4.1 Divisibility and Modular Arithmetic

Wenjing Li

wjli@bupt.edu.cn

School of Computer Science

Beijing University of Posts & Telecommunications

Divisibility and Modular Arithmetic

- Of course, you already know what the integers are, and what division is…
- **But:** There are some specific notations, terminology, and theorems associated with these concepts which you may not know.
- These form the basics of ***number theory***.
  - Vital in many important algorithms today (hash functions, cryptography, digital signatures).

Divides, Factor, Multiple

- **Definition:**
  - Let $a,b\in\mathbf{Z}$ with $a\neq 0$.
  - $a|b \;\equiv$ "*a divides b*" $:\equiv (\exists c\in\mathbf{Z}: b=ac)$
    "There is an integer *c* such that *c* times *a* equals *b*."
  - Example: $3|-12 \Leftrightarrow$ **True**, but $3|7 \Leftrightarrow$ **False**.
  - Iff *a* divides *b*, then we say *a* is a *factor* or a *divisor* of *b*, and *b* is a *multiple* of *a*.
- **Example:**
  - "*b* is even" $:\equiv 2|b$. Is 0 even? Is -4?

The Divides Relation

- **Theorem 1:** $\forall a,b,c \in \mathbf{Z}, a \neq 0$:
  1. $a|0$
  2. $(a|b \wedge a|c) \rightarrow a \mid (b + c)$
  3. $a|b \rightarrow a|bc$
  4. $(a|b \wedge b|c) \rightarrow a|c$
- **Proof** of (2):
  - Let a, b, c be any integers such that $a|b$ and $a|c$, and show that $a \mid (b + c)$.
  - By defn. of $|$, we know $\exists s$: $b=as$, and $\exists t$: $c=at$. Let s, t, be such integers. Then $b+c = as + at = a(s+t)$.
  - so $\exists u$: $b+c=au$, namely $u=s+t$. Thus $a|(b+c)$.

Corollary 1

- $\forall a,b,c,m,n \in \mathbf{Z}$:
  - $(a|b \wedge a|c) \rightarrow a \mid (mb + nc)$

The Division "Algorithm"

- It's really just a *theorem*, not an algorithm…
  - Only called an "algorithm" for historical reasons.
- **Theorem:**
  - For any integer *dividend* a and *divisor* $d \neq 0$, there is a unique integer *quotient* q and *remainder* $r \in \mathbf{N}$ such that $a = dq + r$ and $0 \leq r < |d|$.
  - Formally, **the theorem** is: $\forall a,d \in \mathbf{Z}, d \neq 0: \exists! q,r \in \mathbf{Z}: 0 \leq r < |d|, a = dq + r$.
  - We can find q and r by: $q = \lfloor a/d \rfloor$, $r = a - qd$.
  - $q = a \text{ div } d$, $r = a \text{ mod } d$

The Division "Algorithm"

- Example 3
  - 101/11？
  - What are the quotient and remainder?

- Example 4
  - -11/3?
  - What are the quotient and remainder?

Remainder cannot be negative.

The mod operator

- **Definition:**
  - An integer "division remainder" operator.
  - Let $a,d \in \mathbf{Z}$ with $d>1$. Then a **mod** d denotes the remainder r from the division "algorithm" with dividend a and divisor d;
  - *i.e.* the remainder when a is divided by d.
  - Using *e.g.* long division.
  - We can compute (a **mod** d) by: $a - d \cdot \lfloor a/d \rfloor$.
- In C/C++/Java languages, "%" = mod.

Arithmetic Modulo m (模算数)

- **Definition:**
  - Z_m, the set of nonnegative integers less than m, that is, the set $\{0, 1, \ldots, m-1\}$.
  - we define addition of these integers, denoted by $+_m$ *by* $a +_m b = (a + b) \bmod m$, where the addition on the right-hand side of this equation is the ordinary addition of integers,
  - we define multiplication of these integers, denoted by $\cdot_m$ *by* $a \cdot_m b = (a \cdot b) \bmod m$
- **Example 7**
  - Use the definition of addition and multiplication in Z_m to find $7 +_{11} 9$ and $7 \cdot_{11} 9$.

Modular Congruence (模同余)

- Let $a,b\in\mathbf{Z}$, $m\in\mathbf{Z}^+$.

  Where $\mathbf{Z}^+=\{n\in\mathbf{Z} \mid n>0\}=\mathbf{N}-\{0\}$ (the + integers).

- Then *a is congruent to b modulo m* **(a与b模m同余)**, written "$a\equiv b \pmod{m}$", iff $m \mid a-b$.
  - Note: this is a different use of "$\equiv$" than the meaning "is defined as" I've used before.
  - Proof: $a=sm+r$, $b=tm+r$, $(a-b)=(s-t)m$

- It's also equivalent to: $(a-b) \bmod m = 0$.

