Chapter 2

Basic Integer Division

In this chapter, we introduce some concepts of numbers which are familiar, but key for our further study. In particular, we try to understand *why* they work.

- The division algorithm (Section 2.1),
- The greatest common divisor (Section 2.2), and
- The Euclidean algorithm (Section 2.3).

Then we'll put them together with the *Bezout identity* (Section 2.4).

2.1 The Division Algorithm

2.1.1 Statement and examples

Let's start off with the division algorithm. This is the familiar elementary school fact that if you divide an integer a by a positive integer b, you will always get an integer remainder r that is nonnegative, but less than b.

Equally important, there is *only one possible remainder* under these circumstances.

Theorem 2.1.1 Division Algorithm. *For $a, b \in \mathbb{Z}$ and $b > 0$, we can always write $a = qb + r$ with $0 \leq r < b$ and q an integer. Moreover, given a, b there is only one pair q, r which satisfy these constraints. We call the first element q the quotient, and the second one r the remainder.*

Proof. The proof appears below in Subsection 2.1.2. ■

Finding q and r is easy in small examples like $a = 13, b = 3$.

$$\text{We have } 13 = 4 \cdot 3 + 1 \text{ so } q = 4 \text{ and } r = 1.$$

For bigger values it's nice to have the result implemented in Sage.

```
divmod(281376,29)
```

```
(9702, 18)
```

We can check the correctness of the Sage output by multiplying and adding back together.

```
9702*29+18
```

```
281376
```

Sage note 2.1.2 Counting begins at zero. There are several things to note about this early computation. First, note that the answer to `divmod` came in parentheses, a so-called **tuple** data type.

Second, there is another way to approach this computation, more programmatically so that it's easier to reuse. What do you think the `[0]` and `[1]` mean?

```
divmod(281376,29)[0] * 29 + divmod(281376,29)[1]
```

```
281376
```

To access the first and second parts of the answer (the quotient and remainder), we use square brackets, asking for the *0th and 1st* parts of the tuple `(9702,18)`! (This operation is called **indexing**.) In Python, the programming language behind Sage (as in many other languages), counting begins at zero.

The discussion in the previous note actually turns out to be an enduring argument in number theory, too. Do we only care about positive numbers, or nonnegative ones as well? We saw this in the stamps example, since one could send a package for free under certain circumstances (campus mail), but might not care about that case. Similarly, are we required to use at least one of each type of stamp, or is it okay (as in our problem) to not use one type?

2.1.2 Proof of the Division Algorithm

One neat thing about the division algorithm is that it is not hard to prove but still uses the Well-Ordering Principle; indeed, it depends on it. The key set is the set of all possible remainders of a when subtracting multiples of b, which we call

$$S = \{a - kb \mid k \in \mathbb{Z}\}.$$

(Note that the set looks the same if we *add* multiples of b, since $k \in \mathbb{Z}$, but for the purposes of exposition it is easier to think of it as subtraction.)

The object of main interest in the proof will be the nonnegative piece of S which we will call $S' = S \cap \mathbb{N}$. For example, if $a = 13, b = 3$, then $S = \{\ldots 19, 16, 13, 10, 7, 4, 1, -2, -5, \ldots\}$ while $S' = \{\ldots 19, 16, 13, 10, 7, 4, 1\}$.

Our strategy will be to apply the well-ordering principle to S'. (It is worth thinking briefly about why both S and S' are nonempty.) Give the name r to the smallest element of S', which must be writeable as $r = a - bq$ (that's the definition of being an element of $S' \subset S$, after all).

Now let's briefly suppose by way of contradiction that $r \geq b$. In that case we could subtract b from r, and then $r - b \in S'$ as well. So r would not be the least element of S', which is a contradiction. Hence we know that $r < b$. (Note that r is the smallest nonnegative number in S', just as with our intuition regarding remainders from school.)

We still have to show that r and q are the only numbers fulfilling this statement. Suppose $a = bq' + r'$ for some integers q', r' where $0 \leq r' < b$; clearly if $r = r'$ then we can solve $a - bq = r = r' = a - bq'$ to get $q = q'$ (since $b > 0$), so the only interesting case is if $r \neq r'$. Without loss of generality, we can assume $r < r'$.

