CompSci 102
Discrete Math for Computer Science

TABLE 1 Hexadecimal, Octal, and Binary Representation of the Integers 0 through 15.

Decimal	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Hexadecimal	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Octal	0	1	2	3	4	5	6	7	10	11	12	13	14	15	16	17
Binary	0	1	10	11	100	101	110	111	1000	1001	1010	1011	1100	1101	1110	1111

February 16, 2012

Prof. Rodger

Chap. 4.1 - Division

Definition: If a and b are integers with $a \neq 0$, then *a divides b* if there exists an integer c such that $b = ac$.

- When a divides b we say that a is a *factor* or *divisor* of b and that b is a multiple of a.
- The notation $a \mid b$ denotes that a divides b.
- If $a \mid b$, then b/a is an integer.
- If a does not divide b, we write $a \nmid b$.

Example: Determine whether 3 | 7 and whether 3 | 12.

Properties of Divisibility

Theorem 1: Let a, b, and c be integers, where $a \neq 0$.

i. If $a \mid b$ and $a \mid c$, then $a \mid (b + c)$;
ii. If $a \mid b$, then $a \mid bc$ for all integers c;
iii. If $a \mid b$ and $b \mid c$, then $a \mid c$.

Proof: (i) Suppose $a \mid b$ and $a \mid c$

Corollary: If a, b, and c be integers, where $a \neq 0$, such that $a \mid b$ and $a \mid c$, then $a \mid mb + nc$ whenever m and n are integers.

Proof:

Properties of Divisibility

Theorem 1: Let a, b, and c be integers, where $a \neq 0$.

i. If $a \mid b$ and $a \mid c$, then $a \mid (b + c)$;
ii. If $a \mid b$, then $a \mid bc$ for all integers c;
iii. If $a \mid b$ and $b \mid c$, then $a \mid c$.

Proof: (i) Suppose $a \mid b$ and $a \mid c$

It follows that $\exists$ integers s and t with $b = as$ and $c = at$.

Hence, $b + c = as + at = a(s + t)$.

Hence, $a \mid (b + c)$

Corollary: If a, b, and c be integers, where $a \neq 0$, such that $a \mid b$ and $a \mid c$, then $a \mid mb + nc$ whenever m and n are integers.

Proof: a | b so a | mb, a | c so a | nc. By (ii)

by (i), a | (mb + nc)

Division Algorithm

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 $a = dq + r$

- d is called the *divisor*.
- a is called the *dividend*.
- q is called the *quotient*.
- r is called the *remainder*.

Definitions of Functions **div** and **mod**

$q = a$ **div** d
$r = a$ **mod** d

Examples:

- What are the quotient and remainder when 101 is divided by 11?
- What are the quotient and remainder when -11 is divided by 3?

Division Algorithm

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 $a = dq + r$

- d is called the *divisor*.
- a is called the *dividend*.
- q is called the *quotient*.
- r is called the *remainder*.

Definitions of Functions **div** and **mod**

$q = a$ **div** d
$r = a$ **mod** d

Examples:

- What are the quotient and remainder when 101 is divided by 11?
  Solution: The quotient when 101 is divided by 11 is 9 = 101 **div** 11, and the remainder is 2 = 101 **mod** 11.
- What are the quotient and remainder when -11 is divided by 3?
  Solution: The quotient when -11 is divided by 3 is $-4 = -11$ **div** 3, and the remainder is $1 = -11$ **mod** 3.

Congruence Relation

Definition: If a and b are integers and m is a positive integer, then a is *congruent* to b *modulo* m if m divides $a - b$.

- The notation $a \equiv b \pmod{m}$ says that a is congruent to b modulo m.
- We say that $a \equiv b \pmod{m}$ is a *congruence* and that m is its *modulus.*
- Two integers are congruent mod m if and only if they have the same remainder when divided by m.
- If a is not congruent to b modulo m, we write
  $a \not\equiv b \pmod{m}$

Example: Determine whether 19 is congruent to 3 modulo 4 and whether 26 and 16 are congruent modulo 6.

Solution:

Congruence Relation

Definition: If a and b are integers and m is a positive integer, then a is *congruent* to b *modulo* m if m divides $a - b$.

