Chapter 3. Modular Arithmetic

Many complex cryptographic algorithms are actually based on fairly simple modular arithmetic. In modular arithmetic, the numbers we are dealing with are just integers and the operations used are addition, subtraction, multiplication and division. The only difference between modular arithmetic and the arithmetic you learned in your primary school is that in modular arithmetic all operations are performed regarding a positive integer, i.e. the modulus.

Before going into modular arithmetic, let's review some basic concepts. The division theorem tells us that for two integers a and b where $b \neq 0$, there always exists unique integers q and r such that $a = qb + r$ and $0 \leq r < |b|$. For example, a = 17, b=3, we can find q = 5 and r = 2 so that 17 = 3*5+2. a is called the **dividend**, b is called the **divisor**, q is called the **quotient** and r is called the **remainder**. If $r = 0$, then we say b **divides** a or a is **divisible** by b. This establishes a natural congruence relation on the integers. For a positive integer n, two integers a and b are said to be **congruent modulo** n (or a is congruent to b modulo n), if a and b have the same remainder when divided by n (or equivalently if $a - b$ is divisible by n). It can be expressed as $a \equiv b \mod n$. n is called the **modulus**. For example:

- Two odd numbers are congruent modulo 2 because all odd numbers can be written as 2n+1;
- Two even numbers are congruent modulo 2 because all even numbers can be written as 2n+0;
- 38 ≡ 23 mod 15 because 38 = 15*2 + 8 and 23 = 15 +8;
- -1 ≡ 1 mod 2 because -1 = -1*2+1 and 1 = 0*2+1;
- 8 ≡ 3 mod 5 because 8 = 5+3 and 3 = 0*5+3;
- -8 ≡ 2 mod 5 because -8 = -2*5+2 and 2 = 0*5+2;
- 8 ≢ -8 mod 5 because 8 = 5+3 and -8 = -2*5+2. The remainders 3 and 2 are not the same.

You need to be careful with negative numbers. They are usually not congruent to their positive counter parts, as you can see in the above examples. Congruence is an **equivalence relation**, if a and b are congruent modulo n, then they have no difference in modular arithmetic under modulo n. Because of this, in modular n arithmetic we usually use only n numbers 0, 1, 2, ..., n-1. All the other numbers can be found congruent to one of the n numbers.

So how to perform arithmetic operations with moduli? For **addition**, **subtraction** and **multiplication**, it is quite simple: calculate as in ordinary arithmetic and reduce the result to the smallest positive reminder by dividing the modulus. For example:

- 12+9 ≡ 21 ≡ 1 mod 5
- 12-9 ≡ 3 mod 5
- 12+3 ≡ 15 ≡ 0 mod 5
- 15-23 ≡ -8 ≡ 2 mod 5
- 35*7 ≡ 245 ≡ 0 mod 5
- -47*(5+1) ≡ -282 ≡ 3 mod 5
- 37^3 ≡ 50653 ≡ 3 mod 5 (exponentiation is just a shorthand for repeated multiplication)

Sometimes the calculation can be simplified because for any integer a_1, b_1, a_2 andb_2, if we know that $a_1 \equiv b_1$ mod n and $a_2 \equiv b_2$ mod n then the following always holds:

- $a_1+a_2 \equiv b_1+b_2$ mod n
- $a_1-a_2 \equiv b_1-b_2$ mod n
- $a_1*a_2 \equiv b_1*b_2$ mod n

For example, 35 ≡ 0 mod 5 therefore 35*7 ≡ 0*7 ≡ 0 mod 5. Also 37 ≡ 2 mod 5 so $37^3 \equiv 2^3 \equiv 8 \equiv 3$ mod 5.

But for division, it is not so simple because division is not defined for every number. That means that **it is not always possible to perform division in modular arithmetic**. First of all, as in ordinary arithmetic, division by zero is not defined so 0 cannot be the divisor. The tricky bit is that the multiples of the modulus are congruent to 0. For example, 6, -6, 12, -12, ... are all congruent to 0 when the modulus is 6. So not only 4/0 is not allowed, 4/12 is also not allowed when the modulus is 6. Secondly, going back to the very basics: what does "division" mean in ordinary arithmetic? When we say 12 divided by 4 equals 3, we mean that there is a number 3 such that 3*4 = 12. So division is defined through multiplication. But you run into problems extending this to modular arithmetic. let's have a look at the following table:

Table 3.1. Multiplication modulo 6

*	1	2	3	4	5
1	1	2	3	4	5
2	2	4	0	2	4
3	3	0	3	0	3
4	4	2	0	4	2
5	5	4	3	2	1

Suppose you are working in mod 6 and want to compute 4/5. As we said before, you actually need to find *x* such that 5**x* ≡ 4 mod 6. From the above table, we can find that 2 and only 2 satisfies this equation. That means 4/5 ≡ 2 mod 6. Now suppose you want to compute 4/2 ≡ ? mod 6. It seems easy because 2*2 ≡ 4 mod 6. However, there is another possibility: 2*5 ≡ 4 mod 6. This time division is not uniquely defined, because there are two numbers that can multiply by 2 to give 4. In such cases, division is not allowed.

