Math 173A

Liam Hardiman

June 24, 2022

Abstract

I'm writing these lecture notes for UC Irvine's Math 173A course, taught in the summer of 2022. This is a five-ish week course where I plan to get through the first three chapters of Hoffstein, Pipher and Silverman's book [1]. The class structure consists of a two hour lecture followed by a one hour discussion section three days a week. I'm aiming to get through two sections of the book per lecture with a midterm after chapter 2.

Contents

1 An Introduction to Cryptography

1.1 Simple Substitution Ciphers

One of history's oldest examples of encrypting messages is the *shift cipher*, sometimes called the *Caesar cipher* after Julius Caesar, who allegedly used it to encrypt the orders he'd send to his troops. To encrypt a message, simply shift each letter of the plaintext forward in the alphabet by three, wrapping around if the shifted letter goes past `Z`. For example, if the key[1] is `3` and our plaintext is `hello, world`, then we have the following ciphertext.

$$\texttt{hello world} \mapsto \texttt{KHOOR ZRUOG}$$

Conversely, if we know the key is `3` and we're given the ciphertext `ZHGQH VGDB`, then we simply shift backwards by 3 to obtain the plaintext.

$$\texttt{ZHGQH VGDB} \mapsto \texttt{wedne sday}$$

[1]We won't rigorously define what "plaintext", "ciphertext" or "key" mean. You can think of the plaintext as being being the human-readable or usable message (maybe consisting of letters or a number) and the ciphertext as being some unreadable sequence of letters or numbers. Then you can think of the key as being some piece of information that tells you how to convert between plain- and ciphertext.

One advantage to the shift cipher is that it's really easy to encrypt and decrypt messages if the key is known. The main disadvantage is that it's only slightly challenging (more annoying than challenging) for an adversary to decrypt messages even if they don't know the key. If we use the English alphabet, then there are only 26 possible keys and it doesn't take too long to try them all (a few minutes by hand, a fraction of a second even with bad code). This trial and error method of trying all possible keys, sometimes called *brute forcing*, works because it's pretty unlikely that decrypting with two different keys will yield two plaintexts that are both readable. For example, suppose we happen upon the following ciphertext

`XPPEE ZXZCC ZH.`

If we suspect that this ciphertext came from a shift cipher, we can just try all possible un-shifts to get the following possible plaintexts.

key	plaintext	key	plaintext
1	`woodd ywybb yg`	14	`jbbqq ljloo lt`
2	`vnncc xvxaa xf`	15	`iaapp kiknn ks`
3	`ummbb wuwzz we`	16	`hzzoo jhjmm jr`
4	`tllaa vtvyy vd`	17	`gyynn igill iq`
5	`skkzz usuxx uc`	18	`fxxmm hfhkk hp`
6	`rjjyy trtww tb`	19	`ewwll gegjj go`
7	`qiixx sqsvv sa`	20	`dvvkk fdfii fn`
8	`phhww rpruu rz`	21	`cuujj ecehh em`
9	`oggvv qoqtt qy`	22	`bttii dbdgg dl`
10	`nffuu pnpss px`	23	`asshh cacff ck`
11	`meett omorr ow`	24	`zrrgg bzbee bj`
12	`lddss nlnqq nv`	25	`yqqff ayadd ai`
13	`kccrr mkmpp mu`		

The only plaintext here that's even remotely readable is `meett omorr ow`, corresponding to a key of `11`. This process of decrypting a ciphertext without knowing the key in advance is called *cryptanalysis*.

Notice that with a shift cipher, each instance of `a` encrypts to the same character, and so on. In this setting, once we know what one character maps to, then we know what all the other characters map to as well. E.g. if we know that `m` maps to `X`, then we know that the cipher shifts each character forward by 11, which immediately tells us that `a` maps to `L`, and so on. A more general *simple substitution cipher* decouples the encryptions of different letters, e.g. each `a` maps to `C` and each `b` maps to `J`, etc.