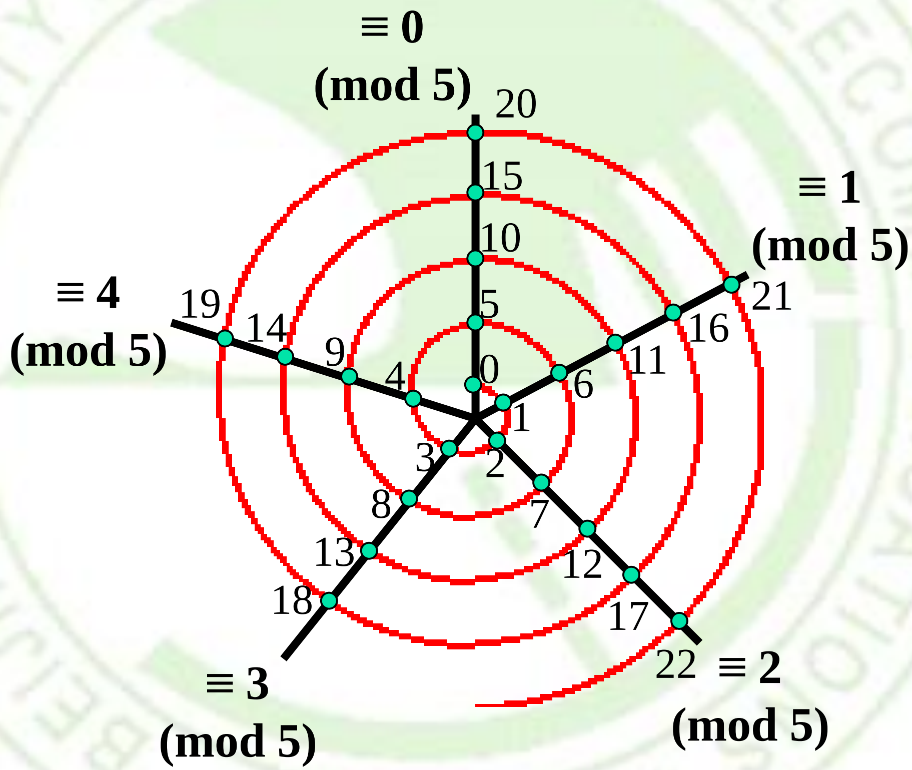
$$\boldsymbol{a \equiv (a \bmod m) \pmod{m}}$$

KARL FRIEDRICH GAUSS
Germany

Modular Congruence (模同余)

- **Spiral Visualization of mod: Example shown**

  modulo-5 arithmetic



Useful Congruence Theorems

- **Theorem 3:**

  Let $a,b \in \mathbf{Z}$, $m \in \mathbf{Z}^+$. Then:
  $a \equiv b \pmod{m} \Leftrightarrow a \pmod{m} \equiv b \pmod{m}$.

- **Theorem 4:**

  Let $a,b \in \mathbf{Z}$, $m \in \mathbf{Z}^+$. Then:
  $a \equiv b \pmod{m} \Leftrightarrow \exists k \in \mathbf{Z}\ a = b + km$.

- **Theorem 5:**

  Let $a,b,c,d \in \mathbf{Z}$, $m \in \mathbf{Z}^+$. Then
  if $a \equiv b \pmod{m}$ and $c \equiv d \pmod{m}$,
  then $a+c \equiv b+d \pmod{m}$ and $ac \equiv bd \pmod{m}$

  和同余 积同余

Useful Congruence Theorems

- **Corollary 2**

  $a \equiv (a \bmod m) \pmod m$

  - Let m be a positive integer and a and b be integers. Then

  $(a+b) \bmod m = ((a \bmod m)+(b \bmod m)) \bmod m$ （和同余推理）

  $ab \bmod m = ((a \bmod m)(b \bmod m)) \bmod m$ （积同余推理）

- **Proof (2):**

  We know ***$a \equiv (a \bmod m) \pmod m$, $b \equiv (b \bmod m) \pmod m$***

  $m|a-(a \bmod m) \Leftrightarrow a-(a \bmod m) = sm \Leftrightarrow a \bmod m = a-sm$

  $m|b-(b \bmod m) \Leftrightarrow b-(b \bmod m) = tm \Leftrightarrow b \bmod m = b-tm$

  $((a \bmod m)(b \bmod m)) \bmod m = ((a-sm)(b-tm)) \bmod m$

  $= (ab - atm - sbm + stmm) \bmod m = ab \bmod m$

What are the benefits?

