# Introduction to Number Theory

*The study of the integers*

# Divisibility of Integers, $\mathbb{Z}$



The set of integers

$$\mathbb{Z} = \{\ldots -3, -2, -1, 0, 1, 2, 3, \ldots\}.$$

*In this lecture, if nothing is said about a variable, it is an integer.*

**Def.** We say that $a$ *divides* $b$ if there is an integer $k$ such that

$$b = a \cdot k.$$

We write $a \mid b$ if $a$ divides $b$. Otherwise, we write $a \nmid b$.

For example, $7 \mid 63$, because $7 \cdot 9 = 63$.

If $a$ divides $b$, then $b$ is a multiple of $a$.

# Definition for $a \mid b$

$a \cdot k = b$ for some integer $k$

notation: $a \mid b$

reads as "$a$ divides $b$"

alternatively: "$b$ is a multiple of $a$"

Example: $6 \mid 54$

# Divisibility



**Lemma 1.** If $a \mid b$ then $a \mid bc$ for all $c$.

Proof. Since $a \mid b$, $\exists k$ such that $ak = b$. Thus $bc = akc$, and therefore by definition, $a|bc$. $\square$

Example: $5 \mid 15$, then,
$5 \mid 30$,
$5 \mid -45$,
$5 \mid -150$.

# Divisibility



**Lemma 2.** If $a \mid b$ and $b \mid c$, then $a \mid c$.

Proof. There exist integers $m$ and $n$ such that $b = am$ and $c = bn$. So, $c = bn = amn$, and therefore, $a \mid c$. $\square$

Example: $7 \mid 14$, and $14 \mid 280$, therefore, by this lemma, $7 \mid 280$.

# Divisibility

**Lemma 3.** If $a \mid b$ and $a \mid c$, then $a \mid (mb + nc)$ for all $m$ and $n$.

Example: $5 \mid 100$ and $5 \mid 15$. Therefore,
$5 \mid 115$,
$5 \mid 1030$,
$5 \mid -245$.

**Lemma 4.** For all $c \neq 0$, $a \mid b$ if and only if $ac \mid bc$.

Example: $17 \mid 34$ if and only if $-170 \mid -340$.

# Division algorithm

**Theorem.** The Division Algorithm. Let $a$ be an integer and $d$ a positive integer. Then there are *unique* integers $q$ and $r$, such that $0 \leq r < d$ and

$$a = dq + r.$$

$d$ = divisor

$a$ = dividend

$q$ = quotient

$r$ = remainder

How can we prove that $q$ and $r$ are unique?

# Division algorithm

Assume that they are not unique, then there exist at least two distinct pairs of $q$ and $r$:

$$a = dq_1 + r_1, \text{ and } a = dq_2 + r_2$$

Subtract one from another:

$$0 = d(q_1 - q_2) + (r_1 - r_2)$$

Since $0 \le r1, r2 < d$, the difference of the remainders is

$$-d < r_1 - r_2 < d,$$

Therefore the same is true for the other term:

$$-d < d(q_1 - q_2) < d$$

It can happen only if $q_1 - q_2 = 0$, which also implies that $r_1 - r_2 = 0$. By contradiction, $q$ and $r$ are unique.

# Prime numbers



**Def.** A number $p > 1$ with no positive divisors other than 1 and itself is called a *prime*.

Every other number greater than 1 is called *composite*.

The number 1 is considered neither prime nor composite.

The first few primes are 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37 ...

**Theorem.** Let $p$ be a prime. If

$$p \mid a_1 a_2 \cdot \ldots \cdot a_n,$$

then $p$ divides some $a_i$.

Example: If you know that $19 \mid 403 \cdot 629$, then you know that either $19 \mid 403$ or $19 \mid 629$, though you might not know which.

# Greatest common divisor



**Def.** The *greatest common divisor* of two positive integers $a_0$ and $a_1$, denoted $\gcd(a_0, a_1)$ is the largest integer $g$ that divides both $a_0$ and $a_1$.

Example. Find $\gcd(12, 18)$.

First, list all positive $x$ such that $x \mid 12$:

$$1, 2, 3, 4, 6, 12.$$

Then, list all positive $x$ such that $x \mid 18$:

$$1, 2, 3, 6, 9, 18.$$

The largest in the both lists, 6, is the $\gcd(12, 18)$.

