Concrete Mathematics

Graham, Knuth, Patashnik

April 2022

Contents

1 Sums

1.1 Notation

2.1 NOTATION

In Chapter 1 we encountered the sum of the first n integers, which we wrote out as $1+2+3+\cdots+(n-1)+n$. The '$\cdots$' in such formulas tells us to complete the pattern established by the surrounding terms. Of course we have to watch out for sums like $1+7+\cdots+41.7$, which are meaningless without a mitigating context. On the other hand, the inclusion of terms like 3 and $(n-1)$ was a bit of overkill; the pattern would presumably have been clear if we had written simply $1+2+\cdots+n$. Sometimes we might even be so bold as to write just $1+\cdots+n$.

We'll be working with sums of the general form

$$a_1 + a_2 + \cdots + a_n\,, \tag{2.1}$$

where each a_k is a number that has been defined somehow. This notation has the advantage that we can "see" the whole sum, almost as if it were written out in full, if we have a good enough imagination.

Each element a_k of a sum is called a *term*. The terms are often specified implicitly as formulas that follow a readily perceived pattern, and in such cases we must sometimes write them in an expanded form so that the meaning is clear. For example, if

$$1+2+\cdots+2^{n-1}$$

is supposed to denote a sum of n terms, not of 2^{n-1}, we should write it more explicitly as

$$2^0+2^1+\cdots+2^{n-1}.$$

The three-dots notation has many uses, but it can be ambiguous and a bit long-winded. Other alternatives are available, notably the delimited form

$$\sum_{k=1}^{n} a_k \,, \tag{2.2}$$

which is called Sigma-notation because it uses the Greek letter $\sum$ (upper-case sigma). This notation tells us to include in the sum precisely those terms a_k whose index k is an integer that lies between the lower and upper limits 1 and n, inclusive. In words, we "sum over k, from 1 to n." Joseph Fourier introduced this delimited $\sum$-notation in 1820, and it soon took the mathematical world by storm.

Incidentally, the quantity after $\sum$ (here a_k) is called the *summand*.

The index variable k is said to be *bound* to the $\sum$ sign in (2.2), because the k in a_k is unrelated to appearances of k outside the Sigma-notation. Any other letter could be substituted for k here without changing the meaning of (2.2). The letter i is often used (perhaps because it stands for "index"), but we'll generally sum on k since it's wise to keep i for $\sqrt{-1}$.

It turns out that a generalized Sigma-notation is even more useful than the delimited form: We simply write one or more conditions under the $\sum$, to specify the set of indices over which summation should take place. For example, the sums in (2.1) and (2.2) can also be written as

$$\sum_{1 \leqslant k \leqslant n} a_k \,. \tag{2.3}$$

In this particular example there isn't much difference between the new form and (2.2), but the general form allows us to take sums over index sets that aren't restricted to consecutive integers. For example, we can express the sum of the squares of all odd positive integers below 100 as follows:

$$\sum_{\substack{1 \leqslant k < 100 \\ k \text{ odd}}} k^2 \,.$$

The delimited equivalent of this sum,

$$\sum_{k=0}^{49} (2k+1)^2 \,,$$

The biggest advantage of general Sigma-notation is that we can manipulate it more easily than the delimited form. For example, suppose we want to change the index variable k to $k+1$. With the general form, we have

$$\sum_{1\leqslant k\leqslant n} a_k \;=\; \sum_{1\leqslant k+1\leqslant n} a_{k+1}\,;$$

it's easy to see what's going on, and we can do the substitution almost without thinking. But with the delimited form, we have

$$\sum_{k=1}^{n} a_k \;=\; \sum_{k=0}^{n-1} a_{k+1}\,;$$

it's harder to see what's happened, and we're more likely to make a mistake.

On the other hand, the delimited form isn't completely useless. It's nice and tidy, and we can write it quickly because (2.2) has seven symbols compared with (2.3)'s eight. Therefore we'll often use $\sum$ with upper and lower delimiters when we state a problem or present a result, but we'll prefer to work with relations-under-$\sum$ when we're manipulating a sum whose index variables need to be transformed.