4.2 Integers Representations & Algorithms

Wenjing Li

wjli@bupt.edu.cn

School of Computer Science

Beijing University of Posts & Telecommunications

Particular Bases of Interest

- Ordinarily, we use *base-10, base-2, base-8 and base-16* representations of numbers.
  - Base b=10 (decimal):
    10 digits: 0,1,2,3,4,5,6,7,8,9.
  - Base b=2 (binary):
    2 digits: 0,1. ("Bits"="binary digits.")
  - Base b=8 (octal):
    8 digits: 0,1,2,3,4,5,6,7.
  - Base b=16 (hexadecimal):
    16 digits: 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F

Used only because we have 10 fingers

Used internally in all modern computers

Octal digits correspond to groups of 3 bits

Hex digits give groups of 4 bits

Base-b number systems

- But, any base $b>1$ will work.
- For any positive integers n,b, there is a unique sequence $a_k\, a_{k-1}\ldots\, a_1 a_0$ of *digits* $a_i<b$ such that:

$$n = \sum_{i=0}^{k} a_i b^i$$

The "*base b expansion of n*"

Base-b number systems

- **Converting to Base-b:**
- An algorithm, informally stated
  - To convert any integer n to any base $b>1$:
  - To find the value of the *rightmost* (lowest-order) digit, simply compute n mod b.
  - Now, replace n with the quotient $\lfloor n/b \rfloor$.
  - Repeat above two steps to find subsequent digits, until n is gone (=0).

Exercise for offline: Write this out in pseudocode…

- **Addition of Binary Numbers**

procedure $add(a_{n-1}\ldots a_0, b_{n-1}\ldots b_0$: binary representations of non-negative integers $a,b)$

$carry := 0$

for $bitIndex := 0$ **to** $n-1$ { {go through bits}

$bitSum := a_{bitIndex}+b_{bitIndex}+carry$ {2-bit sum}

$s_{bitIndex} := bitSum$ **mod** 2 {low bit of sum: remainder}

$carry := \lfloor bitSum / 2 \rfloor$ } {high bit of sum: quotient}

$s_n := carry$

return $s_n\ldots s_0$: binary representation of integer s

Algorithms for Integer Operations

- **Multiplication of Binary Numbers**

procedure *multiply*($a_{n-1}\ldots a_0$, $b_{n-1}\ldots b_0$: binary representations of $a,b\in\mathbf{N}$)

$product := 0$

for $i := 0$ to $n-1$

if $b_i = 1$ **then**

$product := add(a_{n-1}\ldots a_0\mathbf{0}^{\boldsymbol{i}}, product)$

return *product*

i extra 0-bits appended after the digits of a (shift i bits)

```
      1011    a
      1001    b
      ----------
      1011
     00000
    000000
   1011000
   -------
   1100011
```

Algorithms for Integer Operations

- **Binary Division with Remainder**

procedure *div-mod*($a_{n-1}\ldots a_0$, $d_{n-1}\ldots d_0$: binary representations of $a,d \in \mathbf{Z}^+$) {Quotient & rem. of a/d.}

n := max(length of a in bits, length of d in bits)

for $i := n-1$ **downto** 0

if $a \geq d0^i$ **then** {Can we subtract at this position?}

$q_i := 1$ {This bit of quotient is 1.}

$a := a - d0^i$ {Subtract to get remainder.}

else $q_i := 0$ {This bit of quotient is 0.}

$r := a$

return q,r {q = quotient, r = remainder}

Modular Exponentiation Problem

- **Problem:**
  - Given large integers b (base), n (exponent), and m (modulus), efficiently compute $\boldsymbol{b^n \bmod m}$.
  - Note that b^n itself may be completely infeasible to compute and store directly (e.g. 999^{1279}).
  - *E.g.* if n is a 1,000-bit number, then b^n itself will have far more digits than there are atoms in the universe!
- Yet, this is a type of calculation that is commonly required in modern cryptographic algorithms! Both encryption and decryption.

Modular Exponentiation(模指数)

The binary expansion of n

$b^n \bmod m$

- **Note that:**

$$b^n = b^{\overbrace{n_{k-1}\cdot 2^{k-1}+n_{k-2}\cdot 2^{k-2}+\ldots+n_0\cdot 2^0}^{\text{The binary expansion of } n}}$$

$$= (b^{2^{k-1}})^{n_{k-1}} \times (b^{2^{k-2}})^{n_{k-2}} \times \ldots \times (\overbrace{b^{2^0}}^{= b^1 = b})^{n_0}$$

- Crucially, we can do the **mod** m operations <u>as we go along</u>, because of the various **identity laws** of modular arithmetic. – All the numbers stay small.

ab mod m=((a mod m)(b mod m)) mod m

- Features of binary expansion of n:
  - We can compute b to various powers of 2 by repeated squaring.
  - Then multiply them into the partial product, or not, depending on whether the corresponding n_i bit is 1.