Question 1.1. *Explain why this particular substitution cipher is not a shift cipher.*

Question 1.2. *How many possible keys are there in a substitution cipher? Hint: think of encryption as a function. What properties should this function have?*

What would cryptanalysis of a simple substitution cipher look like? There are more than 10^{26} keys in this case. If we could try a million keys every second, it would still take more than 10^{13} years to try them all, so the brute-force solution is infeasible. Despite the huge number of possible keys, simple substitution ciphers are often really easy to cryptanalyze in practice with simple *frequency analysis*. The idea is that if the plaintext is more than a few sentences long, then one might expect to see a lot of `e`'s, `t`'s and `a`'s and not many `z`'s or `q`'s. Consequently, if we look at the frequencies of the letters in the ciphertext, it would be reasonable to guess that the most common ciphertext letters correspond to the most common plaintext letters.

For example, suppose we intercept the following message.

```
LWNSOZ BNWVWB AYBNVB SQWVUO HWDIZW RBBNPB POOUWR PAWXAW
PBWZWM YPOBNP BBNWJP AWWRZS LWZQJB NVIAXA WPBSAL IBNXWA
BPIRYR POIWRP QOWAIE NBVBNP BPUSRE BNWVWP AWOIHW OIQWAB
JPRZBN WFYAVY IBSHNP FFIRWV VBNPBB SVWXYA WBNWVW AIENBV
ESDWAR UWRBVP AWIRVB IBYBWZ PUSREU WRZWAI DIREBH WIATYV
BFSLWA VHASUB NWXSRV WRBSHB NWESDW ARWZBN PBLNWR WDWAPR
JHSAUS HESDWA RUWRBQ WXSUWV ZWVBAY XBIDWS HBNWVW WRZVIB
IVBNVA IENBSH BNWFWS FOWBSP OBWASA BSPQSO IVNIBP RZBSIR
VBIBYB WRWLES DWARUW RBOPJI REIBVH SYRZPB ISRSRV YXNFAI
RXIFOW VPRZSA EPRIKI REIBVF SLWAVI RVYXNH SAUPVB SVWWUU
SVBOIC WOJBSW HHWXBB NWIAVP HWBJPR ZNPFFI RWVV
```

Let's arrange the letters in the ciphertext by frequency.

W	B	R	S	I	V	A	P	N	O	···
76	64	39	36	36	35	34	32	30	16	···

The letters in standard English text have the following frequencies.

E	T	A	O	N	R	I	S	H	D	···
.131	.105	.082	.080	.071	.068	.064	.061	.053	.038	···

Since the letter `W` appears much more frequently than the other letters in the ciphertext, it tips us off that we might be dealing with a substitution cipher and that an `e` in the plaintext probably maps to a `W` in the ciphertext. It's also reasonable to guess that the letters `B`, `R`, `S` and `I` correspond to the letters `t`, `a`, `o` and `i` in some order.

Looking at individual letter frequencies lets us get our foot in the door, but it doesn't help us much when it comes to differentiating between letters that appear with roughly the same frequency (like `R` and `S` in this ciphertext). If we think about English text for a bit, we notice that certain pairs of letters, called *bigrams*, appear together more frequently than others (e.g. `q` is almost always followed by a `u` and `th` is a common pair). Here are a few of the bigram frequencies from our ciphertext

	W	B	R	S	I	V	A	P	N
W	3	4	12	2	4	10	14	3	1
B	4	4	0	11	5	5	2	4	20
R	5	5	0	1	1	5	0	3	0
S	1	0	5	0	1	3	5	2	0
I	1	8	10	1	0	2	3	0	0
V	8	10	0	0	2	2	0	3	1
A	7	3	4	2	5	4	0	1	0
P	0	8	6	0	1	1	4	0	0
N	14	3	0	1	1	1	0	7	0

That is, this table tells us that WN appears once and NW appears 14 times. In English, the letter h frequently comes before e and rarely comes after it, so it's a safe guess that h maps to N in this particular substitution. Since th is the most common digram in English and BN is the most common digram in the ciphertext, we guess that t maps to B. Other features of the English language lead to more educated guesses that lead to a full cryptanalysis of the ciphertext.

Problem 1.3. Finish decrypting the ciphertext. One place to start is by looking for vowels and noting that some vowels like a, i and o tend to avoid each other.