In that case, $a - bq = r < r' = a - bq'$, which can be rewritten as $0 < r' - r = b(q - q')$. Since $q, q' \in \mathbb{Z}$, by Fact 1.2.2 $q - q'$ must be at least one if

it isn't zero. But then $b = b \cdot 1 \leq r' - r = b(q - q')$ or $b \leq r + b \leq r'$, which contradicts $0 \leq r' < b$. Thus $q - q' = 0$ and hence $q = q'$ and $r = r'$.

It's worth actually trying out the details of this proof with some a and b, say with $a = 26$ and $b = 3$.

As a scholium (see Exercise 2.5.1) note that if $b < 0$ there can still be a positive remainder, but here we would need $0 \leq r < |b|$ in the theorem.

2.1.3 Uses of the division algorithm

It's kind of fun to prove interesting things about powers using the division algorithm, and likely you did in a previous course. For instance, there is an interesting pattern in the remainders of integers when dividing by 4. If you are online, evaluate the following Sage cell to see the pattern. (It's also easy to just get the remainders of the first ten or so perfect squares by hand.)

```
for i in [0..10]:
    pretty_print(html("The␣remainder␣of␣{}␣squared␣with␣
        respect␣to␣4␣is␣{}".format(i,divmod(i^2,4)[1])))
```

Sage note 2.1.3 Repeating commands for different input. The syntax `for i in [0..10]:` just means we want to do the next command for integers from 0 to 10. Such a repetition is called a **loop**.

Another way Python uses to generate the list of different input is the `range` command; try substituting `range(11)` for `[0..10]` in the Sage cell above. Can you discover what the difference is between these?

The rest of the command (all the percent symbols and so forth) is mostly for correct formatting. That includes the indentation in the second line – an essential part of Python and Sage.

This certainly provides strong numerical evidence for the following proposition. But better than that will be the proof!

Proposition 2.1.4 *A perfect square always leaves remainder $r = 0$ or $r = 1$ when divided by 4.*

Proof. Using the division algorithm, we can write $n = 4q + r$. What happens if we square it, $(4q + r)^2$?

Algebraically this yields $16q^2 + 8qr + r^2$. Clearly this is a multiple of 4 plus r^2. So the only possible remainders of n are the remainders of r^2, where r *is already known to be less than 4!*

Now check these yourself to see that the only possibilities are the ones in the statement of the proposition. ■

One cool thing about this proof is that if we just change the proof from using $n = (4q+r)^2$ to one using $n = (mq+r)^2$, we can essentially do the same thing for several divisions at once. If the number we divide by is m, then

$$(mq + r)^2 = m^2q^2 + 2mqr + r^2 = m(mq^2 + 2qr) + r^2,$$

hence all that matters for the final remainder is r^2, since the rest is already divisible by m.

But we know that there are only m possibilities for r, so it's easy to check all *their* squares. For $m = 6$, the following cell checks for you if you don't want to check them by hand.

```
for i in [0..5]:
    pretty_print(html("The␣remainder␣of␣%s␣squared␣with␣
        respect␣to␣6␣is␣%s"%(i,divmod(i^2,6)[1])))
```

This verifies that $r = 0, 1, 3, 4$ are the only possible remainders of perfect squares when you divide by six.

2.2 The Greatest Common Divisor

It seems intuitive that of all the numbers dividing a number (the divisors of the number), one is biggest. We can carry that idea to two numbers.

Definition 2.2.1 Common Divisors. If we consider the various divisors of two numbers a and b, we say that d is a **common divisor** of a and b if $d \mid a$ and $d \mid b$. If d is the biggest such common divisor, it is called the **greatest common divisor**, or **gcd**, of a and b, written $d = \gcd(a, b)$. ◊

Example 2.2.2 What are all the common divisors of 6 and 10? What is their gcd? □

Remark 2.2.3 What is the greatest common divisor of zero and zero? By definition, there is none (or it is infinity?). Some authors (such as [E.2.1]) simply don't allow this case at all; others (like [E.2.4]) define it to be zero without further comment. As for computation, both SageMath[1] and Wolfram Alpha[2] apparently compute it to be zero (perhaps by The Euclidean Algorithm), while one online calculator[3] throws an error.