- The notation $a \equiv b \pmod{m}$ says that a is congruent to b modulo m.
- We say that $a \equiv b \pmod{m}$ is a *congruence* and that m is its *modulus.*
- Two integers are congruent mod m if and only if they have the same remainder when divided by m.
- If a is not congruent to b modulo m, we write
  $a \not\equiv b \pmod{m}$

Example: Determine whether 19 is congruent to 3 modulo 4 and whether 26 and 16 are congruent modulo 6.

Solution:

- $19 \equiv 3 \pmod{4}$ because 4 divides $19 - 3 = 16$.
- $26 \not\equiv 16 \pmod{6}$ since $26 - 16 = 10$ is not divisible by 6.

More on Congruences

Theorem 4: 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:

More on Congruences

Theorem 4: 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:

– If $a \equiv b \pmod m$, then

(by the definition of congruence) $m \mid a - b$.

Hence, there is an integer k such that $a - b = km$ and equivalently $a = b + km$.

– Conversely, if $\exists$ integer k such that $a = b + km$,

then $km = a - b$.

Hence, $m \mid a - b$ and $a \equiv b \pmod m$.

The Relationship between (mod m) and **mod** m Notations

- The use of "mod" in $a \equiv b \pmod m$ and a **mod** $m = b$ are different.
  - $a \equiv b \pmod m$ is a relation on the set of integers.
  - In a **mod** $m = b$, the notation **mod** denotes a function.
- The relationship between these notations is made clear in this theorem.
- **Theorem 3**: Let a and b be integers, and let m be a positive integer. Then $a \equiv b \pmod m$ if and only if a **mod** $m = b$ **mod** m. (*Proof in the exercises*)

Congruences of Sums and Products

Theorem 5: 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:

- Because $a \equiv b \pmod m$ and $c \equiv d \pmod m$, by Theorem 4 there are integers s and t with $b = a + sm$ and $d = c + tm$.
- Therefore,

Example: Because $7 \equiv 2 \pmod 5$ and $11 \equiv 1 \pmod 5$, it follows from Theorem 5 that ◄

Congruences of Sums and Products

Theorem 5: 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:

- Because $a \equiv b \pmod m$ and $c \equiv d \pmod m$, by Theorem 4 there are integers s and t with $b = a + sm$ and $d = c + tm$.
- Therefore,
  - $b + d = (a + sm) + (c + tm) = (a + c) + m(s + t)$ and
  - $b\,d = (a + sm)(c + tm) = ac + m(at + cs + stm)$.
- Hence, $a + c \equiv b + d \pmod m$ and $ac \equiv bd \pmod m$.

Example: Because $7 \equiv 2 \pmod 5$ and $11 \equiv 1 \pmod 5$, it follows from Theorem 5 that

$18 = 7 + 11 \equiv 2 + 1 = 3 \pmod 5$

$77 = 7\ 11 \equiv 2 * 1 = 2 \pmod 5$

Algebraic Manipulation of Congruences

- Multiplying both sides of a valid congruence by an integer preserves validity.

  If $a \equiv b \pmod m$ holds then $c \cdot a \equiv c \cdot b \pmod m$, where c is any integer, holds by Theorem 5 with $d = c$.
- Adding an integer to both sides of a valid congruence preserves validity.

  If $a \equiv b \pmod m$ holds then $c + a \equiv c + b \pmod m$, where c is any integer, holds by Theorem 5 with $d = c$.
- Dividing a congruence by an integer does not always produce a valid congruence.

Example: The congruence $14 \equiv 8 \pmod 6$ holds. But dividing both sides by 2 does not produce a valid congruence since

$14/2 = 7$ and $8/2 = 4$, but $7 \not\equiv 4 \pmod 6$.

See Section 4.3 for conditions when division is ok.

Computing the **mod** *m* Function of Products and Sums

- We use the following corollary to Theorem 5 to compute the remainder of the product or sum of two integers when divided by *m* from the remainders when each is divided by *m*.

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

$(a + b)$ **(mod** m**)** $=$ **((**a **mod** m**) + (**b **mod** m**)) mod** m

and

ab **mod** m $=$ **((**a **mod** m**) (**b **mod** m**)) mod** m.

(*proof in text*)

Arithmetic Modulo *m*

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

- The operation $+_m$ is defined as $a +_m b = (a + b)$ **mod** m. This is *addition modulo m*.
- The operation $\cdot_m$ is defined as $a \cdot_m b = (a \cdot b)$ **mod** m. This is *multiplication modulo m*.
- Using these operations is said to be doing *arithmetic modulo m*.