The $\sum$ sign occurs more than 1000 times in this book, so we should be sure that we know exactly what it means. Formally, we write

$$\sum_{P(k)} a_k \tag{2.4}$$

as an abbreviation for the sum of all terms a_k such that k is an integer satisfying a given property $P(k)$. (A "property $P(k)$" is any statement about k that can be either true or false.) For the time being, we'll assume that only finitely many integers k satisfying $P(k)$ have $a_k \neq 0$; otherwise infinitely many nonzero numbers are being added together, and things can get a bit

tricky. At the other extreme, if $P(k)$ is false for all integers k, we have an "empty" sum; the value of an empty sum is defined to be zero.

A slightly modified form of (2.4) is used when a sum appears within the text of a paragraph rather than in a displayed equation: We write '$\sum_{P(k)} a_k$', attaching property $P(k)$ as a subscript of $\sum$, so that the formula won't stick out too much. Similarly, '$\sum_{k=1}^{n} a_k$' is a convenient alternative to (2.2) when we want to confine the notation to a single line.

People are often tempted to write

$$\sum_{k=2}^{n-1} k(k-1)(n-k) \qquad \text{instead of} \qquad \sum_{k=0}^{n} k(k-1)(n-k)$$

because the terms for $k = 0$, 1, and n in this sum are zero. Somehow it seems more efficient to add up $n-2$ terms instead of $n+1$ terms. But such temptations should be resisted; efficiency of computation is not the same as efficiency of understanding! We will find it advantageous to keep upper and lower bounds on an index of summation as simple as possible, because sums can be manipulated much more easily when the bounds are simple. Indeed, the form $\sum_{k=2}^{n-1}$ can even be dangerously ambiguous, because its meaning is not at all clear when $n = 0$ or $n = 1$ (see exercise 1). Zero-valued terms cause no harm, and they often save a lot of trouble.

So far the notations we've been discussing are quite standard, but now we are about to make a radical departure from tradition. Kenneth E. Iverson introduced a wonderful idea in his programming language APL [191, page 11; see also 220], and we'll see that it greatly simplifies many of the things we want to do in this book. The idea is simply to enclose a true-or-false statement in brackets, and to say that the result is 1 if the statement is true, 0 if the statement is false. For example,

$$[p \text{ prime}] = \begin{cases} 1, & \text{if } p \text{ is a prime number;} \\ 0, & \text{if } p \text{ is not a prime number.} \end{cases}$$

Iverson's convention allows us to express sums with no constraints whatever on the index of summation, because we can rewrite (2.4) in the form

$$\sum_k a_k [P(k)] \,. \tag{2.5}$$

If $P(k)$ is false, the term $a_k[P(k)]$ is zero, so we can safely include it among the terms being summed. This makes it easy to manipulate the index of summation, because we don't have to fuss with boundary conditions.

A slight technicality needs to be mentioned: Sometimes a_k isn't defined for all integers k. We get around this difficulty by assuming that $[P(k)]$ is "very strongly zero" when $P(k)$ is false; it's so much zero, it makes $a_k[P(k)]$

equal to zero even when a_k is undefined. For example, if we use Iverson's convention to write the sum of reciprocal primes $\leqslant N$ as

$$\sum_p [p \text{ prime}][p \leqslant N]/p\,,$$

there's no problem of division by zero when $p = 0$, because our convention tells us that $[0 \text{ prime}][0 \leqslant N]/0 = 0$.

Let's sum up what we've discussed so far about sums. There are two good ways to express a sum of terms: One way uses '$\cdots$', the other uses '$\sum$'. The three-dots form often suggests useful manipulations, particularly the combination of adjacent terms, since we might be able to spot a simplifying pattern if we let the whole sum hang out before our eyes. But too much detail can also be overwhelming. Sigma-notation is compact, impressive to family and friends, and often suggestive of manipulations that are not obvious in three-dots form. When we work with Sigma-notation, zero terms are not generally harmful; in fact, zeros often make $\sum$-manipulation easier.

1.2 Manipulation Of Sums

2.3 MANIPULATION OF SUMS

The key to success with sums is an ability to change one $\sum$ into another that is simpler or closer to some goal. And it's easy to do this by learning a few basic rules of transformation and by practicing their use.

