DISCRETE MATHEMATICS FOR COMPUTER SCIENCE

Dr. Qi Wang

Department of Computer Science and Engineering
Office: Room903, Nanshan iPark A7 Building
Email: wangqi@sustc.edu.cn

Godfrey Harold Hardy

was a prominent British mathematician, known for his achievements in number theory and mathematical analysis.

b. 1877. d. 1947

Godfrey Harold Hardy

was a prominent British mathematician, known for his achievements in number theory and mathematical analysis.

In his 1940 autobiography *A Mathematician's Apology*, Hardy wrote

"The great modern achievements of applied mathematics have been in relativity and quantum mechanics, and these subjects are, at present, **almost as 'useless' as the theory of numbers**."

b. 1877. d. 1947

Godfrey Harold Hardy

was a prominent British mathematician, known for his achievements in number theory and mathematical analysis.

In his 1940 autobiography *A Mathematician's Apology*, Hardy wrote "The great modern achievements of applied mathematics have been in relativity and quantum mechanics, and these subjects are, at present, **almost as 'useless' as the theory of numbers**."

b. 1877. d. 1947

If he could see the world now, Hardy would be spinning in his grave.

Number Theory

- *Number theory* is a branch of mathematics that explores integers and their properties, is the basis of **cryptography**, **coding theory**, **computer security**, **e-commerce**, etc.

Number Theory

- *Number theory* is a branch of mathematics that explores integers and their properties, is the basis of **cryptography**, **coding theory**, **computer security**, **e-commerce**, etc.

- At one point, the largest employer of mathematicians in the United States, and probably the world, was the National Security Agency (NSA). The NSA is the largest spy agency in the US (bigger than CIA, Central Intelligence Agency), and has the responsibility for code design and breaking.

Division

- If a and b are integers with $a \neq 0$, we say that *a divides b* if there is an integer c such that $b = ac$, or equivalently b/a is an integer. In this case, we say that a is a *factor* or *divisor* of b, and b is a *multiple* of a. (We use the notations $a|b$, $a \not| b$)

Division

- If a and b are integers with $a \neq 0$, we say that a *divides* b if there is an integer c such that $b = ac$, or equivalently b/a is an integer. In this case, we say that a is a *factor* or *divisor* of b, and b is a *multiple* of a. (We use the notations $a|b$, $a \not| b$)

Example

- ◇ $4 \mid 24$
- ◇ $3 \nmid 7$

Divisibility

- **All integers divisible by** $d > 0$ can be enumerated as:

$$\ldots, -kd, \ldots, -2d, -d, 0, d, 2d, \ldots, kd, \ldots$$

Divisibility

- **All integers divisible by** $d > 0$ can be enumerated as:

$$\ldots, -kd, \ldots, -2d, -d, 0, d, 2d, \ldots, kd, \ldots$$

- **Question:** Let n and d be two positive integers. How many positive integers not exceeding n are divisible by d?

Divisibility

- **All integers divisible by** $d > 0$ can be enumerated as:

$$\ldots, -kd, \ldots, -2d, -d, 0, d, 2d, \ldots, kd, \ldots$$

- **Question:** Let n and d be two positive integers. How many positive integers not exceeding n are divisible by d?

  Answer: Count the number of integers such that $0 < kd \le n$. Therefore, there are $\lfloor n/d \rfloor$ such positive integers.

Divisibility

- **Properties**

Let a, b, c be integers. Then the following hold:

1. if $a|b$ and $a|c$, then $a|(b+c)$
2. if $a|b$ then $a|bc$ for all integers c
3. if $a|b$ and $b|c$, then $a|c$

Divisibility

Properties

Let a, b, c be integers. Then the following hold:

1. if $a|b$ and $a|c$, then $a|(b+c)$
2. if $a|b$ then $a|bc$ for all integers c
3. if $a|b$ and $b|c$, then $a|c$

Proof.

Divisibility

- **Corollary** If a, b, c are integers, where $a \neq 0$, such that $a|b$ and $a|c$, then $a|(mb + nc)$ whenever m and n are integers.

Divisibility

- **Corollary** If a, b, c are integers, where $a \neq 0$, such that $a|b$ and $a|c$, then $a|(mb + nc)$ whenever m and n are integers.

Proof. By part (ii) and part (i) of Properties.

The Division Algorithm

- If a is an integer and d a positive integer, then there are **unique** integers q and r, with $0 \leq r < d$, such that $a = dq + r$. In this case, d is called the *divisor*, a is called the *dividend*, q is called the *quotient*, and r is called the *remainder*.

The Division Algorithm

- If a is an integer and d a positive integer, then there are **unique** integers q and r, with $0 \leq r < d$, such that $a = dq + r$. In this case, d is called the *divisor*, a is called the *dividend*, q is called the *quotient*, and r is called the *remainder*.

In this case, we use the notations $q = a\ div\ d$ and $r = a \bmod d$.