# Greatest common divisor



For two positive integers $a$ and $b$:

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

# Greatest common divisor

For two positive integers $a$ and $b$:

$$\text{If } a \mid b, \text{ what is the } \gcd(a, b)?$$

# Greatest common divisor



For two positive integers $a$ and $b$:

$$\text{If } a \mid b, \text{ what is the } \gcd(a, b)?$$

$a$ is one of the divisors of $b$. But $a$ is the greaterst possible divisor of itself.

Thus $a$ is the greatest common divisor.

So, if $b = ka$,

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

# Greatest common divisor

Let's find a way to compute $\gcd(a_0, a_1)$ without simply trying every single positive integer from 1 to $\min(a_0, a_1)$.

For simplicity, without loss of generality we can say that $a_0 \geq a_1$.

Then, by the division algorithm,

$$a_0 = a_1 q + r. \qquad (\text{and } q \geq 1)$$

**Lemma.** If $a_0 = a_1 q + r$ then $\gcd(a_0, a_1) = \gcd(a_1, r)$.

# Greatest common divisor

**Lemma.** If $a_0 = a_1 q + r$ then $\gcd(a_0, a_1) = \gcd(a_1, r)$.

Proof. We are going to prove that the common divisors of $a_0$ and $a_1$ are the same as the common divisors of $a_1$ and $r$.

In other words, we have to prove that
$d$ divides $a_0$ and $a_1$ *if and only if* $d$ divides $a_1$ and $r$.

# Greatest common divisor

**Lemma.** If $a_0 = a_1 q + r$ then $\gcd(a_0, a_1) = \gcd(a_1, r)$.

Proof. We are going to prove that the common divisors of $a_0$ and $a_1$ are the same as the common divisors of $a_1$ and $r$.

In other words, we have to prove that
$d$ divides $a_0$ and $a_1$ *if and only if* $d$ divides $a_1$ and $r$.

($\Rightarrow$) Let $d$ be a divisor of $a_0$ and $a_1$, that is $d \mid a_0$ and $d \mid a_1$.

By Lemma 3, $d \mid (a_0 - a_1 q)$, and since $r = a_0 - a_1 q$, we get $d \mid r$. Thus $d$ divides $a_1$ and $r$.

# Greatest common divisor



**Lemma.** If $a_0 = a_1 q + r$ then $\gcd(a_0, a_1) = \gcd(a_1, r)$.

Proof. We are going to prove that the common divisors of $a_0$ and $a_1$ are the same as the common divisors of $a_1$ and $r$.

In other words, we have to prove that
$d$ divides $a_0$ and $a_1$ *if and only if* $d$ divides $a_1$ and $r$.

($\Rightarrow$) Let $d$ be a divisor of $a_0$ and $a_1$, that is $d \mid a_0$ and $d \mid a_1$.

By Lemma 3, $d \mid (a_0 - a_1 q)$, and since $r = a_0 - a_1 q$, we get $d \mid r$. Thus $d$ divides $a_1$ and $r$.

($\Leftarrow$) Let $d$ be a divisor of $a_1$ and $r$, that is $d \mid a_1$ and $d \mid r$.

Again, by Lemma 3, $d \mid (a_1 q + r)$, so $d \mid a_0$. So, $d$ divides $a_0$ and $a_1$.

Threofore, $\gcd(a_0, a_1) = \gcd(a_1, r)$. $\square$

# Greatest common divisor



Compute $\gcd(a_0, a_1)$.

1) We find the quotient and the remainder:

$$a_0 = q_1 a_1 + r_1$$

Let $a_2 = r_1$: $\quad \gcd(a_0, a_1) = \gcd(a_1, r_1) = \gcd(a_1, a_2)$.

2) Find the new quotient and the remainder:

$$a_1 = q_2 a_2 + r_2$$

Let $a_3 = r_2$: $\quad \gcd(a_1, a_2) = \gcd(a_2, r_2) = \gcd(a_2, a_3)$.

3) $\ldots$
continue the process, computing $a_4$, $a_5$, $a_6$, $\ldots$ until what?

# Greatest common divisor



Compute $\gcd(300, 18)$.

$a_0 = 300$,
$a_1 = 18$. $\gcd(300, 18)$?

$$300 = 16 \cdot 18 + 12$$

$a_2 = 12$. $\gcd(18, 12)$?

$$18 = 1 \cdot 12 + 6$$

$a_3 = 6$. $\gcd(12, 6)$?

$$12 = 2 \cdot 6 + 0$$

$6 \mid 12$, and $6 \mid 6$. And there is simply no larger divisors of 6, so $\gcd(300, 18) = \gcd(12, 6) = 6$.