Modular Exponentiation(模指数)

b^n mod m

$$b^n = b^{n_{k-1}\cdot 2^{k-1}+n_{k-2}\cdot 2^{k-2}+\ldots+n_0\cdot 2^0}$$

$$= (b^{2^{k-1}})^{n_{k-1}} \times (b^{2^{k-2}})^{n_{k-2}} \times \ldots \times (b^{2^0})^{n_0}$$

ab mod m=((a mod m)(b mod m)) mod m

- **E.g.** $2^{11}=2^{(1011)}=2^{(8+0+2+1)}=2^8\cdot 2^2\cdot 2^1$
- 2^{11} (mod m) = $(2^8\cdot 2^2\cdot 2^1)$ (mod m)

  = ((2^8 mod m) (2^2 mod m) (2^1 mod m)) (mod m)

- 2^2 mod m= ((2^1 mod m) (2^1 mod m)) (mod m)= (2^1 mod m)^2 (mod m)
- 2^4 mod m= ((2^2 mod m) (2^2 mod m)) (mod m)= (2^2 mod m)^2 (mod m)
- 2^8 mod m= ((2^4 mod m) (2^4 mod m)) (mod m)= (2^4 mod m)^2 (mod m)

思路：从最右边开始求模，然后向左，依次和左边求模结果相乘再求模. 其中左边一位的模是前一位求模结果的平方再求模，将前一个结果记录下来后面直接使用

Modular Exponentiation (模指数)

- **Modular Exponentiation**

$b^n \bmod m$

procedure *modularExp* (b: integer, $n = (a_{k-1} \ldots a_0)_2$, m: positive integers)

$x := 1$ {result will be accumulated here}

$b^{2^i} := b \ \mathbf{mod}\ m$ { b^{2^i} is power; i=0 initially, b^{2^0}} **power**

for $i := 0$ to $k-1$ {go thru all k bits of n}

  if $a_i = 1$ **then** $x := (x \cdot b^{2^i})\ \mathbf{mod}\ m$ {when i^{th} bit is 1}

  $b^{2^i} := (b^{2^i} \cdot b^{2^i})\ \mathbf{mod}\ m$ {shift to $(i+1)^{th}$ }

return x

$$b^{2^{i+1}} = b^{2 \cdot 2^i} = (b^{2^i}) \cdot (b^{2^i})$$

前一个求模结果的平方再求模

$b^2 \bmod m = ((b \bmod m)(b \bmod m)) \bmod m$

Modular Exponentiation (模指数)

- **Example 12:**
  - Use Algorithm ***modularExp*** to find 3^{644} mod 645.
  - $644=(1010000100)_2$

Initial x=1
Initial power: $b^{2^0}=3^{2^0}$ mod 645=3

$i=0$: Because $a_0=0$, we have $x=1$ and $power=3^2 \bmod 645 = 9 \bmod 645 = 9$;
$i=1$: Because $a_1=0$, we have $x=1$ and $power=9^2 \bmod 645 = 81 \bmod 645 = 81$;
$i=2$: Because $a_2=1$, we have $x=1\cdot 81 \bmod 645 = 81$ and $power=81^2 \bmod 645 = 6561 \bmod 645 = 111$;
$i=3$: Because $a_3=0$, we have $x=81$ and $power=111^2 \bmod 645 = 12{,}321 \bmod 645 = 66$;
$i=4$: Because $a_4=0$, we have $x=81$ and $power=66^2 \bmod 645 = 4356 \bmod 645 = 486$;
$i=5$: Because $a_5=0$, we have $x=81$ and $power=486^2 \bmod 645 = 236{,}196 \bmod 645 = 126$;
$i=6$: Because $a_6=0$, we have $x=81$ and $power=126^2 \bmod 645 = 15{,}876 \bmod 645 = 396$;
$i=7$: Because $a_7=1$, we find that $x=(81\cdot 396) \bmod 645 = 471$ and $power=396^2 \bmod 645 = 156{,}816 \bmod 645 = 81$;
$i=8$: Because $a_8=0$, we have $x=471$ and $power=81^2 \bmod 645 = 6561 \bmod 645 = 111$;
$i=9$: Because $a_9=1$, we find that $x=(471\cdot 111) \bmod 645 = 36$.

