = \left\lfloor \frac{19n + 252}{235} \right\rfloor \tag{1.91}$$

for the year in which month n occurs; we use this formula in Section 8.3 to get the month/year corresponding to *elapsed-months* in **fixed-from-molad**.

Formula (1.90) makes sense when $n < 0$, too. In this case it gives the correct year as a negative number (but, as discussed earlier, this is off by one for Julian B.C.E. years).

A more general approach to leap-year distribution is to imagine a sequence of *mean years* of (noninteger) length $\bar{L}$, with year 1 starting on day 0 at time δ, $0 \leqslant \delta < 1$, where δ expresses time as a fraction of a day. We define a *calendar year* y to begin at the start of the day on which mean year y begins; that is, mean year y begins at moment $\delta + (y - 1)\bar{L}$, and thus calendar year y begins on day

$$\boxed{n = \lfloor (y - 1)\bar{L} + \delta \rfloor} \tag{1.92}$$

Calendar year y is an ordinary year if

$$\lfloor y\bar{L} + \delta \rfloor - \lfloor (y-1)\bar{L} + \delta \rfloor = \lfloor \bar{L} \rfloor$$

and a leap year if

$$\lfloor y\bar{L} + \delta \rfloor - \lfloor (y-1)\bar{L} + \delta \rfloor = \lfloor \bar{L} \rfloor + 1$$

By definition (1.17), this latter equation tells us that calendar year y is a leap year if

$$\left(\delta + (y-1)(\bar{L} \bmod 1)\right) \bmod 1 \geqslant 1 - \left(\bar{L} \bmod 1\right)$$

or, equivalently, if

$$\boxed{\left(\delta + (y-1)\bar{L}\right) \bmod 1 \geqslant 1 - \left(\bar{L} \bmod 1\right)} \tag{1.93}$$

For the old Hindu lunisolar calendar, with the year count beginning at 0 (not 1), average year length of

$$\bar{L} = \frac{2226389}{180000} \approx 12.368828$$

months, and

$$\delta = \frac{2093611}{2160000}$$

inequality (1.93) means that y is a leap year if

$$\left(\frac{2093611}{2160000} + y\frac{2226389}{180000}\right) \bmod 1 \geqslant 1 - \frac{66389}{180000} = \frac{113611}{180000}$$

or, equivalently,

$$(2093611 + 796668\, y) \bmod 2160000 \geqslant 1363332$$

(See page 163.) However, this test is not needed for other calculations on the old Hindu calendar.

When $\delta = 0$, mean year 1 and calendar year 1 both begin at the same moment, and equation (1.92) tells us that leap years follow the same pattern as for $\Delta = 0$ in our earlier discussion. More generally, given any Δ, if we choose

$$\delta = \frac{\Delta l}{c} \bmod 1 \tag{1.94}$$

the leap-year test (1.93) simplifies to (1.83), and thus we have the same leap-year structure. For example, the Coptic calendar has $\delta = [(1 \times 1)/4] \bmod 1 = 1/4$.

Our δ formulas generalize our Δ formulas because formula (1.94) gives a corresponding value of δ for each Δ. However, there need not be a value of Δ for arbitrary $\bar{L}$ and δ; indeed, there is no such Δ for calendars in which the mean and calendar years never begin at exactly the same moment. Given $\bar{L}$ and δ, we have $l/c = \bar{L} \bmod 1$, and (1.94) means that Δ exists only if δ is an integer multiple, modulo 1, of $\bar{L}$. In the old Hindu lunisolar calendar, for example, formula (1.83) cannot be used directly: $\bar{L} \bmod 1 = 66389/180000$, and we must have an integer Δ such that

$$\frac{2093611}{2160000} = \left(\Delta \frac{66389}{180000}\right) \bmod 1$$

or

$$2093611 = (796668\,\Delta) \bmod 2160000$$

No such Δ exists because 796668 and 2160000 are both even, but 2093611 is odd. When $\bar{L}$ is rational, and $\bar{L} = l/c$ for relatively prime l and c, then we *can* use formulas (1.83)–(1.90), with Δ such that $l\Delta \equiv \lfloor c\delta \rfloor \pmod{c}$. For example, for the old Hindu lunisolar calendar we can use $c = 180000$, $l = 66389$, $\lfloor c\delta \rfloor = 174467$ and $\Delta = 147703$. However, the δ formula is more general in that it applies even if average year length is not rational.