# Greatest common divisor



So, to compute $\gcd(a_0, a_1)$, you compute a sequence remainders $a_k$, until some $a_k$ divides $a_{k-1}$, and therefore

$$\gcd(a_0, a_1) = \gcd(a_{k-1}, a_k) = a_k,$$

where $a_k$ is *the last non-zero remainder*.

This procedure for computing GCD is called *Euclid's algorithm*.

# Greatest common divisor

If we use the following notation for the remainder of a division:

$$c = a \ \mathbf{rem} \ b$$

Euclid's algorithm works as follows:

$$\begin{aligned} \gcd(300, 18) &= \gcd(18, \underbrace{300 \ \mathbf{rem} \ 18}_{=12}) \\ &= \gcd(12, \underbrace{18 \ \mathbf{rem} \ 12}_{6}) \\ &= \gcd(12, 6) \\ &= 6 \end{aligned}$$

In C and C++, there is a similar operator % (though it behaves differently when $a$ or $b$ are negative)

# Greatest common divisor



Compute

$$\gcd(1110, 777)$$

# Efficiency of Euclid's algorithm

*Worst case number of steps.*

In how many steps $k$ the Euclidean algorithm computes $\gcd(a_0, a_1)$?

In the best case, if $a_1 \mid a_0$, we immediately find that the GCD is equal to $a_1$, and it takes just a single step.

What is the worst possible input?
That is, what are the smallest integers $a_0 \geq a_1$, such that $\gcd(a_0, a_1)$ is computed in $k$ steps.

# Efficiency of Euclid's algorithm



What are the smallest integers $a_0 \geq a_1$, such that $\gcd(a_0, a_1)$ is computed in $k$ steps.

We are going to construct a sequence of $a_i$ such that $a_k$ is the $\gcd(a_0)$, and $a_0$ is the smallest possible.

Observe that

$$a_k = \gcd(a_0, a_1) \geq 1$$

$$a_{k-1} \geq 2$$

We want to construct all the previous terms of the sequence

$$a_{k-2},\ a_{k-3},\ \ldots,\ a_0$$

in this backward order.

# Efficiency of Euclid's algorithm



Observe that the sequence of $a_i$ is strictly decreasing ($a_i > a_{i+1}$).

The recurrence looks like this

$$a_i = q_{i+1} a_{i+1} + a_{i+2},$$

and the quotient $q_{i+1} \geq 1$.

Thus

$$a_i \geq a_{i+1} + a_{i+2}$$

If, eventually, we are want to end up with the smallest possible $a_0$, then on each step, when constructing $a_i$ from $a_{i+1}$ and $a_{i+2}$, we should choose the smallest possible number:

$$a_i = a_{i+1} + a_{i+2}$$

# Efficiency of Euclid's algorithm



Let's summarize our analysis.

We are constructing a decreasing sequence of positive integers

$$a_0 > a_1 > a_2 > \ldots > a_{k-1} > a_k$$

Such that

$$\begin{aligned} a_k &\geq 1 \\ a_{k-1} &\geq 2 \\ a_i &= a_{i+1} + a_{i+2} \end{aligned}$$

# Efficiency of Euclid's algorithm



$$a_0 > a_1 > a_2 > \ldots > a_{k-1} > a_k$$

$$\begin{aligned} a_k &\geq 1 \\ a_{k-1} &\geq 2 \\ a_i &= a_{i+1} + a_{i+2} \end{aligned}$$

The Fibonacci numbers satisfy all the requirements

$$\begin{array}{cccccccc} F_1 & F_2 & F_3 & F_4 & \ldots & F_{k+2-i} & \ldots & F_{k+2} \\ \hline 1 & 1 & 2 & 3 & & & & \\ & a_k & a_{k-1} & a_{k-2} & \ldots & a_i & \ldots & a_0 \end{array}$$

$$F_1 = 1; \quad F_2 = 1; \quad F_i = F_{i-1} + F_{i-2}$$

# Efficiency of Euclid's algorithm

Lets see, how bad they are:

Compute $\gcd(21, 34)$.

$$\begin{aligned}
a_0 &= 34, \\
a_1 &= 21, \quad 34 = 1 \cdot 21 + 13 \\
a_2 &= 13, \quad 21 = 1 \cdot 13 + 8 \\
a_3 &= 8, \quad 13 = 1 \cdot 8 + 5 \\
a_4 &= 5, \quad 8 = 1 \cdot 5 + 3 \\
a_5 &= 3, \quad 5 = 1 \cdot 3 + 2 \\
a_6 &= 2, \quad 3 = 1 \cdot 2 + 1 \\
a_7 &= 1, \quad 2 = 2 \cdot 1
\end{aligned}$$

$\gcd(21, 34) = a_7 = 1$. And it took $k = 7$ steps.