This text chooses to remain agnostic on this point. However, ring theory and lattice theory both allow for an alternate definition which naturally yields zero as the answer; either consult an abstract algebra text, or see all the answers to this question at Mathematics StackExchange[4] for some good fireside reading after you do your homework for this section.

We now come to a great *definition-theorem.*

Theorem 2.2.4 Characterizing the greatest common divisor. *Let a and b be integers, not both zero. Then the greatest common divisor of a and b is all of the following:*

- *The largest integer d such that $d \mid a$ and $d \mid b$. (This is Definition 2.2.1.)*
- *The number achieved by applying the Euclidean algorithm (a repeated division algorithm) to a and b. (See Section 2.3.)*
- *The smallest positive number which can be written as $ax + by$ for some integers x and y. (See Section 2.4 and Subsection 2.4.2.)*

This is amazing, and the first real indication of the power of having multiple perspectives on a problem. It means that the very theoretical issue of when a gcd exists (*and* finding it) can be treated as a purely computational problem, *completely independent* of finding divisors in the usual sense. And further, there is a definition purely in terms of addition and multiplication, nothing more complex.

If you need to actually calculate a gcd, you use the algorithm. If you want to prove something about it that has to do with dividing, you use the original definition. And if you need to prove something about it where division is hard to use, you use the third characterization. This sort of idea will come up again and again in this book – that having multiple ways to define something *really helps.*

[1] sagecell.sagemath.org/?z=eJxLT07RMNAx0AQACuICDA==
[2] www.wolframalpha.com/input/?i=gcd(0,0)
[3] www.dcode.fr/gcd
[4] math.stackexchange.com/questions/495119/what-is-gcd0-0

2.3 The Euclidean Algorithm

The Euclidean algorithm says that to find the gcd of a and b, one performs the division algorithm until zero is the remainder, each time replacing the previous divisor by the previous remainder, and the previous number to be divided (sometimes called dividend) by the previous divisor. The last non-zero remainder is the gcd.

We'll state and prove this momentarily (Algorithm 2.3.3). Let's try it with a reasonably sized problem.

Example 2.3.1 Let $a = 60$ and $b = 42$.

$$\begin{aligned} 60 &= 42 \cdot 1 + 18 \\ 42 &= 18 \cdot 2 + 6 \\ 18 &= 6 \cdot 3 + 0 \end{aligned}$$

So $\gcd(60, 42) = 6$. □

This procedure is named after Euclid because of Proposition VII.2[5] in Euclid's Elements. There is an amazing complete Java interactive implementation of all the propositions, by David Joyce[6], whose version of this proposition includes some explanation of Euclid's background assumptions. In particular, Euclid basically assumes the Well-Ordering Principle, although of course he didn't think of it in such anachronistic terms.

Historical remark 2.3.2 Euclid's Elements. Euclid, a mathematician in Alexandria during the Hellenistic era, appears to have written the Elements as a compendium of rigorous mathematical knowledge. In addition to being the main geometry textbook in the Western and Islamic worlds for two millennia (as late a teacher as Charles Dodgson a.k.a. Lewis Carroll extolled its virtues in print in *Euclid and His Modern Rivals*[7]), there are substantial number-theoretic portions as well. No one really knows how much of the Elements is original to Euclid, but the work as a whole is monumental and well-organized, despite some well-known criticisms (see e.g. the discussion in [E.5.5]).