The generalization of formula (1.90) in terms of δ follows by solving equation (1.92) for y, to yield

$$\boxed{y = \left\lceil \frac{n + 1 - \delta}{\bar{L}} \right\rceil} \tag{1.95}$$

For the Coptic calendar, this becomes

$$y = \left\lceil \frac{n + 1 - 1/4}{1461/4} \right\rceil = \left\lceil \frac{4n + 3}{1461} \right\rceil$$

as we knew before.

For the old Hindu lunisolar calendar, in every 180000-year cycle there are 66389 evenly distributed leap years of 13 months. Because the year count begins with year 0, month m falls in year

$$y = \left\lceil \frac{m + 1 - \dfrac{2093611}{2160000}}{\dfrac{2226389}{180000}} \right\rceil - 1$$

The application of these formulas to the old Hindu lunisolar calendar is discussed in Chapter 10.

In the foregoing discussion we have counted days beginning with the epoch of the calendars, and thus when formulas (1.86) and (1.90) are used in our calendrical functions, the epoch must be added or subtracted to refer to R.D. dates. For example, to compute the Islamic year of R.D. d, we must write

$$\left\lfloor \frac{30(d - \text{Islamic epoch}) + 10646}{10631} \right\rfloor$$

because R.D. d is $(d - \text{Islamic epoch})$ elapsed days on the Islamic calendar.

1.15 Approximating the Year Number

At the expiration of the years, come challenge me.
Shakespeare: *Love's Labour's Lost*, Act V, scene ii (1598)

For calendars that do not follow the strict paradigm of the previous section, a useful method to determine the exact year number of a fixed date d is to estimate the year number and then correct it if necessary. Let Y be the *average* year length. Given a_1, the (actual or approximate) first day of year 1 on the calendar, the *mean* new year a_j of year j, for any year j, is simply

$$a_j = a_1 + (j - 1)Y$$

Conversely, for any moment d, the *approximate* year number y can be determined by division:

$$y = \left\lfloor \frac{d - a_1}{Y} \right\rfloor + 1$$

For all years j, let n_j be the actual fixed date of the start of the year on the calendar. Assuming that the actual date n_j is within a year of the mean date a_j, we need only check whether the above estimate y is off by 1:

$$\mathit{year} = \begin{cases} y - 1 & \textbf{if } d < n_y \\ y + 1 & \textbf{if } d \geqslant n_{y+1} \\ y & \textbf{otherwise} \end{cases}$$

The calculation of the exact new year can be relatively expensive, so we would like to avoid computing n_y and n_{y+1} whenever possible. Suppose that we can bound the difference between the mean dates, a_j, and the actual dates, n_j, so that it is guaranteed that

$$n_j - \varepsilon \leqslant a_j \leqslant n_j + \delta$$

for bounds $\delta \geqslant 0$ and $\varepsilon \leqslant Y$; suppose further that these bounds hold for all years j—or at least for all years within a 100 centuries of the current date (which is all that we demand of our algorithms). Then, whenever the given date d is far enough away from the two mean new years a_y and $a_y + Y$, we can be sure that the approximation y is accurate.