Example: Find $7 +_{11} 9$ and $7 \cdot_{11} 9$.

Solution: Using the definitions above:

Arithmetic Modulo *m*

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

- The operation $+_m$ is defined as $a +_m b = (a + b)$ **mod** m. This is *addition modulo m*.
- The operation $\cdot_m$ is defined as $a \cdot_m b = (a \cdot b)$ **mod** m. This is *multiplication modulo m*.
- Using these operations is said to be doing *arithmetic modulo m*.

Example: Find $7 +_{11} 9$ and $7 \cdot_{11} 9$.

Solution: Using the definitions above:

$7 +_{11} 9 = (7 + 9)$ **mod** $11 = 16$ **mod** $11 = 5$

$7 \cdot_{11} 9 = (7 \cdot 9)$ **mod** $11 = 63$ **mod** $11 = 8$

Arithmetic Modulo *m*

- The operations $+_m$ and $\cdot_m$ satisfy many of the same properties as ordinary addition and multiplication.
  - *Closure*: If a and b belong to $\mathbf{Z}_m$, then $a +_m b$ and $a \cdot_m b$ belong to $\mathbf{Z}_m$.
  - *Associativity*: If a, b, and c belong to $\mathbf{Z}_m$, 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 and b belong to $\mathbf{Z}_m$, then $a +_m b = b +_m a$ and $a \cdot_m b = b \cdot_m a$.
  - *Identity elements*: The elements 0 and 1 are identity elements for addition and multiplication modulo m, respectively.
    - If a belongs to $\mathbf{Z}_m$, then $a +_m 0 = a$ and $a \cdot_m 1 = a$.

→

Arithmetic Modulo *m*

  - *Additive inverses*: If $a \neq 0$ belongs to $\mathbf{Z}_m$, then $m - a$ is the additive inverse of a modulo m and 0 is its own additive inverse.
    - $a +_m (m - a) = 0$ and $0 +_m 0 = 0$
  - *Distributivity*: If a, b, and c belong to $\mathbf{Z}_m$, then
    - $a \cdot_m (b +_m c) = (a \cdot_m b) +_m (a \cdot_m c)$ and $(a +_m b) \cdot_m c = (a \cdot_m c) +_m (b \cdot_m c)$.
- Exercises 42-44 ask for proofs of these properties.
- Multiplicative inverses have not been included since they do not always exist. For example, there is no multiplicative inverse of 2 modulo 6.

Example

- What is the distributive property of multiplication over addition for $\boldsymbol{Z}_m$ with $m \geq 2$ an integer?

- **Proof:**

Example

- What is the distributive property of multiplication over addition for $\boldsymbol{Z}_m$ with $m \geq 2$ an integer?
  $a \cdot_m (b +_m c) = (a \cdot_m b) +_m (a \cdot_m c)$
  with $a, b, c, \in \boldsymbol{Z}_m$
- **Proof:** by def and corollary 2

Left side is a(b+c) **mod** m

Right side is ab + bc **mod** m

Thus a(b+c) mod m = ab + bc mod m

Since mult. distributive over add. for integers

Chap. 4.2 Representations of Integers

- In the modern world, we use *decimal*, or *base* 10, *notation* to represent integers. For example when we write 965, we mean $9 \cdot 10^2 + 6 \cdot 10^1 + 5 \cdot 10^0$.
- We can represent numbers using any base *b*, where *b* is a positive integer greater than 1.
- The bases *b* = 2 (*binary*), *b* = 8 (*octal*) , and *b*= 16 (*hexadecimal*) are important for computing and communications
- The ancient Mayans used base 20 and the ancient Babylonians used base 60.

Base *b* Representations

- We can use positive integer *b* greater than 1 as a base, because of this theorem:

  Theorem 1: Let *b* be a positive integer greater than 1. Then if *n* is a positive integer, it can be expressed uniquely in the form:

  $$n = a_k b^k + a_{k-1} b^{k-1} + \ldots + a_1 b + a_0$$

  where *k* is a nonnegative integer, $a_0, a_1, \ldots a_k$ are nonnegative integers less than *b*, and $a_k \neq 0$. The a_j, $j = 0, \ldots, k$ are called the base-*b* digits of the representation.

- The representation of n given in Theorem 1 is called the *base b expansion of n* and is denoted by $(a_k a_{k-1} \ldots a_1 a_0)_b$.
- We usually omit the subscript 10 for base 10 expansions.