1.2 Divisibility and Greatest Common Divisors

Some of the most widely-used cryptosystems today make heavy use of abstract algebra and number theory. Roughly speaking, number theory is concerned with properties of the integers, $\mathbb{Z}$, like divisibility and solutions to equations with integer variables.

Definition 1.4. Let a and b be integers with $b \neq 0$. We say that b *divides* a if $a = bc$ for some integer c, in which case, we write $b \mid a$.

Example 1.5. (a) We call the integers divisible by 2 *even* and those that aren't *odd*. Is zero even or odd?

(b) 713 is divisible by 23 since $713 = 23 \cdot 31$. The numbers used in everyday cryptographic applications are hundreds or even thousands of digits long.

(c) A number n is divisible by 5 if and only if it ends in a 0 or a 5 (when written in base 10, of course). To see this, write

$$n = d_0 + 10d_1 + 10^2 d_2 + \cdots + 10^k d_k,$$

where $k \geq 0$ and $d_i \in \{0, 1, 2, \ldots, 9\}$ for all i. Then d_0 is the number that n "ends" with, so if it's 0 or 5, we can just factor a 5 out of the right-hand side to see that n is divisible by 5. Conversely, if we rearrange this,

$$d_0 = n - 10d_1 - 10^2 d_2 - \cdots - 10^k d_k,$$

we see that if n is divisible by 5, then the whole right-hand side (which is equal to d_0) is also divisible by 5.

We record some basic properties of divisibility here. The proof of this proposition is a straightforward exercise.

Proposition 1.6. *Let a, b and c be integers.*

(a) If $a \mid b$ and $b \mid c$, then $a \mid c$.

(b) If $a \mid b$ and $b \mid c$, then $a = \pm b$.

(c) If $a \mid b$ and $a \mid c$, then $a \mid (b + c)$ and $a \mid (b - c)$.

Question 1.7. *For those familiar with equivalence relations, is divisibility an equivalence relation on $\mathbb{Z}$?*

Definition 1.8. A *common divisor* of integers a and b is a positive integer d that divides both of them. The *greatest common divisor* of a and b is the largest positive integer d such that $d \mid a$ and $d \mid b$ and we write $d = \gcd(a, b)$ or $d = (a, b)$ if there is no possibility of confusion.

Example 1.9. (a) Find the greatest common divisor of 132 and 66 by listing out all of their divisors.

(b) Find the greatest common divisor of 80 and 5. Other than the number being pretty small, why was this easy to do? Prove your idea.

Of course given integers a and b, it's not always the case that $a \mid b$ or $b \mid a$. In this case, we get a (unique) remainder.

Proposition 1.10. *For any positive integers a and b, there exist unique integers q and r such that*

$$a = bq + r \qquad \text{with } 0 \leq r < b. \tag{1}$$

Here we call q the quotient *and r the* remainder *when a is divided by b.*

Proof. Homework exercise. □

Division with remainder provides us with a way of finding the gcd of two integers. To see this, rearrange (1) to obtain

$$r = a - bq.$$

If d is a common divisor of a and b, then it clearly divides the right-hand side of this equation, so it must divide r as well. A similar rearrangement (which?) shows that if c is a common divisor of b and r, then it must also divide a. We then have that the common divisors of a and b are the common divisors of b and r, so we must have that

$$\gcd(a, b) = \gcd(b, r).$$

This is great because if we assume that $a > b$, then we've reduced the problem of finding $\gcd(a, b)$ to finding the gcd of two smaller numbers, b and r. We can then repeat this: divide b by r to obtain

$$b = q'r + r', \qquad \text{with } 0 \leq r' < r.$$

By the same reasoning, we have that

$$\gcd(a, b) = \gcd(b, r) = \gcd(r, r').$$

Since the remainders are positive numbers that get strictly smaller after each division, we must eventually reach a remainder of zero. The remainder right before this one is then the gcd of a and b.

Example 1.11. Let's compute $\gcd(12345, 11111)$. Even without a calculator it's sometimes easy to eyeball how many times one number goes into another.

$$\begin{aligned} 12345 &= 11111 \cdot 1 + 1234 \\ 11111 &= 1234 \cdot 9 + 5 \\ 1234 &= 5 \cdot 246 + 4 \\ 5 &= 4 \cdot 1 + 1 \\ 4 &= 1 \cdot 4 + 0 \end{aligned}$$

The second-to-last remainder we found was 1, so we conclude that $\gcd(12345, 11111) = 1$. Note that even though the numbers involved started out somewhat large (for by-hand computations), we were able to calculate the gcd in just a few steps.