Let $\Delta = (d - a_1) \bmod Y$. If $\Delta \leqslant \delta$, then d falls in the "twilight zone" and we need to check whether $d < n_y$; if so then the estimate is actually wrong, and the correct year is $y - 1$. Similarly, if $\Delta \geqslant Y - \epsilon$, then d is too close to the end of year y for us to be certain, and we need to check if $d \geqslant n_{y+1}$, in which case the correct year is $y + 1$. The test for $\Delta \leqslant \delta$ may be omitted if $\delta = 0$ (Δ is nonnegative, and if $\Delta = \delta = 0$, then y must be exactly $(d - a_1)/Y + 1$, in which case $n_y = n_y - \varepsilon \leqslant a_y = a_1(y-1)Y = d$). Likewise, the test for $\Delta \geqslant Y - \varepsilon$ may be omitted if $\varepsilon = 0$ (because $\Delta < Y$). By shifting the initial estimate a_1, one can ensure that $\varepsilon = 0$; then we get the precise year number for fixed date d with

$$\mathit{year} = \begin{cases} y - 1 & \textbf{if } (d - a_1) \bmod Y \leqslant \delta \text{ and } d < n_y \\ y & \textbf{otherwise} \end{cases} \tag{1.96}$$

where

$$y = \left\lfloor \frac{d - a_1}{Y} \right\rfloor + 1$$

We use this method for Gregorian calendar years in (2.30), for Hebrew calendar years in (8.28), and for arithmetic French Revolutionary calendar years in (17.10).

1.16 Warnings about the Calculations

Caveat emptor. [*Let the buyer beware.*]
Latin motto

We have been careful to ensure that our conversion functions work for at least ±10000 years from the present, if not forever. We have worked hard to make sure that our conversion algorithms do not suffer from a Y10K problem!

Many holiday calculations assume that the Gregorian year and the true solar year, and/or the mean year length of a specific calendar, maintain the same alignment, which will not remain the case over millennia. We have endeavored to make these calculations robust for at least ±2000 years from the present. Of course, the dates of most holidays will not be historically correct over that range.

The astronomical code we use is not the best available but it works quite well in practice, especially for dates near the present time, around which its approximations are centered. More precise code would be more time-consuming and complex and would not necessarily yield more accurate results for those calendars that depended on observations, tables, or less accurate calculations. Thus, the correctness of a date on any of the astronomical calendars is contingent on the historical accuracy of the astronomical code used in its calculation.

We have chosen not to optimize the algorithms at the expense of clarity; consequently, considerable improvements in economy are possible, some of which are pointed out. In particular, our algorithms are designed to convert individual dates from one calendar to another; thus the preparation of monthly or yearly calendars would benefit enormously if intermediate results were stored and used for subsequent days. This standard algorithmic technique (called “caching” or “memoization”) is ignored in this book.

We do not do error checking in the code. If one asks for the R.D. date corresponding to a date in Julian year 0, or to February 29, 1990, an answer will be forthcoming despite the nonexistence of such dates. Similarly, the code will not object to the absurdity of asking for the R.D. date corresponding to December 39, or even the thirty-ninth day of the thirteenth month. In other cases, we use the special constant

$$\textbf{bogus} \tag{1.97}$$

to indicate that a calendar date, holiday, or astronomical event is nonexistent. For each calendar *x*, the validity of a date *x-date* on that calendar can be checked by a function

$$\textbf{valid-}\boldsymbol{x}\textbf{-date}(x\text{-}date) \overset{\text{def}}{=}$$

$$x\text{-}date = \textbf{\textit{x}-from-fixed}(\textbf{fixed-from-\textit{x}}(x\text{-}date))$$

All our functions give "correct" (mathematically sensible) results for negative years and for dates prior to the epoch of a calendar. However, these results may be *culturally* wrong in the sense that, say, the Copts may not refer to a year 0 or -1. It may be considered heretical on some calendars to refer to years before the creation of the world.

All Gregorian dates before the Common Era that appear in this book follow the astronomical convention of using nonpositive year numbers—including 0 for the year preceding the onset of the era. (The varying conventions with regard to Gregorian year 0 have led to many errors in the converting of historical dates.) Year 0 is assumed to exist for all calendars *except* the Julian (Chapter 3) and the Persian (Chapter 15).

Except for our summation and product operators (page 23) and search functions (page 23), we avoid iteration and instead use recursion, which is natural because we use functional notation. The use of recursion, however, is not essential: it is invariably "tail" recursion and can easily be replaced by iteration.