Binary Expansions

Most computers represent integers and do arithmetic with binary (base 2) expansions of integers. In these expansions, the only digits used are 0 and 1.

Example: What is the decimal expansion of the integer that has $(1\ 0101\ 1111)_2$ as its binary expansion?

Solution:

Example: What is the decimal expansion of the integer that has $(11011)_2$ as its binary expansion?

Solution:

Binary Expansions

Most computers represent integers and do arithmetic with binary (base 2) expansions of integers. In these expansions, the only digits used are 0 and 1.

Example: What is the decimal expansion of the integer that has $(1\ 0101\ 1111)_2$ as its binary expansion?

Solution:

$(1\ 0101\ 1111)_2 = 1\cdot 2^8 + 0\cdot 2^7 + 1\cdot 2^6 + 0\cdot 2^5 + 1\cdot 2^4 + 1\cdot 2^3 + 1\cdot 2^2 + 1\cdot 2^1 + 1\cdot 2^0 = 351.$

Example: What is the decimal expansion of the integer that has $(11011)_2$ as its binary expansion?

Solution: $(11011)_2 = 1\cdot 2^4 + 1\cdot 2^3 + 0\cdot 2^2 + 1\cdot 2^1 + 1\cdot 2^0 = 27.$

Octal Expansions

The octal expansion (base 8) uses the digits $\{0,1,2,3,4,5,6,7\}$.

Example: What is the decimal expansion of the number with octal expansion $(7016)_8$?

Solution: $7\cdot 8^3 + 0\cdot 8^2 + 1\cdot 8^1 + 6\cdot 8^0 = 3598$

Example: What is the decimal expansion of the number with octal expansion $(111)_8$?

Solution: $1\cdot 8^2 + 1\cdot 8^1 + 1\cdot 8^0 = 64 + 8 + 1 = 73$

Octal Expansions

The octal expansion (base 8) uses the digits $\{0,1,2,3,4,5,6,7\}$.

Example: What is the decimal expansion of the number with octal expansion $(7016)_8$?

Solution: $7\cdot 8^3 + 0\cdot 8^2 + 1\cdot 8^1 + 6\cdot 8^0 = 3598$

Example: What is the decimal expansion of the number with octal expansion $(111)_8$?

Solution: $1\cdot 8^2 + 1\cdot 8^1 + 1\cdot 8^0 = 64 + 8 + 1 = 73$

Hexadecimal Expansions

The hexadecimal expansion needs 16 digits, but our decimal system provides only 10. So letters are used for the additional symbols. The hexadecimal system uses the digits $\{0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F\}$. The letters A through F represent the decimal numbers 10 through 15.

Example: What is the decimal expansion of the number with hexadecimal expansion $(2AE0B)_{16}$?

Example: What is the decimal expansion of the number with hexadecimal expansion $(E5)_{16}$?

Hexadecimal Expansions

The hexadecimal expansion needs 16 digits, but our decimal system provides only 10. So letters are used for the additional symbols. The hexadecimal system uses the digits {0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F}. The letters A through F represent the decimal numbers 10 through 15.

Example: What is the decimal expansion of the number with hexadecimal expansion $(2AE0B)_{16}$?

Solution:

$2\cdot16^4 + 10\cdot16^3 + 14\cdot16^2 + 0\cdot16^1 + 11\cdot16^0 = 175627$

Example: What is the decimal expansion of the number with hexadecimal expansion $(E5)_{16}$?

Solution: $1\cdot16^2 + 14\cdot16^1 + 5\cdot16^0 = 256 + 224 + 5 = 485$

Base Conversion

To construct the base b expansion of an integer n:

- Divide n by b to obtain a quotient and remainder.
  $n = bq_0 + a_0 \quad 0 \le a_0 < b$
- The remainder, a_0, is the rightmost digit in the base b expansion of n. Next, divide q_0 by b.
  $q_0 = bq_1 + a_1 \quad 0 \le a_1 < b$
- The remainder, a_1, is the second digit from the right in the base b expansion of n.
- Continue by successively dividing the quotients by b, obtaining the additional base b digits as the remainder. The process terminates when the quotient is 0.

continued →

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} ,…, a_1,a_0){($a_{k-1} \ldots a_1a_0)_b$ is base b expansion of n}

- q represents the quotient obtained by successive divisions by b, starting with $q = n$.
- The digits in the base b expansion are the remainders of the division given by q **mod** b.
- The algorithm terminates when $q = 0$ is reached.

