Arithmetic Modulo $m$

**Definitions:** Let $\mathbb{Z}_m$ 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 $m$

- **Additive inverses:** If $a = 0$ belongs to $\mathbb{Z}_m$, then $m - a$ is the additive inverse of a modulo $m$ and $0$ is its own additive inverse. 
- **Closure:** If $a$ and $b$ belong to $\mathbb{Z}_m$, then $a +_{m} b$ and $a \cdot_{m} b$ belong to $\mathbb{Z}_m$. 
- **Associativity:** If $a$, $b$, and $c$ belong to $\mathbb{Z}_m$, 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 $\mathbb{Z}_m$, 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 $\mathbb{Z}_m$, then $a +_{m} 0 = a$ and $a \cdot_{m} 1 = a$.

**Representations of Integers**

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 $a_k \neq 0$.

**Example for $b=10$:** 
$859 = 8 \cdot 10^2 + 5 \cdot 10^1 + 9 \cdot 10^0$

**Representations of Integers**

How can we construct the base $b$ expansion of an integer $n$?

First, divide $n$ by $b$ to obtain a quotient $q_0$ and remainder $a_0$, that is, 
\[ n = bq_0 + a_0, \text{ where } 0 \leq a_0 < b. \]

The remainder $a_0$ is the rightmost digit in the base $b$ expansion of $n$.

Next, divide $q_0$ by $b$ to obtain: 
\[ q_0 = bq_1 + a_1, \text{ where } 0 \leq a_1 < b. \]

$a_1$ is the second digit from the right in the base $b$ expansion of $n$. Continue this process until you obtain a quotient equal to zero.
Representations of Integers

Example:
What is the base 8 expansion of \((12345)_{10}\) ?

First, divide 12345 by 8:
\[
\begin{align*}
12345 &= 8 \cdot 1543 + 1 \\
1543 &= 8 \cdot 192 + 7 \\
192 &= 8 \cdot 24 + 0 \\
24 &= 8 \cdot 3 + 0 \\
3 &= 8 \cdot 0 + 3
\end{align*}
\]
The result is: \((12345)_{10} = (30071)_8\).

procedure base_b_expansion(n, b: positive integers)
q := n
k := 0
while q ≠ 0
begin
\[a_k := q \mod b\]
\[q := \lfloor q / b \rfloor\]
\[k := k + 1\]
end
{the base b expansion of n is \((a_{k-1} \ldots a_1 a_0)_b\)}

Addition of Integers

How do we (humans) add two integers?

Example:
\[
\begin{array}{c}
\text{carry} \\
111 \\
+ 4932 \\
\hline
12515
\end{array}
\]

Binary expansions:
\[
\begin{align*}
(111)_2 + (1010)_2 &= (10101)_2
\end{align*}
\]

Addition of Integers

Let a = \((a_n, a_{n-1}, \ldots, a_1, a_0)_2\), b = \((b_n, b_{n-1}, \ldots, b_1, b_0)_2\).
How can we algorithmically add these two binary numbers?
First, add their rightmost bits:
\[a_0 + b_0 = c_0 \cdot 2 + s_0,\]
where \(s_0\) is the rightmost bit in the binary expansion of \(a + b\), and \(c_0\) is the carry.

Then, add the next pair of bits and the carry:
\[a_1 + b_1 + c_0 = c_1 \cdot 2 + s_1,\]
where \(s_1\) is the next bit in the binary expansion of \(a + b\), and \(c_1\) is the carry.

Continue this process until you obtain \(c_{n-1}\).
The leading bit of the sum is \(s_n = c_{n-1}\).
The result is:
\[a + b = (s_n s_{n-1} \ldots s_1 s_0)_2\]

Example:
Add a = \((1110)_2\) and b = \((1011)_2\).
\[a_0 + b_0 = 0 + 1 = 0 \cdot 2 + 1,\] so that \(c_0 = 0\) and \(s_0 = 1\).
\[a_1 + b_1 + c_0 = 1 + 1 + 0 = 1 \cdot 2 + 0,\] so \(c_1 = 1\) and \(s_1 = 0\).
\[a_2 + b_2 + c_1 = 1 + 0 + 1 = 1 \cdot 2 + 0,\] so \(c_2 = 1\) and \(s_2 = 0\).
\[a_3 + b_3 + c_2 = 1 + 1 + 1 = 1 \cdot 2 + 1,\] so \(c_3 = 1\) and \(s_3 = 1\).
\[c_4 = c_3 = 1\].
Therefore, \(s = a + b = (11001)_2\).
Addition of Integers
procedure add(a, b: positive integers)
  // a_i, b_i are the bits of a and b.
  c := 0
  for j := 0 to n-1
  begin
    d := \lfloor (a_j + b_j + c)/2 \rfloor // gives the high bit of sum
    s_j := a_j + b_j + c – 2d  // gives the low bit of sum
    c := d
  end
  s_n := c
  {the binary expansion of the sum is (s_n s_{n-1}…s_1 s_0)_{2}}

