

4 Modular Arithmetic

We begin the chapter on number theory by introducing modular integer arithmetic. One of its uses is in the encryption of secret messages. In this section, all numbers are integers.

Private key cryptography. The problem of sending secret messages is perhaps as old as humanity or older. We have a sender who attempts to encrypt a message in such a way that the intended receiver is able to decipher it but any possible adversary is not. Following the traditional protocol, the sender and receiver agree on a secret code ahead of time, and they use it to both encrypt and decipher the message. The weakness of the method is the secret code, which may be stolen or cracked.

As an example, consider Caesar’s cipher, which consists of shifting the alphabet by some fixed number of positions, e.g.,

\[
\begin{array}{cccccccccccc}
A & B & C & \ldots & V & W & X & Y & Z \\
\downarrow & \downarrow & \downarrow & \ldots & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \downarrow
\end{array}
\]

E F G \ldots Z ABCD.

If we encode the letters as integers, this is the same as adding a fixed integer but then subtracting 26, the number of letters, if the sum exceeds this number. We consider this kind of integer arithmetic more generally.

Public key cryptography. Today, we use more powerful encryption methods that give a more flexible way to transmit secret information. We call this public key cryptography which roughly works as follows. As before, we have a sender, called Alice, and a receiver, called Bob. Both Alice and Bob have a public key, \( KP_A \) and \( KP_B \), which they publish for everyone to see, and a secret key, \( KS_A \) and \( KS_B \), which is only known to themselves. They do not exchange the secret key even among each other. The keys are used to change messages so we can think of them as functions. The function that corresponds to the public and the secret keys are inverses of each other, that is,

\[
S_A(P_A(x)) = P_A(S_A(x)) = x;
\]

\[
S_B(P_B(x)) = P_B(S_B(x)) = x.
\]

The crucial point is that \( P_A \) is easy to compute for everybody and \( S_A \) is easy to compute for Alice but difficult for everybody else, including Bob. Symmetrically, \( P_B \) is easy for everybody but \( S_B \) is easy only for Bob. Perhaps this sound contradictory since everybody knows \( P_A \) and \( S_A \) is just its inverse, but it turns out that there are pairs of functions that satisfy this requirement. Now, if Alice wants to send a message to Bob, she proceeds as follows:

1. Alice gets Bob’s public key, \( P_B \).
2. Alice applies it to encrypt her message, \( y = P_B(x) \).
3. Alice sends \( y \) to Bob, publically.
4. Bob applies \( S_B(y) = S_B(P_B(x)) = x \).

We note that Alice does not need to know Bob’s secret key to encrypt her message and she does not need secret channels to transmit her encrypted message.

Arithmetic modulo \( n \). We begin by defining what it means to take one integer, \( m \), modulo another integer, \( n \).

**Definition.** Letting \( n \leq 1 \), \( m \mod n \) is the smallest integer \( r \geq 0 \) such that \( m = nq + r \) for some integer \( q \).

Given \( m \) and \( n \geq 1 \), it is not difficult to see that \( q \) and \( r \) exist. Indeed, \( n \) partitions the integers into intervals of length \( n \):

\[\ldots, -n, \ldots, 0, \ldots, n, \ldots, 2n, \ldots\]

The number \( m \) lies in exactly one of these intervals. More precisely, there is an integer \( q \) such that \(qn \leq m < ((q + 1)n)\). The integer \( r \) is the amount by which \( m \) exceeds \( qn \), that is, \( r = m - qn \). We see that \( q \) and \( r \) are unique, which is known as

**Euclid’s Division Theorem.** Letting \( n \geq 1 \), for every \( m \) there are unique integers \( q \) and \( 0 \leq r < n \) such that \( m = nq + r \).

**Computations.** It is useful to know that modulos can be taken anywhere in the calculation if it involves only addition and multiplication. We state this more formally.

**Lemma 1.** Letting \( n \geq 1 \), \( i \mod n = i + kn \mod n \).

This should be obvious because adding \( k \) times \( n \) moves the integer \( i \) to the right by \( k \) intervals but maintains its relative position within the interval.

**Lemma 2.** Letting \( n \geq 1 \), we have

\[
(i + j) \mod n = (i \mod n) + (j \mod n) \mod n;
\]

\[
(i \cdot j) \mod n = (i \mod n) \cdot (j \mod n) \mod n.
\]
PROOF. By Euclid’s Division Theorem, there are unique integers \( q_i, q_j \) and \( 0 \leq r_i, r_j < n \) such that
\[
i = q_i n + r_i; \\
j = q_j n + r_j.
\]
Plugging this into the left hand side of the first equation, we get
\[
(i + j) \mod n = (q_i + q_j)n + (r_i + r_j) \mod n \\
= (r_i + r_j) \mod n \\
= (i \mod n) + (j \mod n) \mod n.
\]
Similarly, it is easy to show that \((ij) \mod n = (r_i r_j) \mod n\), which implies the second equation.

Algebraic properties. Before we continue, we introduce some notation. Let
\[
\mathbb{Z}_n = \{0, 1, \ldots, n - 1\}
\]
and write \(+_n\) and \(\cdot_n\) for addition and multiplication modulo \(n\), that is,
\[
i +_n j = (i + j) \mod n; \\
i \cdot_n j = (i \cdot j) \mod n.
\]
It is easy to see that \(+_n\) and \(\cdot_n\) are commutative and associative, that is,
\[
i +_n j = j +_n i; \\
(i +_n j) +_n k = i +_n (j +_n k),
\]
and the same for \(\cdot_n\). Furthermore, multiplication distributes over addition, that is,
\[
i \cdot_n (j +_n k) = (i \cdot_n j) +_n (i \cdot_n k).
\]
Note also that \(0 +_n i = i\) for all \(i\) in \(\mathbb{Z}_n\). In other words, \(0\) is the neutral element under addition. Similarly, \(1 \cdot_n j = j\) for all \(j\) in \(\mathbb{Z}_n\) and hence \(1\) is the neutral element under multiplication.

Addition and multiplication modulo \(n\). We may be tempted to use modular arithmetic for the purpose of transmitting secret messages. As a first step, the message is interpreted as an integer, possibly a very long integer. For example, we may write each letter in ASCII and read the bit pattern as a number. Then we concatenate the numbers. Now suppose Alice and Bob agree on two integers, \(n \geq 1\) and \(a\), and they exchange messages using
\[
P(x) = x +_n a; \\
S(y) = y +_n (-a) = y -_n a.
\]
This works fine but not as a public key cryptography system. Knowing that \(P\) is the same as adding \(a\) modulo \(n\), it is easy to determine its inverse, \(S\). Alternatively, let us use multiplication instead of addition,
\[
P(x) = x \cdot_n a; \\
S(y) = y \cdot_n (-a) = y -_n a.
\]
The trouble now is that division modulo \(n\) is not as straightforward an operation as for integers. Indeed, if \(n = 12\) and \(a = 4\), we have \(0 \cdot 4 = 3 \cdot 4 = 6 \cdot 4 = 9 \cdot 4 = 0 \mod n\). Since multiplication with 4 is not injective, the inverse operation is not well defined. Indeed, \(0 :_n 4\) could be 0, 3, 6, or 9.

Summary. We learned about private and public key cryptography, ways to send a secret message from a sender to a receiver. We also made first steps into number theory, introducing modulo arithmetic and Euclid’s Division Theorem. We have seen that addition and multiplication modulo \(n\) are both commutative and associative, and that multiplication distributes over addition, same as in ordinary integer arithmetic.