Base Conversion

Example: Find the octal expansion of $(12345)_{10}$

Solution: Successively dividing by 8 gives:

Base Conversion

Example: Find the octal expansion of $(12345)_{10}$

Solution: Successively dividing by 8 gives:

- $12345 = 8 \cdot 1543 + 1$
- $1543 = 8 \cdot 192 + 7$
- $192 = 8 \cdot 24 + 0$
- $24 = 8 \cdot 3 + 0$
- $3 = 8 \cdot 0 + 3$

The remainders are the digits from right to left yielding $(30071)_8$.

Comparison of Hexadecimal, Octal, and Binary Representations

TABLE 1 Hexadecimal, Octal, and Binary Representation of the Integers 0 through 15.

Decimal	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Hexadecimal	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Octal	0	1	2	3	4	5	6	7	10	11	12	13	14	15	16	17
Binary	0	1	10	11	100	101	110	111	1000	1001	1010	1011	1100	1101	1110	1111

Initial 0s are not shown

Each octal digit corresponds to a block of 3 binary digits.
Each hexadecimal digit corresponds to a block of 4 binary digits.
So, conversion between binary, octal, and hexadecimal is easy.

Conversion Between Binary, Octal, and Hexadecimal Expansions

Example: Find the octal and hexadecimal expansions of $(11\ 1110\ 1011\ 1100)_2$.

Solution:

Conversion Between Binary, Octal, and Hexadecimal Expansions

Example: Find the octal and hexadecimal expansions of $(11\ 1110\ 1011\ 1100)_2$.

Solution:

- To convert to octal, we group the digits into blocks of three $(011\ 111\ 010\ 111\ 100)_2$, adding initial 0s as needed. The blocks from left to right correspond to the digits 3,7,2,7, and 4. Hence, the solution is $(37274)_8$.
- To convert to hexadecimal, we group the digits into blocks of four $(0011\ 1110\ 1011\ 1100)_2$, adding initial 0s as needed. The blocks from left to right correspond to the digits 3,E,B, and C. Hence, the solution is $(3EBC)_{16}$.

Binary Addition of Integers

- Algorithms for performing operations with integers using their binary expansions are important as computer chips work with binary numbers. Each digit is called a *bit*.

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 $(s_n,s_{n-1},\ldots,s_0)_2$}

- The number of additions of bits used by the algorithm to add two n-bit integers is $O(n)$.

Binary Multiplication of Integers

- Algorithm for computing the product of two n bit integers.

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}

- The number of additions of bits used by the algorithm to multiply two n-bit integers is $O(n^2)$.

Binary Modular Exponentiation

- In cryptography, it is important to be able to find b^n **mod** m efficiently, where b, n, and m are large integers.
- Use the binary expansion of n, $n = (a_{k-1},\ldots,a_1,a_0)_2$, to compute b^n .

  Note that:

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

- Therefore, to compute b^n, we need only compute the values of b, b^2, $(b^2)^2 = b^4$, $(b^4)^2 = b^8$, …, b^{2^k} and the multiply the terms b^{2^j} in this list, where $a_j = 1$.

Example: Compute 3^{11} using this method.

Solution

continued →

Binary Modular Exponentiation

- In cryptography, it is important to be able to find b^n **mod** m efficiently, where b, n, and m are large integers.
- Use the binary expansion of n, $n = (a_{k-1},\ldots,a_1,a_0)_2$, to compute b^n .

  Note that:

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

- Therefore, to compute b^n, we need only compute the values of b, b^2, $(b^2)^2 = b^4$, $(b^4)^2 = b^8$, …, b^{2^k} and the multiply the terms b^{2^j} in this list, where $a_j = 1$.

Example: Compute 3^{11} using this method.

Solution: Note that $11 = (1011)_2$ so that $3^{11} = 3^8\, 3^2\, 3^1 = ((3^2)^2)^2\, 3^2\, 3^1 = (9^2)^2 \cdot 9 \cdot 3 = (81)^2 \cdot 9 \cdot 3 = 6561 \cdot 9 \cdot 3 = 117{,}147$.

continued →

Binary Modular Exponentiation Algorithm

- The algorithm successively finds b **mod** m, b^2 **mod** m, b^4 **mod** m, …, $b^{2^{k-1}}$ **mod** 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 are used to find b^n **mod** m.