This procedure for computing the gcd of two integers is called the *Euclidean algorithm* after the ancient Greek mathematician. We summarize it here.

Theorem 1.12. *Let $a \geq b$ be positive integers. Then the following algorithm computes* $\gcd(a, b)$ *in a finite number of steps (i.e., the algorithm eventually terminates).*

1: *Let $r_0 = a$ and $r_1 = b$.*
2: *Set $i = 1$.*
3: *Divide r_{i-1} by r_i with remainder to obtain quotient q_i and remainder r_{i+1}.*

$$r_{i-1} = r_i \cdot q_i + r_{i+1}, \qquad \textit{with } 0 \leq r_{i+1} < r_i.$$

4: *If $r_{i+1} = 0$, then $r_i = \gcd(a, b)$ and the algorithm terminates.*
5: *Otherwise, $r_{i+1} > 0$. Set $i = i + 1$ and go to Step 3.*

How many times do we need to repeat the division step of the algorithm? Let's start by looking at how much the remainders drop at each step. At each step we have two possibilities: either $r_{i+1} \leq \frac{1}{2} r_i$ or $r_{i+1} > \frac{1}{2} r_i$. In the first case, since the remainders are strictly decreasing, we have

$$r_{i+2} < r_{i+1} \leq \frac{1}{2} r_i.$$

In the other case we must have $r_i = r_{i+1} \cdot 1 + r_{i+2}$. Rearranging, we have

$$r_{i+2} = r_i - r_{i+1} < r_i - \frac{1}{2} r_i = \frac{1}{2} r_i.$$

In either case, we have that the remainder drops by at least half every two steps. After $2k+1$ steps we then have

$$r_{2k+1} < \frac{1}{2} r_{2k-1} < \frac{1}{2^2} r_{2k-3} < \cdots < \frac{1}{2^k} r_1 = \frac{1}{2^k} b.$$

If k is the smallest integer such that $b/2^k < 1$, then we have $r_{2k+1} = 0$. Setting $k = \lfloor \log_2 b \rfloor + 1$ does the trick. The *gcd* is then found on step at most $2k = 2\lfloor \log_2 b \rfloor + 2$.

Remark 1.13. Pretty much all cryptography software includes some implementation of the Euclidean algorithm. Computers store integers in their binary representations where an integer N takes $n = \lfloor \log_2 N \rfloor + 1$ bits of memory (why?). The above analysis shows that the Euclidean algorithm runs in a number of steps equal to at most twice the number of bits ($2n$) it takes to store the smaller of its two inputs. When the number of steps it takes an algorithm to complete grows (at most) like a polynomial in its input size, then we consider it to be (reasonably) efficient.

The Euclidean algorithm also gives us a way of writing $\gcd(a, b)$ as a linear combination of a and b.

Example 1.14. Let's return to Example 1.11.

Write $a = 12345$ and $b = 11111$ and solve for the first remainder, 1234, in terms of a and b:

$$1234 = a - b.$$

Now plug this into the second equation to get

$$b = (a - b) \cdot 9 + 5,$$

So the next remainder, 5, can be written in terms of a and b as

$$5 = -9a + 10b.$$

Plug this along with the expression for 1234 into the third equation to get

$$a - b = (-9a + 10b) \cdot 246 + 4,$$

which gives the next remainder, 4, in terms of a and b:

$$4 = 2215a - 2461b.$$

Finally, plug the expressions for 4 and 5 into the second-to-last equation to get

$$1 = (-9a + 10b) - (2215a - 2461b) = -2224a + 2471b.$$

This example is more or less a proof of the following theorem.

Theorem 1.15. *Let a and b be positive integers. Then the equation*

$$ax + by = c$$

has integer solutions for x and y if and only if c is divisible by $\gcd(a, b)$. *Moreover, if (x_0, y_0) is a particular solution to this equation, then every other solution has the form*

$$x = x_0 + \frac{kb}{\gcd(a,b)}, \quad y = y_0 - \frac{ka}{\gcd(a,b)}$$

for some integer k.