Congruence Relation

- If a and b are integers and m is a positive integer, then a is *congruent to b modulo m if m divides $a - b$*, denoted by $a \equiv b \pmod{m}$. This is called *congruence* and m is its *modulus*.

Congruence Relation

- If a and b are integers and m is a positive integer, then a is *congruent to b modulo m if m divides $a - b$*, denoted by $a \equiv b \pmod{m}$. This is called *congruence* and m is its *modulus*.

Example

- ◇ $15 \equiv 3 \pmod{6}$
- ◇ $-1 \equiv 11 \pmod{6}$

More on Congruences

- Let m be a positive integer. The integers a and b are congruent modulo m if and only if there is an integer k such that $a = b + km$.

More on Congruences

- Let m be a positive integer. The integers a and b are congruent modulo m if and only if there is an integer k such that $a = b + km$.

Proof.

"only if" part

"if" part

$\pmod{m}$ and $\bmod m$ Notations

- $a \equiv b \pmod{m}$ and $a \bmod m = b$ are different.
  - $a \equiv b \pmod{m}$ is a relation on the set of integers
  - In $a \bmod m = b$, the notation mod denotes a function

$(\text{mod } m)$ and $\text{mod}\, m$ Notations

- $a \equiv b \pmod{m}$ and $a \bmod m = b$ are different.

  - $a \equiv b \pmod{m}$ is a relation on the set of integers
  - In $a \bmod m = b$, the notation mod denotes a function

- Let a and b be integers, and let m be a positive integer. Then $a \equiv b \bmod m$ if and only if $a \bmod m = b \bmod m$

$(\mathrm{mod}\ m)$ and $\mathrm{mod}\, m$ Notations

- $a \equiv b \pmod{m}$ and $a \bmod m = b$ are different.
  - $a \equiv b \pmod{m}$ is a relation on the set of integers
  - In $a \bmod m = b$, the notation mod denotes a function

- Let a and b be integers, and let m be a positive integer. Then $a \equiv b \bmod m$ if and only if $a \bmod m = b \bmod m$

  Proof.

Congruences of Sums and Products

- Let m be a positive integer. If $a \equiv b \pmod{m}$ and $c \equiv d \pmod{m}$, then $a + c \equiv b + d \pmod{m}$ and $ac \equiv bd \pmod{m}$.

Congruences of Sums and Products

- Let m be a positive integer. If $a \equiv b \pmod{m}$ and $c \equiv d \pmod{m}$, then $a + c \equiv b + d \pmod{m}$ and $ac \equiv bd \pmod{m}$.

Proof.

Algebraic Manipulation of Congruences

- If $a \equiv b \bmod m$, then
  - $c \cdot a \equiv c \cdot b \pmod{m}$?
  - $c + a \equiv c + b \pmod{m}$?
  - $a/c \equiv b/c \pmod{m}$?

Algebraic Manipulation of Congruences

- If $a \equiv b \bmod m$, then
  - $c \cdot a \equiv c \cdot b \pmod{m}$?
  - $c + a \equiv c + b \pmod{m}$?
  - $a/c \equiv b/c \pmod{m}$?

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

Computing the mod Function

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

$$(a+b) \bmod m = ((a \bmod m) + (b \bmod m)) \bmod m$$
$$ab \bmod m = ((a \bmod m)(b \bmod m)) \bmod m$$

Computing the mod Function

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

$$(a+b) \bmod m = ((a \bmod m) + (b \bmod m)) \bmod m$$
$$ab \bmod m = ((a \bmod m)(b \bmod m)) \bmod m$$

Proof.

Arithmetic Modulo m

- Let $\mathbf{Z}_m$ be the set of nonnegative integers less than m: $\{0, 1, \ldots, m-1\}$.

Arithmetic Modulo m

- Let $\mathbf{Z}_m$ be the set of nonnegative integers less than m: $\{0, 1, \ldots, m-1\}$.

$$+_m : \ a +_m b = (a+b) \bmod m$$

$$\cdot_m : \ a \cdot_m b = ab \bmod m$$

Arithmetic Modulo m

- Let $\mathbf{Z}_m$ be the set of nonnegative integers less than m: $\{0, 1, \ldots, m-1\}$.

$$+_m : \; a +_m b = (a+b) \bmod m$$

$$\cdot_m : \; a \cdot_m b = ab \bmod m$$

Example

- ◇ $7 +_{11} 9 = ?$
- ◇ $7 \cdot_{11} 9 = ?$

Arithmetic Modulo m

- **Closure**: if $a, b \in \mathbf{Z}_m$, then $a +_m b,\ a \cdot_m b \in \mathbf{Z}_m$

Arithmetic Modulo m

- **Closure**: if $a, b \in \mathbf{Z}_m$, then $a +_m b,\ a \cdot_m b \in \mathbf{Z}_m$
- **Associativity**: if $a, b, c \in \mathbf{Z}_m$, then $(a +_m b) +_m c = a +_m (b +_m c)$ and $(a \cdot_m b) \cdot_m c = a \cdot_m (b \cdot_m c)$

