Department of
Computer Science and Engineering
计算机科学与工程系

CS215 DISCRETE MATH

Dr. Qi Wang
Department of Computer Science and Engineering
Office: Room413, CoE South Tower
Email: wangqi@sustech.edu.cn

Application of Number Theory

- G. H. Hardy (1877 - 1947)

  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**."

Application of Number Theory

- G. H. Hardy (1877 - 1947)

  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**."

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 k such that $b = ak$, 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 $a|b$, $a \nmid b$)

Division

- If a and b are integers with $a \neq 0$, we say that a *divides* b if there is an integer k such that $b = ak$, 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 $a|b$, $a \nmid b$)

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 $0 < kd \leq n$. Therefore, there are $\lfloor n/d \rfloor$ such positive integers.

Divisibility

- **Properties**

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

  (i) if $a|b$ and $a|c$, then $a|(b+c)$
  (ii) if $a|b$ then $a|bc$ for all integers c
  (iii) if $a|b$ and $b|c$, then $a|c$

Divisibility

- **Properties**

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

(i) if $a|b$ and $a|c$, then $a|(b+c)$
(ii) if $a|b$ then $a|bc$ for all integers c
iii) 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.

- **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

$\pmod{m}$ 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

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

$\pmod{m}$ 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

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

Group

- A set G of elements, along with a binary operation $\star$, must satisfy the following four properties to be called a *group*.

Group

- A set G of elements, along with a binary operation $\star$, must satisfy the following four properties to be called a *group*.

  Closure: If $a, b \in G$, then $a \star b = c$ also belongs to G.

  Associativity: $(a \star b) \star c = a \star (b \star c)$

  Identity element: There is a unique element 1_e, such that for every $a \in G$, we have $a \star 1_e = a$.

  Inverse: For every $a \in G$, there exists an element, denoted by a^{-1}, such that $a \star a^{-1} = 1_e$.

Group

- A set G of elements, along with a binary operation $\star$, must satisfy the following four properties to be called a *group*.

  Closure: If $a, b \in G$, then $a \star b = c$ also belongs to G.

  Associativity: $(a \star b) \star c = a \star (b \star c)$

  Identity element: There is a unique element 1_e, such that for every $a \in G$, we have $a \star 1_e = a$.

  Inverse: For every $a \in G$, there exists an element, denoted by a^{-1}, such that $a \star a^{-1} = 1_e$.

  Example:
  $(\mathbb{Z}, +)$, $(\mathbb{Q}, +)$, $(\mathbb{R}, +)$, $(\mathbb{M}_{n \times n}, +)$?
  $(\mathbb{Z}^*, \times)$, $(\mathbb{Q}^*, \times)$, $(\mathbb{R}^*, \times)$, $(\mathbb{M}^*_{n \times n}, \cdot)$?

Permutation Group

- Let $s_n =< 1, 2, \ldots, n >$ denote a *sequence* of integers 1 through n. Denote by P_n the set of all *permutations* of the sequence s_n.

Permutation Group

- Let $s_n = < 1, 2, \ldots, n >$ denote a *sequence* of integers 1 through n. Denote by P_n the set of all *permutations* of the sequence s_n.

  For example, $s_3 = < 1, 2, 3 >$
  $P_3 = \{< 1, 2, 3 >, < 1, 3, 2 >, < 2, 1, 3 >, < 2, 3, 1 >, < 3, 1, 2 >, < 3, 2, 1 >\}$

Permutation Group

- Let $s_n =< 1, 2, \ldots, n >$ denote a *sequence* of integers 1 through n. Denote by P_n the set of all *permutations* of the sequence s_n.

  For example, $s_3 =< 1, 2, 3 >$
  $P_3 = \{< 1, 2, 3 >, < 1, 3, 2 >, < 2, 1, 3 >, < 2, 3, 1 >, < 3, 1, 2 >, < 3, 2, 1 >\}$

- Define a binary operation $\circ$ on the elements of P_n: for $\rho, \pi \in P_n$, $\pi \circ \rho$ denotes a *re-permutation* of the elements of ρ according to the elements of π.

Permutation Group

- Consider $s_3 =< 1, 2, 3 >$, and $P_3 = \{< p_1, p_2, p_3 > |p_1, p_2, p_3 \in s_3 \text{ with } p_1 \neq p_2 \neq p_3\}$.

Permutation Group

- Consider $s_3 =< 1, 2, 3 >$, and $P_3 = \{< p_1, p_2, p_3 > | p_1, p_2, p_3 \in s_3$ with $p_1 \neq p_2 \neq p_3\}$.
- $\pi =< 3, 2, 1 >$, $\rho =< 1, 3, 2 >$, what is $\pi \circ \rho$?