Let K be any finite set of integers. Sums over the elements of K can be transformed by using three simple rules:

$$\sum_{k\in K} ca_k = c\sum_{k\in K} a_k\,; \qquad \text{(distributive law)} \qquad (2.15)$$

$$\sum_{k\in K}(a_k+b_k) = \sum_{k\in K} a_k + \sum_{k\in K} b_k\,; \qquad \text{(associative law)} \qquad (2.16)$$

$$\sum_{k\in K} a_k = \sum_{p(k)\in K} a_{p(k)}\,. \qquad \text{(commutative law)} \qquad (2.17)$$

The distributive law allows us to move constants in and out of a $\sum$. The associative law allows us to break a $\sum$ into two parts, or to combine two $\sum$'s into one. The commutative law says that we can reorder the terms in any way we please; here $p(k)$ is any permutation of the set of all integers. For example, if $K=\{-1,0,+1\}$ and if $p(k)=-k$, these three laws tell us respectively that

$$ca_{-1}+ca_0+ca_1 = c(a_{-1}+a_0+a_1)\,; \qquad \text{(distributive law)}$$

$$(a_{-1}+b_{-1})+(a_0+b_0)+(a_1+b_1) = (a_{-1}+a_0+a_1)+(b_{-1}+b_0+b_1)\,; \qquad \text{(associative law)}$$

$$a_{-1}+a_0+a_1 = a_1+a_0+a_{-1}\,. \qquad \text{(commutative law)}$$

Gauss's trick in Chapter 1 can be viewed as an application of these three basic laws. Suppose we want to compute the general sum of an *arithmetic progression*,

$$S = \sum_{0\leqslant k\leqslant n}(a+bk)\,.$$

By the commutative law we can replace k by $n-k$, obtaining

$$S = \sum_{0\leqslant n-k\leqslant n}(a+b(n-k)) = \sum_{0\leqslant k\leqslant n}(a+bn-bk)\,.$$

These two equations can be added by using the associative law:

$$2S = \sum_{0\leqslant k\leqslant n}((a+bk)+(a+bn-bk)) = \sum_{0\leqslant k\leqslant n}(2a+bn)\,.$$

And we can now apply the distributive law and evaluate a trivial sum:

$$2S \;=\; (2a+bn)\sum_{0\leqslant k\leqslant n} 1 \;=\; (2a+bn)(n+1)\,.$$

Dividing by 2, we have proved that

$$\sum_{k=0}^{n}(a+bk) \;=\; (a+\tfrac{1}{2}bn)(n+1)\,. \tag{2.18}$$

The right-hand side can be remembered as the average of the first and last terms, namely $\frac{1}{2}\big(a+(a+bn)\big)$, times the number of terms, namely $(n+1)$.

It's important to bear in mind that the function $p(k)$ in the general commutative law (2.17) is supposed to be a permutation of all the integers. In other words, for every integer n there should be exactly one integer k such that $p(k) = n$. Otherwise the commutative law might fail; exercise 3 illustrates

1.3 Multiple Sums

2.4 MULTIPLE SUMS

The terms of a sum might be specified by two or more indices, not just by one. For example, here's a double sum of nine terms, governed by two indices j and k:

$$\sum_{1\leqslant j,k\leqslant 3} a_j b_k = \begin{aligned} &a_1b_1 + a_1b_2 + a_1b_3 \\ &+ a_2b_1 + a_2b_2 + a_2b_3 \\ &+ a_3b_1 + a_3b_2 + a_3b_3\,. \end{aligned}$$

We use the same notations and methods for such sums as we do for sums with a single index. Thus, if $P(j,k)$ is a property of j and k, the sum of all terms $a_{j,k}$ such that $P(j,k)$ is true can be written in two ways, one of which uses Iverson's convention and sums over *all* pairs of integers j and k:

$$\sum_{P(j,k)} a_{j,k} = \sum_{j,k} a_{j,k}\,[P(j,k)]\,.$$

Only one $\sum$ sign is needed, although there is more than one index of summation; $\sum$ denotes a sum over all combinations of indices that apply.

We also have occasion to use two $\sum$'s, when we're talking about a sum of sums. For example,

$$\sum_j \sum_k a_{j,k}\,[P(j,k)]$$

is an abbreviation for

$$\sum_j \Bigg(\sum_k a_{j,k}\,[P(j,k)]\Bigg)\,,$$

which is the sum, over all integers j, of $\sum_k a_{j,k}\,[P(j,k)]$, the latter being the sum over all integers k of all terms $a_{j,k}$ for which $P(j,k)$ is true. In such cases we say that the double sum is "summed first on k." A sum that depends on more than one index can be summed first on any one of its indices.