Arithmetic Modulo m

- **Closure**: if $a, b \in \mathbf{Z}_m$, then $a +_m b,\ a \cdot_m b \in \mathbf{Z}_m$
- **Associativity**: if $a, b, c \in \mathbf{Z}_m$, then $(a +_m b) +_m c = a +_m (b +_m c)$ and $(a \cdot_m b) \cdot_m c = a \cdot_m (b \cdot_m c)$
- **Commutativity**: if $a, b \in \mathbf{Z}_m$, then $a +_m b = b +_m a$

Arithmetic Modulo m

- **Closure**: if $a, b \in \mathbf{Z}_m$, then $a +_m b,\ a \cdot_m b \in \mathbf{Z}_m$
- **Associativity**: if $a, b, c \in \mathbf{Z}_m$, then
  $(a +_m b) +_m c = a +_m (b +_m c)$ and
  $(a \cdot_m b) \cdot_m c = a \cdot_m (b \cdot_m c)$
- **Commutativity**: if $a, b \in \mathbf{Z}_m$, then $a +_m b = b +_m a$
- **Identity elements**: $a +_m 0 = a$ and $a \cdot_m 1 = a$

Arithmetic Modulo m

- **Closure**: if $a, b \in \mathbf{Z}_m$, then $a +_m b,\ a \cdot_m b \in \mathbf{Z}_m$
- **Associativity**: if $a, b, c \in \mathbf{Z}_m$, then $(a +_m b) +_m c = a +_m (b +_m c)$ and $(a \cdot_m b) \cdot_m c = a \cdot_m (b \cdot_m c)$
- **Commutativity**: if $a, b \in \mathbf{Z}_m$, then $a +_m b = b +_m a$
- **Identity elements**: $a +_m 0 = a$ and $a \cdot_m 1 = a$
- **Additive inverses**: if $a \neq 0$ and $a \in \mathbf{Z}_m$, then $m - a$ is an additive inverse of a modulo m

Arithmetic Modulo m

- **Closure**: if $a, b \in \mathbf{Z}_m$, then $a +_m b, \; a \cdot_m b \in \mathbf{Z}_m$
- **Associativity**: if $a, b, c \in \mathbf{Z}_m$, then
  $(a +_m b) +_m c = a +_m (b +_m c)$ and
  $(a \cdot_m b) \cdot_m c = a \cdot_m (b \cdot_m c)$
- **Commutativity**: if $a, b \in \mathbf{Z}_m$, then $a +_m b = b +_m a$
- **Identity elements**: $a +_m 0 = a$ and $a \cdot_m 1 = a$
- **Additive inverses**: if $a \neq 0$ and $a \in \mathbf{Z}_m$, then $m - a$ is an additive inverse of a modulo m
- **Distributivity**: if $a, b, c \in \mathbf{Z}_m$, then
  $a \cdot_m (b +_m c) = (a \cdot_m b) +_m (a \cdot_m c)$ and
  $(a +_m b) \cdot_m c = (a \cdot_m c) +_m (b \cdot_m c)$

Representations of Integers

- We may use *decimal* (*base* 10) or *binary* or *octal* or *hexadecimal* or other notations to represent integers.

Representations of Integers

- We may use *decimal* (*base* 10) or *binary* or *octal* or *hexadecimal* or other notations to represent integers.

- Let $b > 1$ be an integer. Then if n is a positive integer, it can be expressed uniquely in the form $n = a_k b^k + a_{k-1} b^{k-1} + \cdots + a_1 b + a_0$, where k is nonnegative, a_i's are nonnegative integers less than b. The representation of n is called *the base-b expansion of n* and is denoted by $(a_k a_{k-1} \ldots a_1 a_0)_b$.

Base-b Expansions

- To get the decimal expansion is easy.

Base-b Expansions

- To get the decimal expansion is easy.

Example

- $(101011111)_2 = 2^8 + 2^6 + 2^4 + 2^3 + 2^2 + 2^1 + 2^0 = 351$
- $(7016)_8 = 7 \cdot 8^3 + 1 \cdot 8 + 6 = 3598$

Base-b Expansions

- To get the decimal expansion is easy.

Example

◇ $(101011111)_2 = 2^8 + 2^6 + 2^4 + 2^3 + 2^2 + 2^1 + 2^0 = 351$

◇ $(7016)_8 = 7 \cdot 8^3 + 1 \cdot 8 + 6 = 3598$

- Conversions between binary, octal, hexadecimal expansions are easy.

Base-b Expansions

- To get the decimal expansion is easy.

  Example

  ◇ $(101011111)_2 = 2^8 + 2^6 + 2^4 + 2^3 + 2^2 + 2^1 + 2^0 = 351$

  ◇ $(7016)_8 = 7 \cdot 8^3 + 1 \cdot 8 + 6 = 3598$

- Conversions between binary, octal, hexadecimal expansions are easy.