1.3 Modular Arithmetic

Recall that when encrypting a message with a shift cipher with key k, each letter in the plaintext is shifted forward in the alphabet by k positions. Importantly, we *wrap around* the alphabet if we shift past the letter `Z` (or whatever letter is at the end of the relevant alphabet). This idea of wrapping around the end back to the beginning comes up in our day-to-day lives when we think about telling time. Four hours after 9AM is 1PM since we *wrap around* 12pm back to 1PM (the same idea holds if you prefer to think in military time - three hours after 2300 is 0200). We'll look at this mathematically with the idea of *congruence*.

Definition 1.16. Let $m \geq 1$ be an integer. We say that the integers a and b are *congruent modulo* m if the difference $a - b$ is divisible by m. In this case we write

$$a \equiv b \pmod{m}$$

and call m the *modulus*.

Example 1.17. We have that $2 \equiv 5 \pmod 7$. We also have that $2 \equiv 9 \pmod 7$ and $2 \equiv 16 \pmod 7$.

Importantly, congruences respect familiar operations like addition and multiplication, but are a little trickier when it comes to division.

Proposition 1.18. *Let $m \geq 1$ be an integer.*

1. *If $a_1 \equiv a_2 \pmod{m}$ and $b_1 \equiv b_2 \pmod{m}$, then*

$$a_1 \pm b_1 \equiv a_2 \pm b_2 \pmod{m} \qquad \text{and} \qquad a_1 b_1 \equiv a_2 b_2 \pmod{m}.$$

2. *Let a be an integer. Then there exists an integer b such that*

$$ab \equiv 1 \pmod{m}$$

if and only if $\gcd(a, m) = 1$. *In this case, we call b the* multiplicative inverse of a modulo m *and we write $b = a^{-1} \pmod{m}$.*

Proof. The proof of part (a) isn't super interesting, so we'll skip it.

For part (b), first suppose that $\gcd(a, m) = 1$. Then by Theorem (1.15), we can find u and v such that $au + mv = 1$. But if we rearrange this, we have

$$au - 1 = mv,$$

so the difference $au - 1$ is divisible by m and $au \equiv 1 \pmod{m}$. In this case, u is an inverse of a $\pmod{m}$.

On the other hand, suppose a has a multiplicative inverse $b \pmod{m}$. Then m divides the difference $ab - 1$ so we have

$$ab - km = 1$$

for some integer k. If d is some (positive) common divisor of a and m, then d must divide the left-hand side of this equation. But then d must divide 1, so we must have $d = 1$. It must then be the case that $\gcd(a, m) = 1$. □

Part (b) of this proposition gives us a partial analogue of division modulo m. Just like how the rational number $1/2$ has the property that $(1/2) \cdot 2 = 1$, the number 3 has the property that $3 \cdot 2 = 6 \equiv 1 \pmod{5}$, so 3 plays a similar role to $1/2$. What's more is that our proof of part (b) gives us an algorithm for computing the modular inverse: the extended Euclidean algorithm.

Example 1.19. Let's find the inverse of 4 modulo 7 (if it exists at all). First compute $\gcd(4, 7)$.

$$\begin{aligned} 7 &= 4 \cdot 1 + 3 \\ 4 &= 3 \cdot 1 + 1 \\ 3 &= 1 \cdot 3 + 0 \end{aligned}$$

So $\gcd(4, 7) = 1$, so we know a modular inverse exists. We find it by substituting in expressions for the remainders.

$$\begin{aligned} 1 &= 4 - 3 \cdot 1 \\ &= 4 - (7 - 4) \\ &= 4 \cdot 2 - 7. \end{aligned}$$

Rearranging this, we see that $4 \cdot 2 - 1 = 7$, so $4 \cdot 2 \equiv 1 \pmod{7}$, and 2 is the inverse of 4 modulo 7.

Remember that the Euclidean algorithm is really efficient (for a computer at least - so is the extended one), so finding inverses is efficient as well.