$\Theta((\log m)^2 \log n)$

Modular Exponentiation (模指数)

- **RSA Encryption application**
  - P,Q are big primes, N=PQ, L= (P-1)(Q-1),
    find E<L and GCD(E, L) = 1, find D<L and D*E ≡1 mod L
  - **Encryption:**
    - $(\text{plaintext})^E \bmod N = (\text{ciphertext})$ (E and N are public key)
  - **Decryption:**
    - $(\text{ciphertext})^D \bmod N = (\text{plaintext})$ (D is private key)

- **Exercise:**
  - Use Algorithm ***modularExp*** to find $123^{101} \bmod 101$.
  - $101=(1100101)_2$

Homework

- § 4.1
  - 5, 13(b,d,f,h)
- § 4.2
  - 26

4.3 Primes and GCD

Wenjing Li

wjli@bupt.edu.cn

School of Computer Science

Beijing University of Posts & Telecommunications

Prime Numbers

- **Definition:**
  - An integer $p>1$ is ***prime*** iff it is not the product of two integers greater than 1:
    $p>1 \wedge \neg\exists a,b \in \mathbf{N}\text{:}\ a>1,\ b>1,\ ab=p$.
  - The only positive factors of a prime p are 1 and p itself. Some primes: 2,3,5,7,11,13...
  - Non-prime integers greater than 1 are called ***composite***, because they can be *composed* by multiplying two integers greater than 1.

Prime Numbers

- **Notation:**
  - $a|b \Leftrightarrow$ "*a divides b*" $\Leftrightarrow \exists c \in \mathbf{Z}: b = ac$
  - "*p* is *prime*" $\Leftrightarrow$
    $p > 1 \wedge \neg \exists a \in \mathbf{N}: (1 < a < p \wedge a|p)$
- **Terms:**
  - *factor*, *divisor*, *multiple*, *composite*, *prime*.

Theorems about Prime

- **Theorem 1: Fundamental Theorem of Arithmetic**
  - **Prime Factorization (质因数分解):**
  - Every positive integer has a ***unique*** representation as the product of a non-decreasing series of ***zero or more primes***. (每个正整数都可唯一地表示为0个或多个非递减素数的乘积)
  - Some examples:
    - $1 = 1$ (product of empty series)
    - $2 = 2$ (product of series with one element 2)
    - $4 = 2\cdot 2$ (product of series 2,2)
    - $2000 = 2\cdot 2\cdot 2\cdot 2\cdot 5\cdot 5\cdot 5$; $2001 = 3\cdot 23\cdot 29$;
      $2002 = 2\cdot 7\cdot 11\cdot 13$; $2003 = 2003$ (no clear pattern!)

Later, we will see how to rigorously prove the Fundamental Theorem of Arithmetic, starting from scratch!

Theorems about Prime

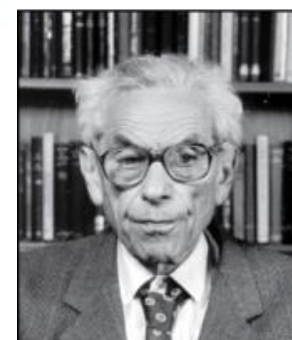
- **Theorem 2：Trial Division (试除定理)**
  - If n is a composite integer, then n has a prime divisor less than or equal to $\sqrt{n}$.
  - **E.g.** Show that 101 is prime.
    - Note that composite integers not exceeding 100 must have a prime factor not exceeding 10. Because the only primes less than 10 are 2, 3, 5, and 7.
    - The primes not exceeding 100 are not divisible by 2, 3, 5, or 7.

Theorems about Prime

- **Theorem 3: There are infinitely many primes.**
- **Proof**:
  - Assume finitely many primes: $p_1, p_2, \ldots\ldots, p_n$
  - Let $q = p_1 p_2 \cdots p_n + 1$
  - Either q is prime or by the ***fundamental theorem of arithmetic*** it is a product of primes.
    - But none of the primes p_j divides q since if $p_j \mid q$, then p_j divides $q - p_1 p_2 \cdots p_n = 1$.
    - Hence, there is a prime not on the list $p_1, p_2, \ldots\ldots, p_n$. It is either q, or if q is composite, it is a prime factor of q. This contradicts the assumption that $p_1, p_2, \ldots\ldots, p_n$ are all the primes.
  - Consequently, there are infinitely many primes.

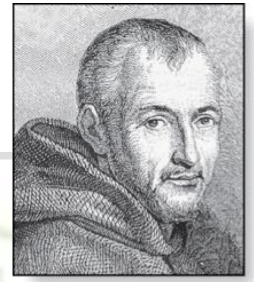
Euclid
(325 – 265 B.C.)