Multiplication of Integers
procedure multiply(a, b: positive integers)
  // a_i, b_i are the bits of a and b.
  for j := 0 to n-1
  begin
    if b_j = 1 then c_j := a shifted left j places
    else c_j := 0  // c_j are the partial products
  end
  p := 0
  for i := 0 to n-1
  p := p + c_j
  {p is the value of the product as an integer. Note that we haven’t computed bits for p}

More Algorithms
Take a look at Algorithms 4 and 5 on pages 253, 254 and be sure you understand them. It’s important to be able to read the code and see what it says.
Algorithm 4 gives a way of doing the division algorithm using repeated subtractions instead of division.
Algorithm 5 gives a way of computing b^n using a binary representation of n

Section Summary
Integer Representations
– Base b Expansions
– Binary Expansions
– Octal Expansions
– Hexadecimal Expansions
Base Conversion Algorithm
Algorithms for Integer Operations

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\cdot10^2 + 6\cdot10^1 + 5\cdot10^0.
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_k, a_{k-1},…, a_0 are nonnegative integers less than b, and a_k \neq 0. The a_j are called the base-b digits of the representation.

(We will prove this using mathematical induction in Section 5.1.)
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}…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 \((10101111)_2\) as its binary expansion?

**Solution:**

\[
(10101111)_2 = 1 \cdot 2^8 + 0 \cdot 2^7 + 1 \cdot 2^6 + 0 \cdot 2^5 + 1 \cdot 2^4 + 1 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 351.
\]

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:**

\[
7 \cdot 8^3 + 0 \cdot 8^2 + 1 \cdot 8^1 + 6 \cdot 8^0 = 3598
\]

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:**

\[
2 \cdot 16^4 + 10 \cdot 16^3 + 14 \cdot 16^2 + 0 \cdot 16^1 + 11 \cdot 16^0 = 175627
\]

Base Conversion

To construct the base \(b\) expansion of an integer \(N\):

- Divide \(n\) by \(b\) to obtain a quotient and remainder.
- The remainder, \(a_0\), is the rightmost digit in the base \(b\) expansion of \(n\). Next, divide \(q_0\) by \(b\).
- 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.
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($a_{k-1}, ..., a_1, a_0$)

$q$ 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:

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

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

Comparison of Hexadecimal, Octal, and Binary Representations

<table>
<thead>
<tr>
<th>Octal</th>
<th>Hexadecimal</th>
<th>Binary</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0000</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0001</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>0010</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>0011</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>0100</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>0101</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>0110</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>0111</td>
</tr>
</tbody>
</table>

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:
- To convert to octal, we group the digits into blocks of three $(001 111 010 111 100)_8$, 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 $(0011 1110 1011 1100)_16$, 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

Procedure: add($a$, $b$: positive integers)

{the binary expansions of $a$ and $b$ are $(a_n, a_{n-1}, ..., a_0)_2$ and $(b_n, b_{n-1}, ..., b_0)_2$, respectively}

$c := 0$
for $j := 0$ to $n - 1$

d := [(a_j + b_j + c) \mod 2]
\ s_j := a_j + b_j + c - 2d
\ c := d
s_0 := c
return($s_n, s_{n-1}, ..., s_0$)

{the binary expansion of the sum is $(s_n, s_{n-1}, ..., s_0)_2$}
Binary Multiplication of Integers

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

\textbf{procedure multiply}(a, b: positive integers)
\begin{itemize}
  \item \text{the binary expansions of $a$ and $b$ are ($a_{n-1},\ldots,a_0$) and ($b_{n-1},\ldots,b_0$), respectively;}
  \item \text{for $j = 0$ to $n - 1$}
    \begin{itemize}
      \item \text{if $b_j = 1$ then $c_j = a$ shifted $j$ places}
      \item \text{else $c_j = 0$}
    \end{itemize}
  \item \text{$c_j$ is the value of the partial products}
  \item \text{return $p$}
\end{itemize}
\end{procedure}

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 $n$, $n = (a_{k-1},\ldots,a_1,a_0)$, to compute $b^n$.

\textbf{Note that:}

\[ b^n \equiv (b^2)^{n_2} \cdot b^{n_1} \cdot b^{n_0} \equiv (b^4)^{2^{n_3}} \cdot b^{n_2} \cdot b^{n_1} \cdot b^{n_0} \]

Therefore, to compute $b^n$, we need only compute the values of $b$, $b^2$, $b^4$, $b^8$, ..., and then multiply the terms in this list, where $a_j = 1$.

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

\textbf{Solution:} Note that $11 = (1011)_2$, so that:

\[ 3^{11} = 3^{10} \cdot 3^1 = (3^2)^5 \cdot 3^1 = (9^2)^5 \cdot 3 = (81)^5 \cdot 3 = 6561^5 \cdot 3 = 117,147. \]

continued →