  Example

  ◇ $(101011111)_2 = (\underline{101}\overline{011}\underline{111}) = (537)_8$

  ◇ $(7016)_8 = (\underline{111}\overline{000}\underline{001}\overline{110})_2$
  $= (\underline{1110}\overline{0000}\underline{1110})_2 = (E0E)_{16}$

Base-b Expansions

$$
\begin{aligned}
n &= a_k b^k + a_{k-1} b^{k-1} + a_{k-2} b^{k-2} + \cdots + a_2 b^2 + a_1 b + a_0 \\
&= b(a_k b^{k-1} + a_{k-1} b^{k-2} + a_{k-2} b^{k-3} + \cdots + a_2 b + a_1) + {\color{red} a_0} \\
&= b(b(a_k b^{k-2} + a_{k-1} b^{k-3} + a_{k-2} b^{k-4} + \cdots + a_2) + {\color{red} a_1}) + {\color{blue} a_0} \\
&= \cdots
\end{aligned}
$$

Base-b Expansions

$$\begin{aligned} n &= a_k b^k + a_{k-1} b^{k-1} + a_{k-2} b^{k-2} + \cdots + a_2 b^2 + a_1 b + a_0 \\ &= b(a_k b^{k-1} + a_{k-1} b^{k-2} + a_{k-2} b^{k-3} + \cdots + a_2 b + a_1) + a_0 \\ &= b(b(a_k b^{k-2} + a_{k-1} b^{k-3} + a_{k-2} b^{k-4} + \cdots + a_2) + a_1) + a_0 \\ &= \cdots \end{aligned}$$

To construct the base-b expansion of an integer n,

- Divide n by b to obtain $n = bq_0 + a_0$, with $0 \leq a_0 < b$
- The remainder a_0 is the rightmost digit in the base-b expansion of n. Then divide q_0 by b to get $q_0 = bq_1 + a_1$ with $0 \leq a_1 < b$
- a_1 is the second digit from the right. Continue by successively dividing the quotients by b until **the quotient is 0**

Algorithm: Constructing Base-b Expansions

procedure *base b expansion*(n, b: positive integers with $b > 1$)
$q := n$
$k := 0$
while ($q \neq 0$)
    $a_k := q$ **mod** b
    $q := q$ **div** b
    $k := k + 1$
return$(a_{k-1}, \ldots, a_1, a_0)\{(a_{k-1} \ldots a_1 a_0)_b$ is base b expansion of $n\}$

Example

- $(12345)_{10} = (30071)_8$

Example

- $(12345)_{10} = (30071)_8$

$$12345 = 8 \cdot 1543 + 1$$

$$1543 = 8 \cdot 192 + 7$$

$$192 = 8 \cdot 24 + 0$$

$$24 = 8 \cdot 3 + 0$$

$$3 = 8 \cdot 0 + 3$$

Binary Addition of Integers

$a = (a_{n-1}a_{n-2}\ldots a_2a_1)$, $b = (b_{n-1}b_{n-2}\ldots b_2b_1)$

procedure *add*(a, b: positive integers)
{the binary expansions of a and b are $(a_{n-1},a_{n-2},\ldots,a_0)_2$ and $(b_{n-1},b_{n-2},\ldots,b_0)_2$, respectively}
$c := 0$
for $j := 0$ to $n-1$
    $d := \lfloor (a_j + b_j + c)/2 \rfloor$
    $s_j := a_j + b_j + c - 2d$
    $c := d$
$s_n := c$
return$(s_0,s_1,\ldots, s_n)${the binary expansion of the sum is $(s_n,s_{n-1},\ldots,s_0)_2$}

Binary Addition of Integers

$a = (a_{n-1}a_{n-2}\ldots a_2 a_1)$, $b = (b_{n-1}b_{n-2}\ldots b_2 b_1)$

procedure *add*(a, b: positive integers)
{the binary expansions of a and b are $(a_{n-1}, a_{n-2}, \ldots, a_0)_2$ and $(b_{n-1}, b_{n-2}, \ldots, b_0)_2$, respectively}
$c := 0$
for $j := 0$ to $n-1$
    $d := \lfloor (a_j + b_j + c)/2 \rfloor$
    $s_j := a_j + b_j + c - 2d$
    $c := d$
$s_n := c$
return$(s_0, s_1, \ldots, s_n)$ {the binary expansion of the sum is $(s_n, s_{n-1}, \ldots, s_0)_2$}

$O(n)$ bit additions

Algorithm: Binary Multiplication of Integers

$a = (a_{n-1}a_{n-2}\ldots a_2 a_1)_2,\ b = (b_{n-1}b_{n-2}\ldots b_2 b_1)_2$

$$\begin{aligned} ab &= a(b_0 2^0 + b_1 2^1 + \cdots + b_{n-1}2^{n-1}) \\ &= a(b_0 2^0) + a(b_1 2^1) + \cdots + a(b_{n-1}2^{n-1}) \end{aligned}$$

procedure *multiply*(a, b: positive integers)
{the binary expansions of a and b are $(a_{n-1},a_{n-2},\ldots,a_0)_2$ and $(b_{n-1},b_{n-2},\ldots,b_0)_2$, respectively}
for $j := 0$ to $n - 1$
  if $b_j = 1$ **then** $c_j = a$ shifted j places
  else $c_j := 0$
{$c_0, c_1, \ldots, c_{n-1}$ are the partial products}
$p := 0$
for $j := 0$ to $n - 1$
  $p := p + c_j$
return p {p is the value of ab}