Our algorithms assume that if $y > 0$, then $(x \bmod y) \geqslant 0$ for all x, even for negative values of x. Thus, as we stated in Section 1.7, care must thus be exercised in implementing our algorithms in computer languages like C or C++, in which the built-in mod function (often the `%` operator) may give $(x \bmod y) < 0$ for $x < 0$, $y > 0$. We also assume, in the case of some functions, that $x \bmod y$ works for real numbers x and y, as well as for integers.

Care must be taken with indices of arrays and cycles. Our arrays are 0-based, while some programming languages begin with index 1. However, when we speak of elements of a sequence as "first," "second," and so on, we intend standard English usage with no zeroth element. Most calendars number their days, months, and years starting with 1, but there are exceptions—Hindu years and Mayan days, for example. Some cycle formulas in this chapter work for arbitrary starting points; others require adjustment when the first element of a cycle is not 0.

Checking the results of conversions against the historical record is sometimes misleading because the different calendars begin their days at different times. For example, a person who died in the evening will have a different Hebrew date of death than if he or she had died in the morning of the same Gregorian calendar date; gravestone inscriptions often err in this. All our conversions are as of noon.

Some of our calculations require extremely large numbers; other calculations depend on numerically accurate approximations to lunar or solar events. All functions for the calendars in Part I, except the old Hindu, work properly (for dates within thousands of years from the present) in 32-bit integer arithmetic; the Hebrew calendar approaches this limit, so we have indicated how to rephrase the calculations to use only small numbers—one exception is **fixed-from-molad** (page 126) which requires 64-bit integers. On the other hand, 64-bit arithmetic is needed to reproduce accurately the results of the astronomical calculations done in Part II. We use exact rational arithmetic, with very large numbers, for the Hindu calendars; 64-bit arithmetic can be used to approximate their calculation.

We use degree-based trigonometric functions throughout for simplicity. For programming languages in which these functions are radian-based, conversions are necessary—see our Lisp code (page 513), for an example of such conversions.

Finally, floating point calculations are platform-dependent. The values given for the sample data of Appendix C will differ slightly for different languages, implementations, or platforms. Double precision is necessary, however, for accurate results; furthermore, in some cases, low precision results when low- and high-precision real numbers are combined—this can have seriously deleterious effects. To avoid such problems, all real numbers should have maximal (double) precision, as they have in our Lisp code. For details, see the introduction to Appendix C.

References

I have, however, read enough in the field to know that many of these treatments of the calendar are sound, some of them brilliant, and some purely fantastic. I also know that practically every theory about the calendar which could conceivably have been devised has been proposed by somebody, and that many have been re-invented several times. I know that very little of what I have to say has not been anticipated . . .

Agnes K. Michels: *The Calendar of the Roman Republic* (1967)

[1] *Explanatory Supplement to the Astronomical Ephemeris and the American Ephemeris and Nautical Almanac*, Her Majesty's Stationery Office, London, 1961.

[2] Goddard Earth Sciences Data Information Services Center website, National Aeronautics and Space Administration, `disc.sci.gsfc.nasa.gov/julian_calendar.html`.

[3] *The Larousse Encyclopedia of Mythology*, Barnes & Noble Books, New York, 1994.

[4] J. P. Allen, *Middle Egyptian*: *An Introduction to the Language and Culture of Hieroglyphs*, Cambridge University Press, Cambridge, 2nd edn., pp. 107–110, 2010.

[5] P. F. W. Bartle, "The Forty Days: The Akan Calendar," *Africa*, vol. 48, pp. 80–84, 1978.

[6] B. Blackburn and L. Holford-Strevens, *The Oxford Companion to the Year*, Oxford University Press, Oxford, 1999.

[7] C. A. Boch, B. Ananthasubramaniam, A. M. Sweeney, F. J. Doyle, and D. E. Morse, "Effects of Light Dynamics on Coral Spawning Synchrony," *Biological Bulletin*, vol. 220, pp. 161–173, 2011.