In this regard we have a basic law called *interchanging the order of summation*, which generalizes the associative law (2.16) we saw earlier:

$$\sum_j \sum_k a_{j,k}[P(j,k)] = \sum_{P(j,k)} a_{j,k} = \sum_k \sum_j a_{j,k}[P(j,k)]\,. \qquad (2.27)$$

The middle term of this law is a sum over two indices. On the left, $\sum_j \sum_k$ stands for summing first on k, then on j. On the right, $\sum_k \sum_j$ stands for summing first on j, then on k. In practice when we want to evaluate a double sum in closed form, it's usually easier to sum it first on one index rather than on the other; we get to choose whichever is more convenient.

1.4 General Methods

2.5 GENERAL METHODS

Now let's consolidate what we've learned, by looking at a single example from several different angles. On the next few pages we're going to try to find a closed form for the sum of the first n squares, which we'll call $\square_n$:

$$\square_n = \sum_{0 \leqslant k \leqslant n} k^2, \qquad \text{for } n \geqslant 0. \tag{2.37}$$

We'll see that there are at least seven different ways to solve this problem, and in the process we'll learn useful strategies for attacking sums in general.

First, as usual, we look at some small cases.

n	0	1	2	3	4	5	6	7	8	9	10	11	12
n^2	0	1	4	9	16	25	36	49	64	81	100	121	144
$\square_n$	0	1	5	14	30	55	91	140	204	285	385	506	650

No closed form for $\square_n$ is immediately evident; but when we do find one, we can use these values as a check.

Method 0: You could look it up.

A problem like the sum of the first n squares has probably been solved before, so we can most likely find the solution in a handy reference book. Sure enough, page 36 of the *CRC Standard Mathematical Tables* [28] has the answer:

$$\square_n = \frac{n(n+1)(2n+1)}{6}, \qquad \text{for } n \geqslant 0. \tag{2.38}$$

Just to make sure we haven't misread it, we check that this formula correctly gives $\square_5 = 5 \cdot 6 \cdot 11/6 = 55$. Incidentally, page 36 of the *CRC Tables* has further information about the sums of cubes, ..., tenth powers.

The definitive reference for mathematical formulas is the *Handbook of Mathematical Functions*, edited by Abramowitz and Stegun [2]. Pages 813–814 of that book list the values of $\square_n$ for $n \leqslant 100$; and pages 804 and 809 exhibit formulas equivalent to (2.38), together with the analogous formulas for sums of cubes, ..., fifteenth powers, with or without alternating signs.

But the best source for answers to questions about sequences is an amazing little book called the *Handbook of Integer Sequences*, by Sloane [330], which lists thousands of sequences by their numerical values. If you come up with a recurrence that you suspect has already been studied, all you have to do is compute enough terms to distinguish your recurrence from other famous ones; then chances are you'll find a pointer to the relevant literature in Sloane's *Handbook*. For example, 1, 5, 14, 30, ... turns out to be Sloane's sequence number 1574, and it's called the sequence of "square pyramidal numbers" (because there are $\square_n$ balls in a pyramid that has a square base of n^2 balls). Sloane gives three references, one of which is to the handbook of Abramowitz and Stegun that we've already mentioned.

Still another way to probe the world's store of accumulated mathematical wisdom is to use a computer program (such as Axiom, MACSYMA, Maple, or Mathematica) that provides tools for symbolic manipulation. Such programs are indispensable, especially for people who need to deal with large formulas.

It's good to be familiar with standard sources of information, because they can be extremely helpful. But Method 0 isn't really consistent with the spirit of this book, because we want to know how to figure out the answers

by ourselves. The look-up method is limited to problems that other people have decided are worth considering; a new problem won't be there.

Method 1: Guess the answer, prove it by induction.

Perhaps a little bird has told us the answer to a problem, or we have arrived at a closed form by some other less-than-rigorous means. Then we merely have to prove that it is correct.

We might, for example, have noticed that the values of $\square_n$ have rather small prime factors, so we may have come up with formula (2.38) as something that works for all small values of n. We might also have conjectured the equivalent formula

$$\square_n = \frac{n(n+\frac{1}{2})(n+1)}{3}, \qquad \text{for } n \geqslant 0, \tag{2.39}$$

which is nicer because it's easier to remember. The preponderance of the evidence supports (2.39), but we must prove our conjectures beyond all reasonable doubt. Mathematical induction was invented for this purpose.