Algorithm: Binary Multiplication of Integers

$a = (a_{n-1}a_{n-2}\ldots a_2a_1)_2,\ b = (b_{n-1}b_{n-2}\ldots b_2b_1)_2$

$$\begin{aligned} ab &= a(b_0 2^0 + b_1 2^1 + \cdots + b_{n-1}2^{n-1}) \\ &= a(b_0 2^0) + a(b_1 2^1) + \cdots + a(b_{n-1}2^{n-1}) \end{aligned}$$

procedure *multiply*(a, b: positive integers)
{the binary expansions of a and b are $(a_{n-1}, a_{n-2}, \ldots, a_0)_2$ and $(b_{n-1}, b_{n-2}, \ldots, b_0)_2$, respectively}
for $j := 0$ to $n - 1$
    if $b_j = 1$ **then** $c_j = a$ shifted j places
    else $c_j := 0$
{$c_0, c_1, \ldots, c_{n-1}$ are the partial products}
$p := 0$
for $j := 0$ to $n - 1$
  $p := p + c_j$
return p {p is the value of ab}

$O(n^2)$ shifts and $O(n^2)$ bit additions

Algorithm: Computing div and mod

procedure *division algorithm* (a: integer, d: positive integer)

$q := 0$

$r := |a|$

while $r \geq d$

$\quad r := r - d$

$\quad q := q + 1$

if $a < 0$ and $r > 0$ **then**

$\quad r := d - r$

$\quad q := -(q+1)$

return (q, r) {$q = a$ **div** d *is the quotient*, $r = a$ **mod** d *is the remainder* }

Algorithm: Computing div and mod

procedure *division algorithm* (a: integer, d: positive integer)
$q := 0$
$r := |a|$
while $r \geq d$
  $r := r - d$
  $q := q + 1$
if $a < 0$ and $r > 0$ **then**
  $r := d - r$
  $q := -(q+1)$
return (q, r) {*$q = a$ **div** d is the quotient, $r = a$ **mod** d is the remainder* }

$O(n^2)$ bit operations, $n = \max(\log a, \log d)$

Algorithm: Binary Modular Exponentiation

$$b^n = b^{a_{k-1}\cdot 2^{k-1}+\cdots+a_1\cdot 2+a_0} = b^{a_{k-1}\cdot 2^{k-1}}\cdots b^{a_1\cdot 2}\cdot b^{a_0}$$

Successively finds $b \bmod m$, $b^2 \bmod m$, $b^4 \bmod m$, $\ldots$, $b^{2^{k-1}} \bmod m$, and multiplies together the terms b^{2^j} where $a_j = 1$.

procedure *modular exponentiation*(b: integer, $n = (a_{k-1}a_{k-2}\ldots a_1a_0)_2$, m: positive integers)
$x := 1$
$power := b$ **mod** m
for $i := 0$ to $k-1$
  if $a_i = 1$ **then** $x := (x \cdot power)$ **mod** m
  $power := (power \cdot power)$ **mod** m
return x {x equals b^n **mod** m }

Algorithm: Binary Modular Exponentiation

$$b^n = b^{a_{k-1}\cdot 2^{k-1}+\cdots+a_1\cdot 2+a_0} = b^{a_{k-1}\cdot 2^{k-1}} \cdots b^{a_1\cdot 2}\cdot b^{a_0}$$

Successively finds $b \bmod m$, $b^2 \bmod m$, $b^4 \bmod m$, $\ldots$, $b^{2^{k-1}} \bmod m$, and multiplies together the terms b^{2^j} where $a_j = 1$.

procedure *modular exponentiation*(b: integer, $n = (a_{k-1}a_{k-2}\ldots a_1a_0)_2$, m: positive integers)
$x := 1$
$power := b$ **mod** m
for $i := 0$ to $k-1$
  if $a_i = 1$ **then** $x := (x \cdot power)$ **mod** m
  $power := (power \cdot power)$ **mod** m
return x {x equals b^n **mod** m }

$O((\log m)^2 \log n)$ bit operations

Primes

- A positive integer p that is greater than 1 and is divisible only by 1 and by itself is called a *prime*.

Primes

- A positive integer p that is greater than 1 and is divisible only by 1 and by itself is called a *prime*.

- A positive integer p that is greater than 1 and is not a prime is called a *composite*.

Primes

- A positive integer p that is greater than 1 and is divisible only by 1 and by itself is called a *prime*.

- A positive integer p that is greater than 1 and is not a prime is called a *composite*.

- **Fundamental Theorem of Arithmetic** Every integer greater than 1 can be written uniquely as a prime or as the product of two or more primes where the prime factors are written in order of nondecreasing size.