The sequence of $a_i$ is the Fibonacci sequence, $a_i = F_{k+2-i}$.

# Efficiency of Euclid's algorithm

The algorithm is still very fast, even on the worst input:

Look at the Fibonacci sequence:

$$1,\quad 1,\quad 2,\quad 3,\quad 5,\quad 8,\quad 13,\quad 21,\quad 34,\quad 55,\quad 89,\quad 144,\quad 233,\quad 377,\quad 610,\quad 987,\quad 1597,\quad 2584,\quad 4181,\quad 6765,\quad 10946,\quad 17711,\quad 28657,\quad 46368,\quad 75025,\quad 121393,\quad 196418,\quad 317811,\quad 514229,\quad 832040,\quad 1346269,\quad 2178309,\quad 3524578,\quad 5702887,\quad 9227465, \ldots$$

In the limit $\dfrac{F_n}{F_{n-1}}$ approaches $\phi \approx 1.618$.

# Efficiency of Euclid's algorithm



It can be shown that $F_n \geq c\phi^{n-1}$ for some constant $c$, so $a_0 \geq c\phi^{n-1}$.

Given $a_0$, the number of steps for the Euclidean algorithm is

$$k \leq \log_\phi \frac{a_0}{c} - 1$$

So the complexity (number of steps) is *logarithmic* in $a_0$.

*Complexity, when the input is a number*
Usually, when the input is a number, the length of the input is measured as the length of the binary string that represents the input. A number $a_0$ can be represented by $\lceil \log_2 a_0 \rceil$ bits.

$$k \leq \log_\phi \frac{a_0}{c} - 1 = \frac{1}{\log_2 \phi} \log_2 a_0 - \log_\phi c - 1 = C_1 \log_2 a_0 + C_2$$

Therefore, the time complexity of the Euclidean algorithm is *linear in the number of bits* required to represent $a_0$.

# Code v1.0



The sender wants to send a message "victory" to the receiver.

**Beforehand.** The sender and receiver agree on a secret key, which is a large *prime number*

$$p = 22801763489$$

# Code v1.0

**Encryption.**

(1) The sender transforms a string of characters into a number:

$$\begin{array}{ccccccc} \text{“v} & \text{i} & \text{c} & \text{t} & \text{o} & \text{r} & \text{y”} \\ 22 & 09 & 03 & 20 & 15 & 18 & 25 \end{array}$$

(2) The resulting number is padded with a few more digits to make a *prime number*

$$m = 2209032015182513$$

(3) After that, the sender encrypts the message $m$ by computing

$$m' = m \cdot p$$

# Code v1.0



**Decryption.** The receiver decrypts $m$ by computing

$$\frac{m'}{p} = \frac{m \cdot p}{p} = m$$

# Code v1.0

The code raises a couple immediate questions.

1. **How can the sender and receiver ensure that $m$ and $p$ are prime numbers?** The general problem of determining whether a large number is prime or composite has been studied for centuries, and reasonably good primality tests were known in the past. In 2002, Manindra Agrawal, Neeraj Kayal, and Nitin Saxena announced a primality test that is guaranteed to work on a number $n$ in about $(\log n)^{12}$ steps.

2. **Is the code secure?** If the adeversary receives the encrypted message $m'$, how easily he can recove the original message $m$? This is the problem of factoring $m' = m \cdot p$.

   Despite immense efforts, no really efficient factoring algorithm has ever been found. It appears to be a fundamentally difficult problem, though a breakthrough is not impossible.

# Code v1.0



Now, consider a situation, when your adversary received two encrypted messages

$$m' = m \cdot p \qquad \text{and} \qquad n' = n \cdot p$$

# GCD is a linear combination



**Theorem** (Bezout's Theorem). If $a$ and $b$ are positive integers, then there exist integers $s$ and $t$ such that $\gcd(a, b) = sa + tb$.

Exmaple: $\gcd(52, 44) = 4$

$$6 \cdot 52 + (-7) \cdot 44 = 4$$

# Factorization of positive integers



**Theorem** (Fundamental theorem of arithmetic). Every positive integer $n$ can be written in a unique way as a product of primes

$$n = p_1 \cdot p_2 \cdot \ldots \cdot p_j \qquad (p_1 \leq p_2 \leq \ldots \leq p_j)$$