[8] M. Clagett, *Ancient Egyptian Science*: vol. 2, *Calendars, Clocks, and Astronomy*, American Philosophical Society, Philadelphia, 1995.

[9] L. Cope, "Calendars of the Indians North of Mexico," *American Archaeology and Ethnology*, vol. 16, pp. 119–176, 1919.

[10] J. Davenport, T. T. Jones, T. M. Work, and G. H. Balazs, "Pink Spot, White Spot: The Pineal Skylight of the Leatherback Turtle (*Dermochelys Coriacea* Vandelli 1761) Skull and its Possible Role in the Phenology of Feeding Migrations," *Journal of Experimental Marine Biology and Ecology*, vol. 461, pp. 1–6, 2014.

[11] N. Dershowitz and E. M. Reingold, "Modulo Intervals: A Proposed Notation," *ACM SIGACT News*, vol. 43, no. 3, pp. 60–64, 2012.

[12] L. E. Doggett, "Calendars," *Explanatory Supplement to the Astronomical Almanac*, P. K. Seidelmann, ed., University Science Books, Mill Valley, CA, pp. 575–608, 1992.

[13] J. Elkins, "On the Impossibility of Close Reading," *Current Anthropology*, vol. 37, pp. 185–226, 1996.

[14] N. Fox, "A Bone Carved Calendar," in *Tel 'Aroer: An Iron Age II Caravan Town and the Hellenistic–Early Roman Settlement. Avraham Biran (1975–1982) and Rudolph Cohen (1975–1976) Excavations*, Y. Thareani, ed., Annual of the Nelson Glueck School of Biblical Archaeology, No. VIII, Jerusalem, pp. 255–258, 468, 2011.

[15] T. Galloway and W. S. B. Woolhouse, "Calendar," *The Encyclopædia Britannica*, 11th edn., vol. 4, pp. 987–1004, The Encyclopædia Britannica Co., New York, 1910. The same article also appears in the 8th (1860) through the 13th (1926) editions.

[16] F. K. Ginzel, *Handbuch der mathematischen und technischen Chronologie*, J. C. Hinrichs'sche Buchhandlung, Leipzig, 1906 (vol. 1), 1911 (vol. 2), and 1914 (vol. 3). Reprinted by F. Ullmann Verlag, Zwickau, 1958.

[17] A. T. Grafton, *Joseph Scaliger*: *A Study in the History of Classical Scholarship, vol. II, Historical Chronography*, Oxford University Press, Oxford, 1993.

[18] R. L. Graham, D. E. Knuth, and O. Patashnik, *Concrete Mathematics*, 2nd edn., Addison-Wesley Publishing Company, Reading, MA, 1994.

[19] G. H. Hardy, and E. M. Wright, *An Introduction to the Theory of Numbers*, 5th edn., Oxford University Press, Oxford, 1979.

[20] M. A. Harris and E. M. Reingold, "Line Drawing, Leap Years, and Euclid," *ACM Computing Surveys*, vol. 36, pp. 68–80, 2004.

[21] O. L. Harvey, *Calendar Conversions by Way of the Julian Day Number*, American Philosophical Society, Philadelphia, 1983.

[22] J. Hastings, edn., *Encyclopædia of Religion and Ethics*, Charles Scribner's Sons, New York, 1908–1922.

[23] H. Henderson and B. Puckett, *Holidays & Festivals Index*, Omnigraphics, Detroit, MI, 1995.

[24] H. Henderson and S. E. Thompson, *Holidays, Festivals & Celebrations of the World Dictionary*, 2nd edn., Omnigraphics, Detroit, MI, 1997.

[25] J. F. W. Herschel, *Outlines of Astronomy*, 3rd edn., Longman, Brown, Green, Longmans, and Roberts, London, 1849.

[26] W. Horowitz, “The 360 and 364 Day Year in Ancient Mesopotamia,” *Journal of the Ancient Near Eastern Society* vol. 24 (1996), pp. 35–41.

[27] A. Ya. Khinchin, *Continued Fractions*, translated by P. Wynn, P. Noordhoff, Groningen, 1964. Reprinted by Dover Publications, Mineola, NY, 1997.