"Well, Your Honor, we know that $\square_0 = 0 = 0(0+\frac{1}{2})(0+1)/3$, so the basis is easy. For the induction, suppose that $n > 0$, and assume that (2.39) holds when n is replaced by $n-1$. Since

$$\square_n = \square_{n-1} + n^2,$$

we have

$$\begin{aligned} 3\square_n &= (n-1)(n-\tfrac{1}{2})(n) \;+\; 3n^2 \\ &= (n^3 - \tfrac{3}{2}n^2 + \tfrac{1}{2}n) \;+\; 3n^2 \\ &= (n^3 + \tfrac{3}{2}n^2 + \tfrac{1}{2}n) \\ &= n(n+\tfrac{1}{2})(n+1)\,. \end{aligned}$$

Therefore (2.39) indeed holds, beyond a reasonable doubt, for all $n \geqslant 0$." Judge Wapner, in his infinite wisdom, agrees.

Induction has its place, and it is somewhat more defensible than trying to look up the answer. But it's still not really what we're seeking. All of the other sums we have evaluated so far in this chapter have been conquered without induction; we should likewise be able to determine a sum like $\square_n$ from scratch. Flashes of inspiration should not be necessary. We should be able to do sums even on our less creative days.

Method 2: Perturb the sum.

So let's go back to the perturbation method that worked so well for the geometric progression (2.25). We extract the first and last terms of $\square_{n+1}$ in

order to get an equation for $\square_n$:

$$\begin{aligned}\square_n + (n+1)^2 \;&=\; \sum_{0\leqslant k\leqslant n} (k+1)^2 \;=\; \sum_{0\leqslant k\leqslant n} (k^2+2k+1)\\ &= \sum_{0\leqslant k\leqslant n} k^2 + 2\sum_{0\leqslant k\leqslant n} k + \sum_{0\leqslant k\leqslant n} 1\\ &= \square_n \;+\; 2\sum_{0\leqslant k\leqslant n} k \;+\; (n+1)\,.\end{aligned}$$

Oops—the $\square_n$'s cancel each other. Occasionally, despite our best efforts, the perturbation method produces something like $\square_n = \square_n$, so we lose.

On the other hand, this derivation is not a total loss; it does reveal a way to sum the first n integers in closed form,

$$2\sum_{0\leqslant k\leqslant n} k \;=\; (n+1)^2 - (n+1)\,,$$

even though we'd hoped to discover the sum of first integers squared. Could it be that if we start with the sum of the integers cubed, which we might call $\square\!\!\square_n$, we will get an expression for the integers squared? Let's try it.

$$\begin{aligned}\square\!\!\square_n + (n+1)^3 \;&=\; \sum_{0\leqslant k\leqslant n} (k+1)^3 \;=\; \sum_{0\leqslant k\leqslant n} (k^3+3k^2+3k+1)\\ &= \square\!\!\square_n + 3\square_n + 3\tfrac{(n+1)n}{2} + (n+1)\,.\end{aligned}$$

Sure enough, the $\square\!\!\square_n$'s cancel, and we have enough information to determine $\square_n$ without relying on induction:

$$\begin{aligned}3\square_n \;&=\; (n+1)^3 - 3(n+1)n/2 - (n+1)\\ &=\; (n+1)(n^2+2n+1-\tfrac{3}{2}n-1) \;=\; (n+1)(n+\tfrac{1}{2})n\,.\end{aligned}$$

Method 3: Build a repertoire.

A slight generalization of the recurrence (2.7) will also suffice for summands involving n^2. The solution to

$$\begin{aligned}&R_0 = \alpha;\\ &R_n = R_{n-1} + \beta + \gamma n + \delta n^2\,, \qquad \text{for } n > 0,\end{aligned} \tag{2.40}$$

will be of the general form

$$R_n \;=\; A(n)\alpha + B(n)\beta + C(n)\gamma + D(n)\delta\,; \tag{2.41}$$

and we have already determined $A(n)$, $B(n)$, and $C(n)$, because (2.40) is the same as (2.7) when $\delta = 0$. If we now plug in $R_n = n^3$, we find that n^3 is the

solution when $\alpha = 0$, $\beta = 1$, $\gamma = -3$, $\delta = 3$. Hence

$$3D(n) - 3C(n) + B(n) \;=\; n^3\,;$$

this determines $D(n)$.