Returning to the above example, note that $4 \cdot 9 = 36 \equiv 1 \pmod 7$ as well, so we can just as easily say that 9 is an inverse of 4 modulo 7. It would be nice if there was just one inverse or a way for two people to pick the same inverse every time. Division with remainder gives us a way of doing this. If b is an inverse of a modulo m, write

$$b = mq + r \quad \text{with } 0 \leq r < m.$$

Then r is always between 0 and $m-1$. Since this r is unique, we can agree that we always work with the integers 0 through $m-1$ when we work modulo m. This idea is encapsulated in the following proposition.

Proposition 1.20. *The integers a and b are congruent modulo m if and only if they have the same remainder when divided by m.*

Recall the notion of equivalence relations.

Definition 1.21. A relation $\sim$ on a set X is an *equivalence relation* if the following all hold.

1. (Reflexivity) $x \sim x$ for all $x \in X$.
2. (Symmetry) $x \sim y$ if and only if $y \sim x$ for any $x, y \in X$.
3. (Transitivity) if $x \sim y$ and $y \sim z$ then $x \sim z$.

For each $x \in X$, the *equivalence class of* x, denoted $[x]$ (or sometimes $\overline{x}$) is

$$[x] = \{y \in X : x \sim y\}.$$

We can form the new set $X/\sim$, the *quotient of X by $\sim$* by just taking equivalence classes.

$$X/\sim = \{[x] : x \in X\}.$$

Modular arithmetic is a concrete example of this.

Proposition 1.22. *Fix a positive integer $m \geq 2$. Then equivalence modulo m is an equivalence relation on $\mathbb{Z}$.*

Moreover, Proposition (1.20) leads us to think that the quotient of $\mathbb{Z}$ by "equivalence modulo m" is the "correct" object to work with and to choose our equivalence classes to be $[0], \ldots, [m-1]$.

Definition 1.23. The set $\mathbb{Z}/m\mathbb{Z}$ is defined to be the set of integers quotiented by the relation "equivalent modulo m". Specifically,

$$\mathbb{Z}/m\mathbb{Z} = \{[0], \ldots, [m-1]\},$$

where $[a] = \{b \in \mathbb{Z} : a \equiv b \pmod m\}$.

Remark 1.24. When working with $\mathbb{Z}/m\mathbb{Z}$, we usually drop the $[\cdot]$ when talking about its elements, which are equivalence classes. That is, it technically doesn't make sense to write $2 \in \mathbb{Z}/5\mathbb{Z}$ since 2 isn't an equivalence class. However, as the next proposition shows, the equivalence class [2] can be made to behave a lot like the ordinary integer 2.

We can carry the notions of addition and multiplication over to the quotient as well.

Definition 1.25. For $[a], [b] \in \mathbb{Z}/m\mathbb{Z}$, define $[a] + [b]$ to be $[a + b]$ and $[a] \cdot [b]$ to be $[ab]$.

Remark 1.26. Technically, the above definition should be made into a proposition that says this definition is *well-defined*. That is, we need to show that if $a \equiv a'$ and $b \equiv b'$ then we want $[a] + [b] = [a'] + [b']$ and $[a] \cdot [b] = [a'] \cdot [b']$. This follows easily from Proposition (1.18).

Let's think about Theorem 1.15 for a bit in this context by looking at equations in $\mathbb{Z}/m\mathbb{Z}$.

Example 1.27.

1. Does $2x \equiv 3$ have a solution modulo 5? It would be nice if we could "divide by 2" and that's exactly what a multiplicative inverse lets us do. It's easy to verify that 4 is the inverse of 2 (mod 5), so multiplying both sides of this equation through by 4 gives $x \equiv 12 \equiv 2 \pmod 5$.

2. What about $2x \equiv 3 \pmod 6$? This equation in $\mathbb{Z}/6\mathbb{Z}$ is equivalent to the integer equation $2x - 3 = 6y$, which has no solution since the left-hand side is always odd while the right-hand side is always even. Another way we can think about it is that we can't "divide by 2" since $\gcd(2, 6) = 2 \neq 1$.