Permutation Group

- Consider $s_3 =< 1, 2, 3 >$, and $P_3 = \{< p_1, p_2, p_3 > | p_1, p_2, p_3 \in s_3 \text{ with } p_1 \neq p_2 \neq p_3\}$.
- $\pi =< 3, 2, 1 >$, $\rho =< 1, 3, 2 >$, what is $\pi \circ \rho$?

  $\pi \circ \rho =< 2, 3, 1 > \in P_3$

Permutation Group

- Consider $s_3 =< 1, 2, 3 >$, and
  $P_3 = \{< p_1, p_2, p_3 > | p_1, p_2, p_3 \in s_3 \text{ with } p_1 \neq p_2 \neq p_3\}$.
- $\pi =< 3, 2, 1 >$, $\rho =< 1, 3, 2 >$, what is $\pi \circ \rho$?

  $\pi \circ \rho =< 2, 3, 1 >\in P_3$
- We can verify the other three properties.

  $\rho_1 \circ (\rho_2 \circ \rho_3) = (\rho_1 \circ \rho_2) \circ \rho_3$

  $< 1, 2, 3 > \circ \rho = \rho \circ < 1, 2, 3 >= \rho$

  For each $\rho \in P_3$, there exists another unique $\pi \in P_3$ such that
  $\rho \circ \pi = \pi \circ \rho =< 1, 2, 3 >$

Permutation Group

- Consider $s_3 =< 1, 2, 3 >$, and
  $P_3 = \{< p_1, p_2, p_3 > | p_1, p_2, p_3 \in s_3$ with $p_1 \neq p_2 \neq p_3\}$.
- $\pi =< 3, 2, 1 >$, $\rho =< 1, 3, 2 >$, what is $\pi \circ \rho$?

  $\pi \circ \rho =< 2, 3, 1 >\in P_3$
- We can verify the other three properties.

  $\rho_1 \circ (\rho_2 \circ \rho_3) = (\rho_1 \circ \rho_2) \circ \rho_3$

  $< 1, 2, 3 > \circ\rho = \rho \circ < 1, 2, 3 >= \rho$

  For each $\rho \in P_3$, there exists another unique $\pi \in P_3$ such that

  $\rho \circ \pi = \pi \circ \rho =< 1, 2, 3 >$

  $(P_n, \circ)$ is called a *permutation group*.

Abelian Group

- If the operation on the set elements is *commutative*, the group is called an *abelian group*. $(a \star b = b \star a)$

Abelian Group

- If the operation on the set elements is *commutative*, the group is called an *abelian group*. $(a \star b = b \star a)$

- **Example**

  $(\mathbb{Z}, +)$, $(\mathbb{Q}, +)$, $(\mathbb{R}, +)$, $(\mathbb{M}_{n \times n}, +)$?

  $(GL(n), \cdot)$, $(P_n, \circ)$?

Abelian Group

- If the operation on the set elements is *commutative*, the group is called an *abelian group*. $(a \star b = b \star a)$

- **Example**

  $(\mathbb{Z}, +)$, $(\mathbb{Q}, +)$, $(\mathbb{R}, +)$, $(\mathbb{M}_{n \times n}, +)$?

  $(GL(n), \cdot)$, $(P_n, \circ)$?

- If the group operation is referred to as addition (*multiplication*), then the group also allows for *subtraction* (*division*).

  $$a - b = a + (-b)$$
  $$a/b = a \cdot b^{-1}$$

Ring

- If $(R, +)$ is an *abelian group*, we define one more operation (denoted as *multiplication* $\times$ for convenience) to have a *ring* $(R, +, \times)$ satisfying the following properties.

Ring

- If $(R, +)$ is an *abelian group*, we define one more operation (denoted as *multiplication* $\times$ for convenience) to have a *ring* $(R, +, \times)$ satisfying the following properties.

  Closure: R must be closed w.r.t. $\times$

  Associativity: $(a \times b) \times c = a \times (b \times c)$

  Distributivity: $a \times (b + c) = a \times b + a \times c$
  $(a + b) \times c = a \times c + b \times c$

Ring

- If $(R, +)$ is an *abelian group*, we define one more operation (denoted as *multiplication* $\times$ for convenience) to have a *ring* $(R, +, \times)$ satisfying the following properties.

  Closure: R must be closed w.r.t. $\times$

  Associativity: $(a \times b) \times c = a \times (b \times c)$

  Distributivity: $a \times (b + c) = a \times b + a \times c$
  $(a + b) \times c = a \times c + b \times c$

  Example:
  $(\mathbb{Z}, +, \times)$, $(\mathbb{Q}, +, \times)$, $(\mathbb{R}, +, \times)$, $(\mathbb{M}_{n\times n}, +, \cdot)$?