We're interested in the sum $\square_n$, which equals $\square_{n-1} + n^2$; thus we get $\square_n = R_n$ if we set $\alpha = \beta = \gamma = 0$ and $\delta = 1$ in (2.41). Consequently $\square_n = D(n)$. We needn't do the algebra to compute $D(n)$ from $B(n)$ and $C(n)$, since we already know what the answer will be; but doubters among us should be reassured to find that

$$3D(n) \;=\; n^3 + 3C(n) - B(n) \;=\; n^3 + 3\frac{(n+1)n}{2} - n \;=\; n(n+\tfrac{1}{2})(n+1)\,.$$

2 Integer Functions

2.1 Floors and Ceilings

3.1 FLOORS AND CEILINGS

We start by covering the floor (greatest integer) and ceiling (least integer) functions, which are defined for all real x as follows:

$$\begin{aligned} \lfloor x \rfloor &= \text{the greatest integer less than or equal to } x; \\ \lceil x \rceil &= \text{the least integer greater than or equal to } x. \end{aligned} \tag{3.1}$$

To actually prove properties about the floor and ceiling functions, rather than just to observe such facts graphically, the following four rules are especially useful:

$$\begin{aligned}
\lfloor x\rfloor = n \quad &\iff \quad n \leqslant x < n+1\,, && \text{(a)}\\
\lfloor x\rfloor = n \quad &\iff \quad x-1 < n \leqslant x\,, && \text{(b)}\\
\lceil x\rceil = n \quad &\iff \quad n-1 < x \leqslant n\,, && \text{(c)}\\
\lceil x\rceil = n \quad &\iff \quad x \leqslant n < x+1\,. && \text{(d)}
\end{aligned} \tag{3.5}$$

(We assume in all four cases that n is an integer and that x is real.) Rules (a) and (c) are immediate consequences of definition (3.1); rules (b) and (d) are the same but with the inequalities rearranged so that n is in the middle.

It's possible to move an integer term in or out of a floor (or ceiling):

$$\lfloor x+n\rfloor \;=\; \lfloor x\rfloor + n\,, \qquad \text{integer } n. \tag{3.6}$$

(Because rule (3.5(a)) says that this assertion is equivalent to the inequalities $\lfloor x\rfloor + n \leqslant x+n < \lfloor x\rfloor + n + 1$.) But similar operations, like moving out a constant factor, cannot be done in general. For example, we have $\lfloor nx\rfloor \neq n\lfloor x\rfloor$ when $n = 2$ and $x = 1/2$. This means that floor and ceiling brackets are comparatively inflexible. We are usually happy if we can get rid of them or if we can prove anything at all when they are present.

It turns out that there are many situations in which floor and ceiling brackets are redundant, so that we can insert or delete them at will. For example, any inequality between a real and an integer is equivalent to a floor or ceiling inequality between integers:

$$\begin{aligned}
x < n \quad &\iff \quad \lfloor x\rfloor < n\,, && \text{(a)}\\
n < x \quad &\iff \quad n < \lceil x\rceil\,, && \text{(b)}\\
x \leqslant n \quad &\iff \quad \lceil x\rceil \leqslant n\,, && \text{(c)}\\
n \leqslant x \quad &\iff \quad n \leqslant \lfloor x\rfloor\,. && \text{(d)}
\end{aligned} \tag{3.7}$$

These rules are easily proved. For example, if $x < n$ then surely $\lfloor x\rfloor < n$, since $\lfloor x\rfloor \leqslant x$. Conversely, if $\lfloor x\rfloor < n$ then we must have $x < n$, since $x < \lfloor x\rfloor + 1$ and $\lfloor x\rfloor + 1 \leqslant n$.

2.2 'MOD': The binary operation

3.4 'MOD': THE BINARY OPERATION

The quotient of n divided by m is $\lfloor n/m \rfloor$, when m and n are positive integers. It's handy to have a simple notation also for the remainder of this

division, and we call it '$n \bmod m$'. The basic formula

$$n = m\underbrace{\lfloor n/m \rfloor}_{\text{quotient}} + \underbrace{n \bmod m}_{\text{remainder}}$$

tells us that we can express $n \bmod m$ as $n - m\lfloor n/m \rfloor$. We can generalize this to negative integers, and in fact to arbitrary real numbers:

$$x \bmod y = x - y\lfloor x/y \rfloor, \qquad \text{for } y \neq 0. \tag{3.21}$$

This defines 'mod' as a binary operation, just as addition and subtraction are binary operations. Mathematicians have used mod this way informally for a long time, taking various quantities mod 10, mod 2π, and so on, but only in the last twenty years has it caught on formally. Old notion, new notation.