[28] D. E. Knuth, *The Art of Computer Programming*, vol. 2: *Seminumerical Algorithms*, 3rd edn., Addison-Wesley Publishing Company, Reading, MA, 1998.

[29] K. Konadu, “The Calendrical Factor in Akan History,” *International Journal of African Historical Studies* vol. 45, pp. 217–246, 2012.

[30] N. Kronfeld-Schor, D. Dominoni, H. de la Iglesia, O. Levy, E. D. Herzog, T. Dayan, and C. Helfrich-Forster, “Chronobiology by Moonlight,” *Proc. Royal Soc. B* vol. 280, 20123088, 2013.

[31] F. Macler, “Calendar (Armenian),” vol. III, pp. 70–72, *Encyclopædia of Religion and Ethics*, J. Hastings, edn., Charles Scribner’s Sons, New York, 1908–1922.

[32] A. Marshack, *The Roots of Civilization*, McGraw-Hill, New York, 1972.

[33] O. Neugebauer, *A History of Ancient Mathematical Astronomy*, Springer-Verlag, Berlin, 1975 (vol. 1, pp. 1–555, vol. 2, pp. 556–1058, vol. 3, pp. 1059–1457).

[34] Ø. Ore, *Number Theory and Its History*, McGraw-Hill., New York, 1948. Reprinted by Dover, Mineola, NY, 1987.

[35] O. K. Osei, *A Discourse on Akan Perpetual Calendar* (*for Religious Ceremonies and Festivals*), Domak Press Ltd., Accra, Ghana, 1997.

[36] F. Parise, ed., *The Book of Calendars*, Facts on File, New York, 1982.

[37] R. A. Parker, *The Calendars of Ancient Egypt*, University of Chicago Press, Chicago, 1950.

[38] R. L. Reese, E. D. Craun, and C. W. Mason, “Twelfth-Century Origins of the 7980-Year Julian Period,” *Amer. J. Physics*, vol. 51, p. 73, 1983.

[39] R. L. Reese, S. M. Everett, and E. D. Craun, “The Origin of the Year Julian Period: An Application of Congruences and the Chinese Remainder Theorem,” *Amer. J. Physics*, vol. 49, pp. 658–661, 1981.

[40] E. G. Richards, *Mapping Time: The Calendar and its History*, Oxford University Press, Oxford, 1998.

[41] R. G. Schram, *Kalendariographische und chronologische Tafeln*, J. C. Hinrichs'sche Buchhandlung, Leipzig, 1908.

[42] R. D. Skeel and J. B. Keiper, *Elementary Numerical Computing with Mathematica*, McGraw-Hill, New York, 1993.

[43] J. Speicher, L. Schreffler, and D. Speicher, "Lunar Influence on the Fall Migration of Northern Saw-whet Owls," *Wilson J. Ornithology*, vol. 123, pp. 158–160, 2011.

[44] R. M. Stallman, *GNU Emacs Manual*, 13th edn., Free Software Foundation, Cambridge, MA, 1997.

[45] D. Steel, *Marking Time*: *The Epic Quest to Invent the Perfect Calendar*, John Wiley & Sons, New York, 2000.

[46] G. L. Steele, Jr., Common LISP: *The Language*, 2nd edn., Digital Press, Bedford, MA, 1990.

[47] S. H. Taqizadeh, "An Ancient Persian Practice Preserved by a Non-Iranian People," *Bulletin of the School of Oriental Studies*, vol. 9, no. 3, pp. 603–619, 1938.

[48] A. Troesch, "Interprétation géométrique de l'algorithme d'Euclide et reconnaissance de segments," *Theoret. Comp. Sci.*, vol. 115, pp. 291–319, 1993.

[49] B. L. van der Waerden, "Tables for the Egyptian and Alexandrian Calendar," *ISIS*, vol. 47, pp. 387–390, 1956.

If you steal from one author it's plagiarism; if you steal from many, it's research.
Attributed to Wilson Mizner