Commutative Ring, Integral Domain

- A *ring* is *commutative* if the multiplication operation is *commutative* for all elements in the ring. ($ab = ba$)

Commutative Ring, Integral Domain

- A *ring* is *commutative* if the multiplication operation is *commutative* for all elements in the ring. ($ab = ba$)

- An *integral domain* $(R, +, \times)$ is a *commutative ring* that satisfies the following two additional properties.

  Identity element for multiplication: $a1 = 1a = a$
  Nonzero product for any two nonzero elements:
    if $ab = 0$, then either a or b must be 0.

Commutative Ring, Integral Domain

- A *ring* is *commutative* if the multiplication operation is *commutative* for all elements in the ring. $(ab = ba)$

- An *integral domain* $(R, +, \times)$ is a *commutative ring* that satisfies the following two additional properties.

  Identity element for multiplication: $a1 = 1a = a$
  Nonzero product for any two nonzero elements:
  if $ab = 0$, then either a or b must be 0.

  Example:
  $(\mathbb{Z}, +, \times)$, $(\mathbb{Q}, +, \times)$, $(\mathbb{R}, +, \times)$?
  $(\mathbb{Z}_m, +, \times)$, $(\mathbb{M}_{n\times n}, +, \cdot)$?

Field

- A *field*, denoted by $(F, +, \times)$, is an *integral domain* whose elements satisfy the following additional property.

  Inverse for multiplication: For every $a \in F$, there exists an element b, denoted by a^{-1}, such that $ab = ba = 1$.

Field

- A *field*, denoted by $(F, +, \times)$, is an *integral domain* whose elements satisfy the following additional property.

  Inverse for multiplication: For every $a \in F$, there exists an element b, denoted by a^{-1}, such that $ab = ba = 1$.

- **Example**:
  $(\mathbb{Z}, +, \times)$, $(\mathbb{Q}, +, \times)$, $(\mathbb{R}, +, \times)$?
  $(\mathbb{Z}_p, +, \times)$?

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

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

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 $0 \le a_0 < b$
- 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 $0 \le a_1 < b$
- 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

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

$$
\begin{aligned}
12345 &= 8 \cdot 1543 + 1 \\
1543 &= 8 \cdot 192 + 7 \\
192 &= 8 \cdot 24 + 0 \\
24 &= 8 \cdot 3 + 0 \\
3 &= 8 \cdot 0 + 3
\end{aligned}
$$

Binary Addition of Integers

$a = (a_{n-1} a_{n-2} \ldots a_1 a_0),\ b = (b_{n-1} b_{n-2} \ldots b_1 b_0)$

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

$a = (a_{n-1}a_{n-2}\dots a_1a_0)$, $b = (b_{n-1}b_{n-2}\dots b_1b_0)$

procedure *add*(a, b: positive integers)
{the binary expansions of a and b are $(a_{n-1}, a_{n-2}, \dots, a_0)_2$ and $(b_{n-1}, b_{n-2}, \dots, b_0)_2$, 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$(s_0, s_1, \dots, s_n)$ {the binary expansion of the sum is $(s_n, s_{n-1}, \dots, s_0)_2$}

$O(n)$ bit additions

Algorithm: Binary Multiplication of Integers

$a = (a_{n-1}a_{n-2}\ldots a_1a_0)_2$, $b = (b_{n-1}b_{n-2}\ldots b_1b_0)_2$

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

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

$a = (a_{n-1}a_{n-2}\ldots a_1a_0)_2,\ b = (b_{n-1}b_{n-2}\ldots b_1b_0)_2$

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

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 $r \geq d$
    $r := r - d$
    $q := q + 1$
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* }

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* }

$O(q \log a)$ bit operations. But there exist more efficient algorithms with complextiy $O(n^2)$, where $n = \max(\log a, \log d)$

Algorithm: Computing div and mod (cont)

- **procedure** *division*2 $(a, d \in \mathbb{N},\ d \geq 1)$
  if $a < d$
      return $(q, r) = (0, a)$
  $(q, r) = division2(\lfloor a/2 \rfloor, d)$
  $q = 2q,\ r = 2r$
  if a is odd
      $r = r + 1$
  if $r \geq d$
      $r = r - d$
      $q = q + 1$
  return (q, r)

Algorithm: Computing div and mod (cont)

- **procedure** $division2\ (a, d \in \mathbb{N},\ d \geq 1)$
  if $a < d$
      return $(q, r) = (0, a)$
  $(q, r) = division2(\lfloor a/2 \rfloor, d)$
  $q = 2q,\ r = 2r$
  if a is odd
      $r = r + 1$
  if $r \geq d$
      $r = r - d$
      $q = q + 1$
  return (q, r)

  $O(\log q \log a)$ bit operations.

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

Next Lecture

- number theory, cryptography ...