We can easily grasp the intuitive meaning of $x \bmod y$, when x and y are positive real numbers, if we imagine a circle of circumference y whose points have been assigned real numbers in the interval $[0\,..\,y)$. If we travel a distance x around the circle, starting at 0, we end up at $x \bmod y$. (And the number of times we encounter 0 as we go is $\lfloor x/y \rfloor$.)

When x or y is negative, we need to look at the definition carefully in order to see exactly what it means. Here are some integer-valued examples:

$$\begin{aligned}
5 \bmod 3 &= 5 - 3\lfloor 5/3 \rfloor &&= 2;\\
5 \bmod -3 &= 5 - (-3)\lfloor 5/(-3) \rfloor &&= -1;\\
-5 \bmod 3 &= -5 - 3\lfloor -5/3 \rfloor &&= 1;\\
-5 \bmod -3 &= -5 - (-3)\lfloor -5/(-3) \rfloor &&= -2.
\end{aligned}$$

The number after 'mod' is called the *modulus*; nobody has yet decided what to call the number before 'mod'. In applications, the modulus is usually positive, but the definition makes perfect sense when the modulus is negative. In both cases the value of $x \bmod y$ is between 0 and the modulus:

$$\begin{aligned}
0 &\leqslant x \bmod y < y, \qquad &&\text{for } y > 0;\\
0 &\geqslant x \bmod y > y, \qquad &&\text{for } y < 0.
\end{aligned}$$

What about $y = 0$? Definition (3.21) leaves this case undefined, in order to avoid division by zero, but to be complete we can define

$$x \bmod 0 = x. \tag{3.22}$$

This convention preserves the property that $x \bmod y$ always differs from x by a multiple of y. (It might seem more natural to make the function continuous at 0, by defining $x \bmod 0 = \lim_{y \to 0} x \bmod y = 0$. But we'll see in Chapter 4

that this would be much less useful. Continuity is not an important aspect of the mod operation.)

We've already seen one special case of mod in disguise, when we wrote x in terms of its integer and fractional parts, $x = \lfloor x \rfloor + \{x\}$. The fractional part can also be written $x \bmod 1$, because we have

$$x = \lfloor x \rfloor + x \bmod 1 .$$

Notice that parentheses aren't needed in this formula; we take mod to bind more tightly than addition or subtraction.

3 Number Theory

3.1 Divisiblity

4.1 DIVISIBILITY

We say that m divides n (or n is divisible by m) if $m > 0$ and the ratio n/m is an integer. This property underlies all of number theory, so it's convenient to have a special notation for it. We therefore write

$$m \backslash n \quad \Longleftrightarrow \quad m > 0 \text{ and } n = mk \text{ for some integer } k . \qquad (4.1)$$

If m does not divide n we write '$m \nmid n$'.

There's a similar relation, "n is a multiple of m," which means almost the same thing except that m doesn't have to be positive. In this case we simply mean that $n = mk$ for some integer k. Thus, for example, there's only one multiple of 0 (namely 0), but nothing is divisible by 0. Every integer is a multiple of -1, but no integer is divisible by -1 (strictly speaking). These definitions apply when m and n are any real numbers; for example, 2π is divisible by π. But we'll almost always be using them when m and n are integers. After all, this is number theory.

The *greatest common divisor* of two integers m and n is the largest integer that divides them both:

$$\gcd(m,n) = \max\{k \mid k\backslash m \quad \text{and} \quad k\backslash n\}. \tag{4.2}$$

For example, $\gcd(12,18) = 6$. This is a familiar notion, because it's the common factor that fourth graders learn to take out of a fraction m/n when reducing it to lowest terms: $12/18 = (12/6)/(18/6) = 2/3$. Notice that if $n > 0$ we have $\gcd(0,n) = n$, because any positive number divides 0, and because n is the largest divisor of itself. The value of $\gcd(0,0)$ is undefined.

Another familiar notion is the *least common multiple*,

$$\operatorname{lcm}(m,n) = \min\{k \mid k > 0, \quad m\backslash k \quad \text{and} \quad n\backslash k\}; \tag{4.3}$$

this is undefined if $m \leqslant 0$ or $n \leqslant 0$. Students of arithmetic recognize this as the least common denominator, which is used when adding fractions with denominators m and n. For example, $\operatorname{lcm}(12,18) = 36$, and fourth graders know that $\frac{7}{12} + \frac{1}{18} = \frac{21}{36} + \frac{2}{36} = \frac{23}{36}$. The lcm is somewhat analogous to the gcd, but we don't give it equal time because the gcd has nicer properties.