This proof was given by Euclid The Elements. The proof is considered to be one of the most beautiful in all mathematics. It is the first proof in The Book, inspired by the famous mathematician Paul Erdős' imagined collection of perfect proofs maintained by God.



Paul Erdős
(1913-1996) Hungary

Mersenne Primes



Marin Mersenne* *(1588-1648)，* *French

- **Definition:**
  - Prime numbers of the form $2^p - 1$, where p is prime, are called ***Mersenne primes,*** noted as M_p.
  - E.g. $2^2 - 1 = 3$, $2^3 - 1 = 7$, $2^5 - 1 = 37$, and $2^7 - 1 = 127$ are Mersenne primes.
  - $2^{11} - 1 = 2047$ is not a Mersenne prime since $2047 = 23 \cdot 89$.
  - The largest known prime numbers are Mersenne primes.
  - There is an efficient test for determining if $2^p - 1$ is prime.
  - On 07/12/2018, the 51st Mersenne primes was found. The largest is $2^{82,589,933} - 1$, which exceed 24 million decimal digits.
  - The *Great Internet Mersenne Prime Search* (*GIMPS:梅森素数大搜索*) is a distributed computing project to search for new Mersenne Primes. http://www.mersenne.org/

Theorems about Prime

- **Theorem 4: The prime number theorem**
  - **Distribution of Primes (素数分布)**
  - The ratio of the number of primes not exceeding x and x/lnx approaches 1 as x grows without bound.
  - The theorem tells us that the number of primes not exceeding x, can be approximated by $x/\ln x$. (不超过x的素数的个数近似为$x/ln\ x$)
  - The odds that a randomly selected positive integer less than n is prime are approximately $(n/\ln n)/n = 1/\ln n$. (随机选择一个小于n的正整数是素数的概率近似为$1/ln\ n$)

How many primes are less than a positive number x?

Greatest Common Divisor

- **Definition:**
  - The *greatest common divisor* gcd(a,b) (最大公约数/公因数) of integers a,b (not both 0) is the largest (most positive) integer d that is a divisor both of a and of b.

$$d = \gcd(a,b) = \max(d: d|a \wedge d|b) \Leftrightarrow$$

$$d|a \wedge d|b \wedge \forall e \in \mathbf{Z}, (e|a \wedge e|b) \rightarrow d \geq e$$

- **Example:**
  - gcd(24,36)=?
    Positive common divisors: 1,2,3,4,6,12.
    The largest one of these is 12.

Greatest Common Divisor

- **GCD Shortcut:**
  - If the *prime factorizations* are written as
    $a = p_1^{a_1} p_2^{a_2} \ldots p_n^{a_n}$ and $b = p_1^{b_1} p_2^{b_2} \ldots p_n^{b_n}$
  - then the GCD is given by:

$$\boxed{\gcd(a,b) = p_1^{\min(a_1,b_1)} p_2^{\min(a_2,b_2)} \ldots p_n^{\min(a_n,b_n)}.}$$

  - **Example** of using the shortcut:
    - a=84=2·2·3·7 $= 2^2 \cdot 3^1 \cdot 7^1$
    - b=96=2·2·2·2·2·3 $= 2^5 \cdot 3^1 \cdot 7^0$
    - gcd(84,96) $= 2^2 \cdot 3^1 \cdot 7^0 = 2 \cdot 2 \cdot 3 = 12$.

Relative Primality(互质)

- **Definition:**
  - Integers a and b are called *relatively prime* or *coprime(互质)* iff $gcd(a,b) = 1$.
    - Example: Neither 21 nor 10 is prime, but they are *coprime*. $21=3\cdot 7$ and $10=2\cdot 5$, so they have no common factors > 1, so their gcd = 1.
  - A *set* of integers $\{a_1,a_2,\ldots\}$ is *pairwise relatively prime (两两互质)* if all pairs (a_i, a_j), for $i \neq j$, are relatively prime.
    - Example: 10,17,21

Least Common Multiple

- **Definition:**
  - lcm(a,b) of positive integers a, b, is the smallest positive integer that is a multiple both of a and of b. *E.g.* lcm(6,10)=30

m = lcm(a,b) = min(m: $a|m \wedge b|m$) $\Leftrightarrow$
$a|m \wedge b|m \wedge \forall n \in \mathbf{Z}$: $(a|n \wedge b|n) \rightarrow (m \le n)$

- **LCM Shortcut:**
  - If the prime factorizations are written as
    $a = p_1^{a_1} p_2^{a_2} \ldots p_n^{a_n}$ and $b = p_1^{b_1} p_2^{b_2} \ldots p_n^{b_n}$
    then the LCM is given by