Then when modular division is defined? When the **multiplicative inverse** (or just inverse) of the divisor exists. The inverse of an integer *a* under modulus *n* is an integer *b* such that *a*b* ≡ 1 mod *n*. An integer can have either one or no inverse. The inverse of *a* can be another integer or *a* itself. In the above table, we can see that 1 has an inverse, which is itself and 5 also has an inverse which is also itself. But 2, 3 and 4 do not have inverses. Whether an integer has the inverse or not depends on the integer itself and also the modulus. Compare the follwing table to table 1:

Table 3.2. Multiplication modulo 5

*	**1**	**2**	**3**	**4**
1	1	2	3	4
2	2	4	1	3
3	3	1	4	2
4	4	3	2	1

You can see that when the modulus is 6, 2 has no inverse. But when the modulus is 5, the inverse of 2 is 3. The rule is that the inverse of an integer *a* exists iff *a* and the modulus *n* are **coprime**. That is, the only positive integer which divides both *a* and *n* is 1. In particular, when *n* is **prime**, then every integer except 0 and the multiples of *n* is coprime to *n*, so every number except 0 has a corresponding inverse under modulo *n*. You can verify this rule with table 1 and 2.

Sometimes it is easy to determine whether two integers are coprime. But most of the time it is not easy. For example, are 357 and 63 coprime? You may not be able to answer immediately. Fortunately, we can use the *Euclidean algorithm* to find out. The Euclidean algorithm describes how to find what is called the **greatest common divisor** (gcd) of two positive integers. Of course, if the gcd of two integers is 1, they are coprime. Let me show you by an example.

We start with two positive integers 357 and 63. **The first step of the Euclidean algorithm is to divide the bigger integer by the smaller one**, so we have:

- 357÷63, quotient = 5 remainder = 42

Then **divide the divisor in last step by the remainder**:

- 63÷42, quotient =1 remainder=21

Continue to divide the previous divisors by the remainders, until the remainder is 0:

- 42÷21, quotient =2 remainder =0

The divisor in the last step is the gcd of the two input integers. To see why the algorithm works, we follow the division steps backwards. From the last step, we know that 21 divides 42. In the step before, we have 63 = 1*42 +21. Because 21 divides both 42 and 21, it must also divide 63. In the first step, we have 357 = 5*63 +42, again 21 divides both 63 and 42 so it must also divide 357. Since 21 divides both 63 and 357, it is indeed a common divisor of those two integers. Now we need to prove that it is the greatest. The proof is based on a theorem which says:

- For any non-negative integers a and b, and any integers x and y, $c = x*a + y*b$ must be a multiple of the gcd of a and b.

What we want to show is that 21 =x*357 + y*63 for some x and y. If this is true, then 21 must be the gcd (why? Figuring this out is left to you as an exercise). Now let's start:

- From step 1, we have 357-5*63=42
- From step 2. we have 63-42=21
- Substitutes 42 with 357 -5*63, now we have 21 = 63-357+5*63 = -1*357+6*63

So the Euclidean algorithm indeed outputs the gcd. If the gcd is 1, we can conclude a and b are coprime.

Knowing that an integer a and a modulus n are coprime is not enough. How can we find the multiplicative inverse of a? Well, since the gcd of a and n is 1, we know we can find a pair (x,y) such that 1 = $x*a+y*n$. Then $x*a = -y*n+1$. That means $x*a \equiv 1$ mod n, in other words, x is the multiplicative inverse of a under modulo n. This can be done by running an extended version of Euclidean algorithm which tracks x when computing the gcd. In the extended Euclidean algorithm, we first initialise x_1 =0 and x_2 =1, then in the following steps, compute $x_i = x_{i-2} - x_{i-1}q_{i-2}$ where q_{i-2} is the quotient computed in step i-2. When the remainder becomes 0, continue the calculation of x for one more round. The final x is the inverse. Here is an example that shows how to find the inverse of 15 when the modulus is 26:

- step 1: 26÷15, quotient q_1= 1, remainder = 11, x_1 = 0
- step 2: 15÷11, quotient q_2 = 1, remainder = 4, x_2 = 1
- step 3: 11÷4, quotient q_3 = 2, remainder = 3, $x_3 = x_1-x_2q_1$ = 0- 1*1 = -1
- step 4: 4÷3, quotient q_4 = 1, remainder = 1, $x_4 = x_2-x_3q_2$ = 1- (-1)*1 = 2
- step 5: 3÷1, quotient q_5 = 3, remainder = 0, $x_5 = x_3-x_4q_3$ = -1- 2*2 = -5

- step 6: $x_6 = x_4 - x_5 q_4$ = 2- (-5)*1 = 7

To verify, 15*7 = 105 = 4*26+1, so 15*7 ≡ 1 mod 26, which means 7 is the multiplicative inverse of 15 under modulo 26.

Useful Links

- Modular arithmetic calculator (addition, multiplication and exponentiation only)
- GCD, multiplicative Inverse calculator (on the bottom of the page)

Prev
Chapter 2. Unique-Prime-Factorisation Theorem

Home

Next
Chapter 4. Finite Fields