One of the nicest properties of the gcd is that it is easy to compute, using a 2300-year-old method called *Euclid's algorithm*. To calculate $\gcd(m,n)$, for given values $0 \leqslant m < n$, Euclid's algorithm uses the recurrence

$$\begin{aligned} \gcd(0,n) &= n; \\ \gcd(m,n) &= \gcd(n \bmod m,\, m), \qquad \text{for } m > 0. \end{aligned} \tag{4.4}$$

Thus, for example, $\gcd(12,18) = \gcd(6,12) = \gcd(0,6) = 6$. The stated recurrence is valid, because any common divisor of m and n must also be a common divisor of both m and the number $n \bmod m$, which is $n - \lfloor n/m \rfloor m$. There doesn't seem to be any recurrence for $\operatorname{lcm}(m,n)$ that's anywhere near as simple as this. (See exercise 2.)

Euclid's algorithm also gives us more: We can extend it so that it will compute integers m' and n' satisfying

$$m'm + n'n = \gcd(m,n). \tag{4.5}$$

Here's how. If $m = 0$, we simply take $m' = 0$ and $n' = 1$. Otherwise we let $r = n \bmod m$ and apply the method recursively with r and m in place of m and n, computing $\overline{r}$ and $\overline{m}$ such that

$$\overline{r}\,r + \overline{m}\,m = \gcd(r,m).$$

Since $r = n - \lfloor n/m \rfloor m$ and $\gcd(r,m) = \gcd(m,n)$, this equation tells us that

$$\overline{r}\,(n - \lfloor n/m \rfloor m) + \overline{m}\,m = \gcd(m,n).$$

The left side can be rewritten to show its dependency on m and n:

$$(\overline{m} - \lfloor n/m \rfloor \overline{r})\, m + \overline{r}\, n \;=\; \gcd(m,n)\,;$$

hence $m' = \overline{m} - \lfloor n/m \rfloor \overline{r}$ and $n' = \overline{r}$ are the integers we need in (4.5). For example, in our favorite case $m = 12$, $n = 18$, this method gives $6 = 0\cdot 0 + 1\cdot 6 = 1\cdot 6 + 0\cdot 12 = (-1)\cdot 12 + 1\cdot 18$.

But why is (4.5) such a neat result? The main reason is that there's a sense in which the numbers m' and n' actually *prove* that Euclid's algorithm has produced the correct answer in any particular case. Let's suppose that our computer has told us after a lengthy calculation that $\gcd(m,n) = d$ and that $m'm + n'n = d$; but we're skeptical and think that there's really a greater common divisor, which the machine has somehow overlooked. This cannot be, however, because any common divisor of m and n has to divide $m'm + n'n$; so it has to divide d; so it has to be $\leqslant d$. Furthermore we can easily check that d does divide both m and n. (Algorithms that output their own proofs of correctness are called *self-certifying*.)

We'll be using (4.5) a lot in the rest of this chapter. One of its important consequences is the following mini-theorem:

$$k \backslash m \quad \text{and} \quad k \backslash n \qquad \Longleftrightarrow \qquad k \backslash \gcd(m,n)\,. \tag{4.6}$$

(Proof: If k divides both m and n, it divides $m'm + n'n$, so it divides $\gcd(m,n)$. Conversely, if k divides $\gcd(m,n)$, it divides a divisor of m and a divisor of n, so it divides both m and n.) We always knew that any common divisor of m and n must be *less than or equal to* their gcd; that's the definition of greatest common divisor. But now we know that any common divisor is, in fact, *a divisor of* their gcd.

Sometimes we need to do sums over all divisors of n. In this case it's often useful to use the handy rule

$$\sum_{m \backslash n} a_m \;=\; \sum_{m \backslash n} a_{n/m}\,, \qquad \text{integer } n > 0, \tag{4.7}$$

which holds since n/m runs through all divisors of n when m does. For example, when $n = 12$ this says that $a_1 + a_2 + a_3 + a_4 + a_6 + a_{12} = a_{12} + a_6 + a_4 + a_3 + a_2 + a_1$.