Try the algorithm on your own by hand for the gcd of 280 and 126. Or, for even more practice, try it with $\gcd(2013, 1066)$ and then check your work with Sage.

```
gcd(2013,1066)
```

```
1
```

Algorithm 2.3.3 Euclidean algorithm. *To get the greatest common divisor of a and b, perform the division algorithm until you hit a remainder of zero, as below.*

$$\begin{aligned} a &= bq_1 + r_1 \\ b &= r_1 q_2 + r_2 \\ r_1 &= r_2 q_3 + r_3 \\ &\cdots \\ r_{n-3} &= r_{n-2} q_{n-1} + r_{n-1} \\ r_{n-2} &= r_{n-1} q_n + 0 \end{aligned}$$

[5] aleph0.clarku.edu/~djoyce/java/elements/bookVII/propVII2.html
[6] aleph0.clarku.edu/~djoyce/java/elements
[7] books.google.com/books?id=rEUMAAAAYAAJ

Then the previous remainder, r_{n-1}, is the greatest common divisor.

Proof. First let's see why this algorithm even terminates. The division algorithm says each r_i is less than the previous one, yet they may not be less than zero. So let's apply the Well-Ordering Principle to the set of remainders. This set must have a least positive element, and will be the answer. Another way to think about it is that since b is finite, there won't be an infinite number of steps.

Of course, that just gives a number, with no guarantee it has any connection to the gcd. So consider the set of common divisors $d \mid a$ and $d \mid b$. All such d also divide

$$a - q_1 b = 1 \cdot a + (-q_1) \cdot b = r_1$$

So these d also divide $r_2 = b - q_2 r_1$, and indeed divide all the remainders, even $r_{n-1} = r_{n-3} - q_{n-1} r_{n-2}$. So all common divisors of a and b are divisors of r_{n-1}.

On the other hand, if d divides r_{n-1}, it divides $r_{n-2} = r_{n-1} q_n$, and thus divides $r_{n-3} = r_{n-2} q_{n-1} + r_{n-1}$, and so forth. Hence d divides a and b.

So the set of common divisors of a and b are equal to the set of divisors of r_{n-1}, so this algorithm really does give the gcd. ■

As you might expect, the proof makes more sense if you try it out with actual numbers; for the theoretical view, see Exercise 2.5.14. Especially if you can find a and b for which the algorithm takes four or five steps, you will gain some insight.

2.4 The Bezout Identity

2.4.1 Backwards with Euclid

Now, before we get to the third characterization of the gcd, we need to be able to do the Euclidean algorithm *backwards*. This is sometimes known as the *Bezout identity*.

Definition 2.4.1 Bezout identity. A representation of the gcd d of a and b as a linear combination $ax + by = d$ of the original numbers is called an instance of the **Bezout identity**. (This representation is not unique.) ◊

It is worth doing some examples[8]. Perhaps you already have gotten one, probably by trial and error. For instance,

$$6 = -2 \cdot 60 + 3 \cdot 42.$$

The third characterization in Theorem 2.2.4 implies that doing this is *always* possible; $\gcd(a, b) = ax + by$ for some integers x and y. Doing the Euclidean algorithm backwards is one way to obtain this.

Example 2.4.2 Sometimes it helps visually when starting to write the Euclidean algorithm down one side of a table, and then go *up* the other side of the table to obtain an instance of the Bezout identity.

Here's an example with the gcd of 8 and 5; follow it from top left to the bottom and then back up the right side. The middle column provides the necessary rewriting.

[8] For convenience, all examples will be in the form $d = xa + yb$, putting the coefficients first, even though we state this in the other order. The habit of using the letters a, b, d and alphabetical order is too hard to break.

$8 = 1 \cdot 5 + 3$	$1 \cdot 8 - 1 \cdot 5 = 3$	$1 = 2 \cdot 3 - 1 \cdot 5 = 2 \cdot (8 - 1 \cdot 5) - 1 \cdot 5 = 2 \cdot 8 - 3 \cdot 5$
$5 = 1 \cdot 3 + 2$	$1 \cdot 5 - 1 \cdot 3 = 2$	$1 = 1 \cdot 3 - 1 \cdot 2 = 1 \cdot 3 - 1 \cdot (5 - 1 \cdot 3) = 2 \cdot 3 - 1 \cdot 5$
$3 = 1 \cdot 2 + 1$	$1 \cdot 3 - 1 \cdot 2 = 1$	$1 = 1 \cdot 3 - 1 \cdot 2$
$2 = 2 \cdot 1 + 0$		Go up this column...