Primes and Composites

- How to determine whether a number is a prime or a composite?

Primes and Composites

- How to determine whether a number is a prime or a composite?

Approach 1: test if each number $x < n$ divides n.

Primes and Composites

- How to determine whether a number is a prime or a composite?

Approach 1: test if each number $x < n$ divides n.

Approach 2: test if each prime number $x < n$ divides n.

Primes and Composites

- How to determine whether a number is a prime or a composite?

Approach 1: test if each number $x < n$ divides n.

Approach 2: test if each prime number $x < n$ divides n.

Approach 3: test if each prime number $x < \sqrt{n}$ divides n.

Primes and Composites

- If n is composite, then n has a prime divisor less than or equal to $\sqrt{n}$.

Primes and Composites

- If n is composite, then n has a prime divisor less than or equal to $\sqrt{n}$.

Proof.

$\diamond$ if n is composite, then it has a positive integer factor a such that $1 < a < n$ by definition. This means that $n = ab$, where b is an integer greater than 1.

$\diamond$ assume that $a > \sqrt{n}$ and $b > \sqrt{n}$. Then $ab > n$, contradiction. So either $a \leq \sqrt{n}$ or $b \leq \sqrt{n}$.

$\diamond$ Thus, n has a divisor less than $\sqrt{n}$.

$\diamond$ By the Fundamental Theorem of Arithmetic, this divisor is either prime, or is a product of primes. In either case, n has a prime divisor less than $\sqrt{n}$.

Primes

- There are infinitely many primes.

Proof (by contradiction)

Greatest Common Divisor (GCD)

- Let a and b be integers, not both 0. The largest integer d such that $d|a$ and $d|b$ is called the *greatest common divisor* of a and b, denoted by $\gcd(a, b)$.

Greatest Common Divisor (GCD)

- Let a and b be integers, not both 0. The largest integer d such that $d|a$ and $d|b$ is called the *greatest common divisor* of a and b, denoted by $\gcd(a, b)$.

  The integers a and b are *relatively prime* if their greatest common divisor is 1.

Greatest Common Divisor (GCD)

- Let a and b be integers, not both 0. The largest integer d such that $d|a$ and $d|b$ is called the *greatest common divisor* of a and b, denoted by $\gcd(a, b)$.

  The integers a and b are *relatively prime* if their greatest common divisor is 1.

  A systematic way to find the gcd is **factorization**. Let $a = p_1^{a_1} p_2^{a_2} \cdots p_n^{a_n}$ and $b = p_1^{b_1} p_2^{b_2} \cdots p_n^{b_n}$. Then $\gcd(a, b) = p_1^{\min(a_1, b_1)} p_2^{\min(a_2, b_2)} \cdots p_n^{\min(a_k, b_k)}$

Least Common Multiple (LCM)

- Let a and b be integers. The *least common multiple* of a and b is the smallest positive integer that is divisible by both a and b, denoted by $\mathrm{lcm}(a, b)$.

Least Common Multiple (LCM)

- Let a and b be integers. The *least common multiple* of a and b is the smallest positive integer that is divisible by both a and b, denoted by $\mathrm{lcm}(a, b)$.

We can also use **factorization** to find the lcm. Let $a = p_1^{a_1} p_2^{a_2} \cdots p_n^{a_n}$ and $b = p_1^{b_1} p_2^{b_2} \cdots p_n^{b_n}$. Then $\gcd(a, b) = p_1^{\max(a_1, b_1)} p_2^{\max(a_2, b_2)} \cdots p_n^{\max(a_k, b_k)}$

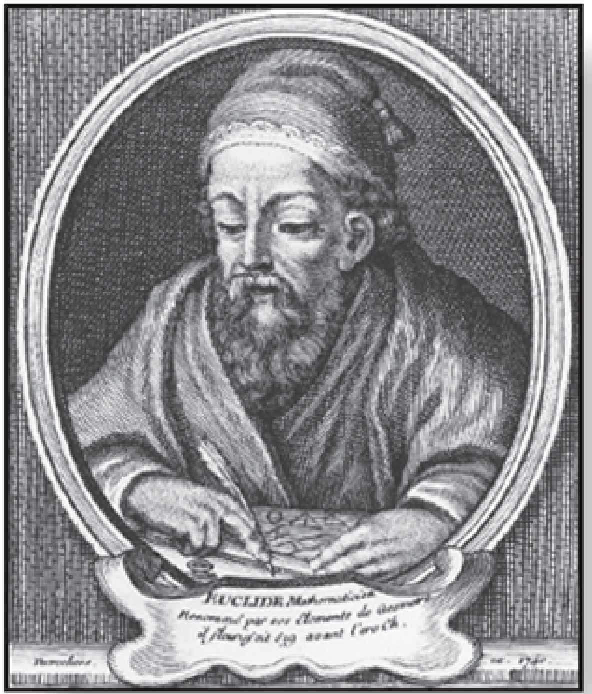
Euclidean Algorithm

- Factorization can be **cumbersome** and **time consuming** since we need to find all factors of the two integers.