$12=2^2*3$, $15=3*5$
$LCM(12,15)=2^2*3*5=60$

$$\boxed{\text{lcm}(a,b) = p_1^{\max(a_1,b_1)} p_2^{\max(a_2,b_2)} \ldots p_n^{\max(a_n,b_n)}.}$$

Least Common Multiple

- **Theorem 5**
  - Let a and b be positive Integers. Then ab=gcd(a,b) · lcm(a,b)

$$a = p_1^{a_1} p_2^{a_2} \ldots p_n^{a_n} \qquad b = p_1^{b_1} p_2^{b_2} \ldots p_n^{b_n}$$

Euclid's Algorithm for GCD

- **Problem:**
  - Finding GCDs by comparing prime factorizations can be difficult when the prime factors are not known!
- **Euclid discovered:**
  - For all ints. a, b,
    $\gcd(a, b) = \gcd((a \bmod b), b)$.
  - Sort a,b so that $a>b$ (given $b>1$), and then
    $(a \bmod b) < b$, so problem is simplified.

Euclid of Alexandria, Greek 325-265 B.C.

Euclid's Algorithm for GCD

- **Examples:**
  - gcd(372,164) = gcd(372 mod 164, 164).

    372 mod 164 = $372-164\lfloor 372/164\rfloor = 372-164\cdot 2 = 372-328 = 44$.
  - gcd(164,44) = gcd(164 mod 44, 44).

    164 mod 44 = $164-44\lfloor 164/44\rfloor = 164-44\cdot 3 = 164-132 = 32$.
  - gcd(44,32)

    = gcd(32, 12) (32, 44 mod 32)

    = gcd(12,8) (12, 32 mod 12)

    = gcd(8,4) (8, 12 mod 8)

    = gcd(4,0) (4, 8 mod 4)

    = 4.

Euclid's Algorithm for GCD

- **Lemma 1:**
  - Let $a = bq + r$, where a, b, q, and r are integers.

    Then $\gcd(a,b) = \gcd(b,r)$.
- **Proof of Lemma 1:**
  - Suppose that d divides both a and b. Then d also divides $a - bq = r$ (by Theorem 1/ Corollary 1 of Section 4.1). Hence, any common divisor of a and b must also be a common divisor of b and r.

    a和b的公因子也是b和r的公因子
  - Suppose that d divides both b and r. Then d also divides $bq + r = a$. Hence, any common divisor of b and r must also be a common divisor of a and b.

    b和r的公因子也是a和b的公因子
  - Therefore, $\gcd(a,b) = \gcd(b,r)$.

Euclid's Algorithm for GCD

- **Proof of Euclid's Algorithm for GCD:**
  - Suppose that a and b are positive integers with $a \geq b$.
  - Let $r_0 = a$ and $r_1 = b$. Successive applications of the division algorithm yields:

$r_0 = r_1 q_1 + r_2$	$0 \leq r_2 < r_1$
$r_1 = r_2 q_2 + r_3$	$0 \leq r_3 < r_2$
……	
$r_{n-2} = r_{n-1} q_{n-1} + r_n$	$0 \leq r_n < r_{n-1}$
$r_{n-1} = r_n q_n$	

$gcd(r_0, r_1)$ is $gcd(a,b)$
$gcd(r_1, r_2)$
$gcd(r_2, r_3)$
……
$gcd(r_{n-2}, r_{n-1})$
$gcd(r_{n-1}, r_n)$
$gcd(r_n, 0)$

  - Eventually, a remainder of zero occurs in the sequence of terms: $a = r_0 > r_1 > r_2 > \cdots r_n \geq 0$.
  - By Lemma 1:$\gcd(a,b)=\gcd(r_0,r_1) = \gcd(r_1,r_2) = \cdots = \gcd(r_{n-1},r_n) = \gcd(r_n, 0) = r_n$.
  - Hence the GCD is the last nonzero remainder in the sequence of divisions.

Euclid's Algorithm for GCD

- **Pseudocode of Euclid's Algorithm for GCD**

procedure *gcd*(*a*, *b*: positive integers)
    while $b \neq 0$
    begin
        $r := a$ **mod** b;
        $a := b$;
        $b := r$;
    end
    return *a*

Fast! Number of while loop iterations turns out to be *O(log(min(a,b)))*.