So $1 = 2 \cdot 8 - 3 \cdot 5$, or $2 \cdot 8 + (-3) \cdot 5$. □

Example 2.4.3 Usually students need a couple of examples of this to get the way this works, so here is another one. Let's do it with the gcd of 60 and 42.

$60 = 1 \cdot 42 + 18$	$1 \cdot 60 - 1 \cdot 42 = 18$	$6 = 1 \cdot 42 - 2 \cdot 18 = 1 \cdot 42 - 2 \cdot (60 - 1 \cdot 42)$
$42 = 2 \cdot 18 + 6$	$1 \cdot 42 - 2 \cdot 18 = 6$	$6 = 1 \cdot 42 - 2 \cdot 18$
$18 = 3 \cdot 6 + 0$		Go up this column...

Simplifying $1 \cdot 42 - 2 \cdot (60 - 1 \cdot 42)$ (the top line on the right), we get $6 = 3 \cdot 42 + (-2) \cdot 60$ again. □

This question of the Bezout identity is implemented in Sage as `xgcd(a,b)`, because this is also known as the **e*X*tended Euclidean algorithm**.

```
xgcd(60,42)
```

```
(6, -2, 3)
```

Or, $6 = -2 \cdot 60 + 3 \cdot 42$, once again.

Example 2.4.4 Try to get the xgcd/Bezout identity for $\gcd(135, 50)$ using this algorithm. You should get $5 = 3 \cdot 135 + (-8) \cdot 50$. Can you get another one a different way?

Try the following Sage cell to check that it works.

```
xgcd(135,50)[1]*135 + xgcd(135,50)[2]*50
```

```
5
```

□

Sage note 2.4.5 Remind how to get list elements. Do you remember what the `[1]` means? What do you think the `[2]` means in this context?

Example 2.4.6 Try to get the xgcd/Bezout identity for $\gcd(1415, 1735)$ using this algorithm. Hopefully you get $5 = 103 \cdot 1415 + (-84) \cdot 1735$, though it may take a while! The previous example might help you on your way. □

Historical remark 2.4.7 Bezout and friends. While Étienne Bézout[9] did indeed prove a version of the Bezout identity for polynomials, the basics of using the extended Euclidean algorithm to solve such equations was known in Europe to Bachet de Méziriac (see Historical remark 3.5.2) about four hundred years ago. However, the Indian mathematician Aryabhata about 1500 years ago in his method later called the Kuṭṭaka[10] used essentially the same algorithm, in fact in a manner more amenable to swift and accurate usage than the one we (and most Western texts) use, with a view toward questions such as Theorem 3.1.2.

2.4.2 Proving the final characterization

The final characterization of the greatest common divisor (Theorem 2.2.4) is that it is the least positive integer which can be written $ax + by$ for integers

[9] mathshistory.st-andrews.ac.uk/Biographies/Bezout
[10] en.wikipedia.org/wiki/Kuttaka

x, y. Let's prove that now.

First, we know there are some positive integers which can be written $ax+by$ (just use positive x, y, or negative ones if a or b are negative). So, by the Well-Ordering Principle, we know there is a smallest such positive integer, which we will call $c = au + bv$. Let's also designate the gcd of a and b to be d.

By Proposition 1.2.8, any integer which divides a and b divides any $ax+by$, so it divides $au + bv = c$. In particular, since d is a divisor of both a and b, it must also divide c. So $d \leq c$.

On the other hand, we know from the backward/extended Euclidean algorithm/Bezout identity that d can be written $d = ax' + by'$ for some integers x' and y'. Since c is the smallest such (positive) integer, $c \leq d$. Thus we conclude that $d = c$.

2.4.3 Other gcd questions

We mentioned earlier there are many such linear combinations for any given pair a, b. How might we find *more than one* such representation?