Euclidean Algorithm

- Factorization can be **cumbersome** and **time consuming** since we need to find all factors of the two integers.

- Luckily, we have an efficient algorithm, called **Euclidean algorithm**. This algorithm has been known since ancient times and named after the ancient Greek mathmaticain Euclid.



Euclidean Algorithm

- For two integers 287 and 91, we want to find $\gcd(287, 91)$.

Euclidean Algorithm

- For two integers 287 and 91, we want to find $\gcd(287, 91)$.

Step 1: $287 = 91 \cdot 3 + 14$

Euclidean Algorithm

- For two integers 287 and 91, we want to find $\gcd(287, 91)$.

  Step 1: $287 = 91 \cdot 3 + 14$

  Step 2: $91 = 14 \cdot 6 + 7$

Euclidean Algorithm

- For two integers 287 and 91, we want to find $\gcd(287, 91)$.

  Step 1: $287 = 91 \cdot 3 + 14$

  Step 2: $91 = 14 \cdot 6 + 7$

  Step 3: $14 = 7 \cdot 2 + 0$

Euclidean Algorithm

- For two integers 287 and 91, we want to find $\gcd(287, 91)$.

Step 1: $287 = 91 \cdot 3 + 14$

Step 2: $91 = 14 \cdot 6 + 7$

Step 3: $14 = 7 \cdot 2 + 0$

$$\gcd(287, 91) = \gcd(91, 14) = \gcd(14, 7) = 7$$

Euclidean Algorithm

- The Euclidean algorithm in pseudocode

ALGORITHM 1 The Euclidean Algorithm.

procedure $gcd(a, b$: positive integers)
$x := a$
$y := b$
while $y \neq 0$
    $r := x$ **mod** y
    $x := y$
    $y := r$
return $x\{\gcd(a, b)$ is $x\}$

Euclidean Algorithm

- The Euclidean algorithm in pseudocode

ALGORITHM 1 The Euclidean Algorithm.

procedure $gcd(a, b$: positive integers)
$x := a$
$y := b$
while $y \neq 0$
    $r := x$ **mod** y
    $x := y$
    $y := r$
return $x\{\gcd(a, b)$ is $x\}$

The number of divisions required to find $\gcd(a, b)$ is $O(\log b)$, where $a \geq b$.

Correctness of Euclidean Algorithm

- **Lemma** Let $a = bq + r$, where a, b, q and r are integers. Then $\gcd(a, b) = \gcd(b, r)$.

Correctness of Euclidean Algorithm

- **Lemma** Let $a = bq + r$, where a, b, q and r are integers. Then $\gcd(a, b) = \gcd(b, r)$.

Proof.

◇ suppose that $d|a$ and $d|b$. Then d also divides $a - bq = r$. Hence, any common divisor of a and b must also be any common divisor of b and r.

◇ suppose that $d|b$ and $d|r$. Then d also divdes $bq + r = a$. Hence, any common divisor of a and b must also be a common divisor of b and r.

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

Correctness of Euclidean Algorithm

- Suppose that a and b are positive integers with $a \geq b$. Let $r_0 = a$ and $r_1 = b$.

Correctness of Euclidean Algorithm

- Suppose that a and b are positive integers with $a \geq b$. Let $r_0 = a$ and $r_1 = b$.

$$
\begin{aligned}
r_0 &= r_1 q_1 + r_2 && 0 \leq r_2 < r_1, \\
r_1 &= r_2 q_2 + r_3 && 0 \leq r_3 < r_2, \\
&\ \vdots \\
r_{n-2} &= r_{n-1} q_{n-1} + r_n && 0 \leq r_n < r_{n-1}, \\
r_{n-1} &= r_n q_n .
\end{aligned}
$$

Correctness of Euclidean Algorithm

- Suppose that a and b are positive integers with $a \geq b$. Let $r_0 = a$ and $r_1 = b$.

$$
\begin{aligned}
r_0 &= r_1 q_1 + r_2 && 0 \leq r_2 < r_1, \\
r_1 &= r_2 q_2 + r_3 && 0 \leq r_3 < r_2, \\
&\vdots \\
r_{n-2} &= r_{n-1} q_{n-1} + r_n && 0 \leq r_n < r_{n-1}, \\
r_{n-1} &= r_n q_n .
\end{aligned}
$$

$$\gcd(a, b) = \gcd(r_0, r_1) = \cdots = \gcd(r_{n-1}, r_n) = \gcd(r_n, 0) = r_n$$

Euclidean Algorithm

- **Example**: Find the GCD of 286 and 503.