3. What about $2x \equiv 4 \pmod 6$? Just like in the last example, we can't divide by 2. However, it's easy to see that $x \equiv 2 \pmod 6$ is a solution. But this solution isn't unique since $x \equiv 5 \pmod 6$ is also a solution.

In short, the existence of an inverse, as determined by Theorem 1.15 determines whether or not equations like $ax \equiv b \pmod m$ have solutions. If $\gcd(a, m) = 1$, then there's a unique solution. Otherwise, there can either be no solution or there might be multiple solutions. If we want to restrict ourselves to the (equivalence classes of) integers that *do* have inverses modulo m, then we use the following object.

Definition 1.28. Fix an integer $m \geq 2$. Then the set of *units modulo m* is denoted by

$$\begin{aligned}(\mathbb{Z}/m\mathbb{Z})^\times &= \{a \in \mathbb{Z}/m\mathbb{Z} : \gcd(a, m) = 1\} \\ &= \{a \in \mathbb{Z}/m\mathbb{Z} : a \text{ has an inverse modulo } m\}.\end{aligned}$$

1.4 Prime Numbers, Unique Factorization, and Finite Fields

The "building blocks" of the integers are the prime numbers.

Definition 1.29. An integer p is called *prime* if $p \geq 2$ and if the only positive integers dividing p are 1 and p.

Note that if p is prime, then $\gcd(a, p) = 1$ for each $1 \leq a < p$ (why?). Consequently, each nonzero element of $\mathbb{Z}/p\mathbb{Z}$ has a multiplicative inverse, i.e.

$$(\mathbb{Z}/p\mathbb{Z})^\times = \{1, 2, \ldots, p - 1\}.$$

The set $\mathbb{Z}/p\mathbb{Z}$ forms a structure that we call a (finite) *field*: a set where we can add and subtract as well as multiply and divide by (nonzero) elements. Other examples of fields include $\mathbb{R}$ and $\mathbb{Q}$ but not $\mathbb{Z}$.

Proposition 1.30. *Let p be a prime number and suppose that $p \mid ab$. Then $p \mid a$ or $p \mid b$. More generally, if*

$$p \mid a_1 a_2 \cdots a_k,$$

then $p \mid a_i$ for some i.

Proof. We'll prove the first statement and you'll prove the second one in discussion. If p divides a then we're done. If $p \nmid a$, then $\gcd(a, p) = 1$ (why?), so we can write

$$au + pv = 1$$

for some integers u and v. Multiplying this through by b gives

$$abu + pbv = b.$$

By assumption, $p \mid ab$ and clearly $p \mid pbv$, so p divides the left-hand side of this equation. Consequently, p divides the right-hand side, which is b. □

Using this, we can prove what we said earlier about primes being "building blocks."

Theorem 1.31 (The Fundamental Theorem of Arithmetic)**.** *Let $a \geq 2$ be an integer. Then a can be factored as a product of prime numbers*

$$a = p_1^{e_1} p_2^{e_2} \cdots p_r^{e_r} \tag{2}$$

for some positive integer r. Furthermore, this factorization is unique up to rearrangement of the primes.

Proof. We prove that we can factor into primes by induction and uniqueness will come later. Our base case is $a = 2$, and this itself is a prime factorization since 2 is prime. Suppose then that every integer less than a can be factored into primes. If a itself is prime, then we're done by the same reasoning we used in the base case. Otherwise, $a = bc$ where $1 < b, c < a$. By the induction hypothesis, we can factor b and c into primes:

$$b = p_1^{e_1} \cdots p_k^{e_k}, \quad c = q_1^{f_1} \cdots q_\ell^{f_\ell}.$$

But then $a = p_1^{e_1} \cdots p_k^{e_k} q_1^{f_1} \cdots q_\ell^{f_\ell}$ is a factorization of a. You'll prove the uniqueness part of this statement in discussion. □

Looking at the factorization of a into primes (2), we call the number of times a particular prime, p, appears in the factorization the *order of p in a* and denote it by $ord_p(a)$. That is, in the factorization (2), $ord_{p_i}(a) = e_i$.

References

[1] Hoffstein, Jeffrey, Jill Pipher and Joseph H. Silverman. *An Introduction to Mathematical Cryptography*. Second Edition. Springer New York, NY. 2014.