Example 2.4.8 Using Bezout to get another Bezout. We used the backwards Euclidean algorithm to see that $6 = -2 \cdot 60 + 3 \cdot 42$. Let's use that to get another.

- Since 6 is itself a divisor of both 60 and 42, let's pick one (the smaller one!), 42, and write *it* as $42 = 7 \cdot 6$.
- Then we can really write

$$42 = 7 \cdot 6 = 7 \cdot (-2 \cdot 60 + 3 \cdot 42),$$

  since after all we just saw that was a way to represent 6!
- Now we plug this back into the original equation:

$$6 = -2 \cdot 60 + 3 \cdot 42 = -2 \cdot 60 + 3 \cdot (7 \cdot 6)$$

$$= -2 \cdot 60 + 3 \cdot (7 \cdot (-2 \cdot 60 + 3 \cdot 42))$$

If we simplify it out, that means $6 = -44 \cdot 60 + 63 \cdot 42$, which is indeed correct! □

So, substituting a Bezout identity into itself yields more and more such identities. How many such identities are there? Is there a general form?

Another interesting question is that some gcds of large numbers are very easy to compute. What makes finding $\gcd(42000, 60000)$ so easy? If you're in a classroom, this is a perfect time to discuss.

On a related note, if $\gcd(a, b) = d$, could you make a guess as to a formula for $\gcd(ka, kb)$ (for $k > 0$)? Can you *prove* it in Exercise 2.5.16? (Hint: here is where our original definition *or* the Bezout version could be useful.)

2.4.4 Relatively prime

There is one final thing that the linear combination version of the gcd can give us. It is something you may think is familiar, but which can arise very naturally from the Bezout identity.

Consider the *smallest* possible greatest common divisor, which is one. Under what circumstances would a and b have $\gcd(a, b) = 1$? By our characterization, it is precisely when you can write $ax + by = 1$ for some integers x and y.

Think about this, though; if the gcd of a and b is 1, then we could write *any* integer as a (linear) combination of a and b! This is a property I think people would have come up with no matter how the development of mathematics had gone; namely, identifying pairs of integers such that you can write any number as a (linear) combination of them.

Definition 2.4.9 Relatively Prime. If the greatest common divisor of two numbers is one, we call them **relatively prime** numbers or **coprime** numbers.

Later, we will need to have a term for the situation where, in a collection of several integers, all possible pairs are relatively prime. We will call this **mutually coprime**, **coprime in pairs**, or an analogous term. ♢

Proposition 2.4.10 *Here are two interesting facts about coprime integers a and b:*

- *If $a \mid c$ and $b \mid c$, then $ab \mid c$.*
- *If $a \mid bc$, then $a \mid c$.*

Proof. The first is not too hard to prove, *if* you think in terms of Bezout. It does need a little cleverness.

- Remember that $1 = ax + by$ for some x, y, by definition of being coprime.
- So $c = cax + cby$.
- Now write $c = kb$ and $c = \ell a$, and substitute them in the *opposite* parts of the previous line.
- This gives $c = (kb)ax + (\ell a)by$, and ab definitely divides both parts of this, so it divides the whole thing by our earlier proposition about divisibility.

We leave the second as an exercise (Exercise 2.5.19). ■

It's also useful to try to find *counterexamples*! Can you find an example where $\gcd(a, b) \neq 1$, $a \mid c$ and $b \mid c$, but ab does not divide c? (See Exercise 2.5.20.)

2.5 Exercises

1. Try stating and proving the division algorithm (Theorem 2.1.1) but for $b < 0$.

2. Can you find an n such that the possible remainders of a perfect square when divided by n are all numbers between zero and $n - 1$? If you can, how many different such n can you find? If not, can you prove there are none?

3. Write the gcd of 3 and 4 as a linear combination of 3 and 4 in three different ways. (Hint: trial and error.)

4. You can define the gcd of more than two numbers as the greatest integer dividing all of the numbers in your set. So, for instance, $\gcd(20, 30, 70) = 10$. Calculate the gcd of some hard-looking sets of three numbers by listing divisors.