- **Example**: Find the GCD of 286 and 503.

$$\gcd(503, 286)$$
$$= \gcd(286, 217)$$

$$503 = 1 \cdot 286 + 217$$

Euclidean Algorithm

- **Example**: Find the GCD of 286 and 503.

$$
\begin{aligned}
&\gcd(503, 286) \\
&= \gcd(286, 217) \\
&= \gcd(217, 69)
\end{aligned}
\qquad
\begin{aligned}
503 &= 1 \cdot 286 + 217 \\
286 &= 1 \cdot 217 + 69
\end{aligned}
$$

Euclidean Algorithm

- **Example**: Find the GCD of 286 and 503.

$$\begin{aligned} &\gcd(503, 286) \\ &= \gcd(286, 217) \\ &= \gcd(217, 69) \\ &= \gcd(69, 10) \end{aligned}$$

$$\begin{aligned} 503 &= 1 \cdot 286 + 217 \\ 286 &= 1 \cdot 217 + 69 \\ 217 &= 3 \cdot 69 + 10 \end{aligned}$$

Euclidean Algorithm

- **Example**: Find the GCD of 286 and 503.

$$\begin{aligned}
&\gcd(503, 286) \\
&= \gcd(286, 217) \\
&= \gcd(217, 69) \\
&= \gcd(69, 10) \\
&= \gcd(10, 9) \\
&= 1
\end{aligned}$$

$$\begin{aligned}
503 &= 1 \cdot 286 + 217 \\
286 &= 1 \cdot 217 + 69 \\
217 &= 3 \cdot 69 + 10 \\
69 &= 6 \cdot 10 + 9 \\
10 &= 1 \cdot 9 + 1 \\
9 &= 9 \cdot 1
\end{aligned}$$

GCD as Linear Combinations

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

GCD as Linear Combinations

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

  We may use *extended Euclidean algorithm* to find Bezout's identity.

GCD as Linear Combinations

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

  We may use *extended Euclidean algorithm* to find Bezout's identity.

  Example: Express 1 as the linear combination of 503 and 286.

GCD as Linear Combinations

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

We may use *extended Euclidean algorithm* to find Bezout's identity.

Example: Express 1 as the linear combination of 503 and 286.

$$503 = 1 \cdot 286 + 217$$
$$286 = 1 \cdot 217 + 69$$
$$217 = 3 \cdot 69 + 10$$
$$69 = 6 \cdot 10 + 9$$
$$10 = 1 \cdot 9 + 1$$

GCD as Linear Combinations

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

We may use *extended Euclidean algorithm* to find Bezout's identity.

Example: Express 1 as the linear combination of 503 and 286.

$$503 = 1 \cdot 286 + 217$$
$$286 = 1 \cdot 217 + 69$$
$$217 = 3 \cdot 69 + 10$$
$$69 = 6 \cdot 10 + 9$$
$$10 = 1 \cdot 9 + 1$$

$$\begin{aligned} 1 &= 10 - 1 \cdot 9 \\ &= 7 \cdot 10 - 1 \cdot 69 \\ &= 7 \cdot 217 - 22 \cdot 69 \\ &= 29 \cdot 217 - 22 \cdot 286 \\ &= 29 \cdot 503 - 51 \cdot 286 \end{aligned}$$

Corollaries of Bezout's Theorem

- If a, b, c are positive integers such that $\gcd(a, b) = 1$ and $a \mid bc$, then $a \mid c$.

Corollaries of Bezout's Theorem

- If a, b, c are positive integers such that $\gcd(a, b) = 1$ and $a|bc$, then $a|c$.

  Proof. Since $\gcd(a, b) = 1$, by Bezout's Theorem there exist s and t such that $1 = sa + tb$. This yields $c = sac + tbc$. Since $a|bc$, we have $a|tbc$, and then $a|(sac + tbc) = c$.

Corollaries of Bezout's Theorem

- If a, b, c are positive integers such that $\gcd(a, b) = 1$ and $a|bc$, then $a|c$.

  Proof. Since $\gcd(a, b) = 1$, by Bezout's Theorem there exist s and t such that $1 = sa + tb$. This yields $c = sac + tbc$. Since $a|bc$, we have $a|tbc$, and then $a|(sac + tbc) = c$.

- If p is prime and $p|a_1 a_2 \cdots a_n$, then $p|a_i$ for some i.

Corollaries of Bezout's Theorem

- If a, b, c are positive integers such that $\gcd(a, b) = 1$ and $a | bc$, then $a | c$.

  Proof. Since $\gcd(a, b) = 1$, by Bezout's Theorem there exist s and t such that $1 = sa + tb$. This yields $c = sac + tbc$. Since $a | bc$, we have $a | tbc$, and then $a | (sac + tbc) = c$.

- If p is prime and $p | a_1 a_2 \cdots a_n$, then $p | a_i$ for some i.

  Proof. by induction

Uniqueness of Prime Factorization

- We prove that a prime factorization of a positive integer where the primes are in nondecreasing order is unique.

Uniqueness of Prime Factorization

- We prove that 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:

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

Remove all common primes from the factorizations to get

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

It then follows that p_{i_1} divides q_{j_k} for some k, contradicting the assumption that p_{i_1} and q_{j_k} are distinct primes.

Next Lecture

- number theory II, cryptography, ...