GCDS AS LINEAR COMBINATIONS

- **Bézout's Theorem(贝祖定理):**
  - If a and b are positive integers, then there exist integers s and t such that $\gcd(a,b) = sa + tb$. (*proof in exercises of Section* 5.2)
- **Definition:**
  - If a and b are positive integers, then integers s and t such that $\gcd(a,b) = sa + tb$ are called ***Bézout coefficients*** (***贝祖系数***) of a and b. The equation $\gcd(a,b) = sa + tb$ is called ***Bézout's identity***.
  - By Bézout's Theorem, the gcd of integers a and b can be expressed in the form $sa + tb$ where s and t are integers. This is a ***linear combination*** with integer coefficients of a and b.
  - $\gcd(6,14) = (-2)\cdot 6 + 1\cdot 14$

Étienne Bézout
(1730-1783)
French

Finding gcds as Linear Combinations

- **Example 17**:
  - Express gcd(252,198) = 18 as a linear combination of 252 and 198.
- **Solution:**
  - First use the Euclidean algorithm to show gcd(252,198) = 18
    1. gcd(198,54) $252 = 1\cdot 198 + 54$
    2. gcd(54,36) $198 = 3\cdot 54 + 36$
    3. gcd(36,18) $54 = 1\cdot 36 + 18$
    4. gcd(18,0) $36 = 2\cdot 18$
  - Now working backwards, from above equations
    - $18 = 54 - 1\cdot 36$
    - $36 = 198 - 3\cdot 54$
    - $18 = 54 - 1\cdot(198 - 3\cdot 54) = 4\cdot 54 - 1\cdot 198$
    - $18 = 4\cdot(252 - 1\cdot 198) - 1\cdot 198 = 4\cdot 252 - 5\cdot 198$

*This method is a **two pass method**. It first uses the Euclidian algorithm to find the gcd and then works backwards to express the gcd as a linear combination of the original two integers.*

To solve the Diophantine Equation (求解线性丢番图方程)
To prove some theorems.

Consequences of Bézout's Theorem

- **Lemma 2**:
  - If a, b, and c are positive integers such that $\gcd(a, b) = 1$ and $a \mid bc$, then $a \mid c$.
- **Proof**:
  - Assume $\gcd(a, b) = 1$ and $a \mid bc$
  - Since $\gcd(a, b) = 1$, by Bézout's Theorem there are integers s and t such that $sa + tb = 1$.
  - Multiplying both sides of the equation by c, yields:

    $sac + tbc = c.$
  - From Theorem 1 of Section 4.1:

    $a \mid sac$ and $a \mid tbc$

    $a \mid (tbc+sac)$

    and We conclude $a \mid c$, since $sac + tbc = c$.

Consequences of Bézout's Theorem

- **Theorem 7:**
  - Let m be a positive integer and let a, b, and c be integers. If $ac \equiv bc \pmod{m}$ and $gcd(c, m) = 1$, then $a \equiv b \pmod{m}$.
- **Proof:**
  - Because $ac \equiv bc \pmod{m}$, $m \mid ac - bc \Leftrightarrow m \mid c(a-b)$.
  - By Lemma 2, because $gcd(c, m) = 1$, it follows that

  $m \mid (a - b)$.

  We conclude that $a \equiv b \pmod{m}$.

$14 \equiv 8 \pmod{6}$ but $7 \not\equiv 4 \pmod{6}$

$24 \equiv 4 \pmod{5}$ and $6 \equiv 1 \pmod{5}$

Consequences of Bézout's Theorem

- **Lemma 3**:
  - If p is prime and $p \mid a_1 a_2 \cdots a_n$, then $p \mid a_i$ for some i.
  - *proof uses mathematical induction; see Exercise* 64 *of Section* 5.1
- Lemma 3 is crucial in the proof of the uniqueness of prime factorizations.

To prove the uniqueness of Prime Factorization.

Uniqueness of Prime Factorization

- **Uniqueness theorem:**
  - A prime factorization of a positive integer where the primes are in *nondecreasing order* is ***unique***.
- **Proof**: (*by contradiction*)
  - Suppose that the positive integer n can be written as a product of primes in two distinct ways:

$$n = p_1 p_2 \cdots p_s \text{ and } n = q_1 q_2 \cdots q_t$$

  - Remove all common primes from the factorizations to get

$$p_{i_1} p_{i_2} \cdots p_{i_u} = q_{j_1} q_{j_2} \cdots q_{j_v}.$$

  - That, $p_{i1} \mid q_{j1} q_{j2} \cdots q_{jv}$
  - By Lemma 3, it follows that p_{i1} divides q_{jk} for some k, contradicting the assumption that p_{i1} and q_{jk} are distinct primes (why contradicting?).
  - Hence, there can be at most one factorization of n into primes in nondecreasing order.

Exercise

- Use the Euclidean algorithm to express gcd(1001, 100001) as a linear combination of 1001 and 100001.

Homework

- § 4.3
  - 16, 24(a,c,f), 30, 40(a,c,e,g,i)