FPT UNIVERSITY
DREAM OF INNOVATION

CHAPTER 4

Number Theory

Representations of Integers

Algorithms for Integer Operations

Modular Exponentiation

4.1 Divisibility and Modular Arithmetic

4.5 Applications of Congruences

4.3 Primes and Greatest Common Divisors

4.2 Integer Representations and Algorithms

4.2 Integer Representations and Algorithms

Representations of Integers

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

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

where k is a nonnegative integer, $a_0, a_1, \ldots, a_k$ are nonnegative integers less than b, and $a_k \neq 0$.

$965 = 9 \cdot 10^2 + 6 \cdot 10 + 5.$

BINARY EXPANSIONS

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

OCTAL AND HEXADECIMAL EXPANSIONS

$(2AE0B)_{16} = 2 \cdot 16^4 + 10 \cdot 16^3 + 14 \cdot 16^2 + 0 \cdot 16 + 11 = 175627.$

0, 1, 2, 3, 4, 5, 6, 7, 8, 9,
A, B, C, D, E, and F,
10 11 12 13 14 15.

$(245)_8$ represents $2 \cdot 8^2 + 4 \cdot 8 + 5 = 165.$

Representations of Integers

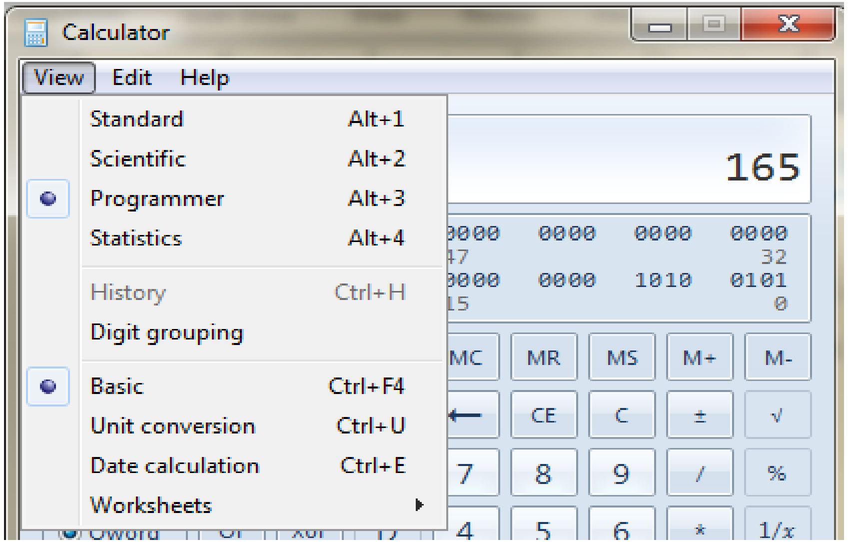
Constructing Base b Expansions.

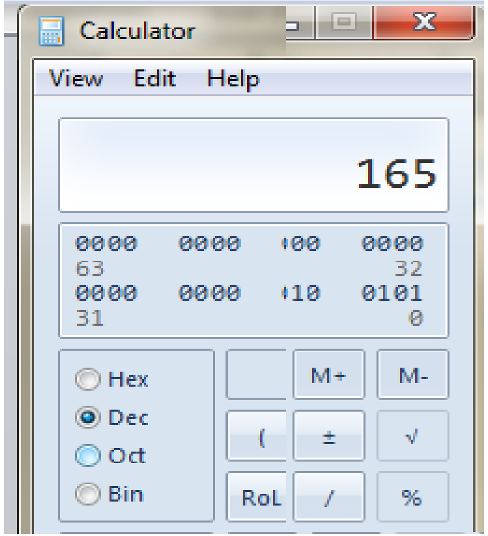
procedure *base b expansion*(n, b: positive integers with $b > 1$)
$q := n$
$k := 0$
while $q \neq 0$
  $a_k := q$ **mod** b
  $q := q$ **div** b
  $k := k + 1$
return $(a_{k-1}, \ldots, a_1, a_0)$ {the base b expansion of n}

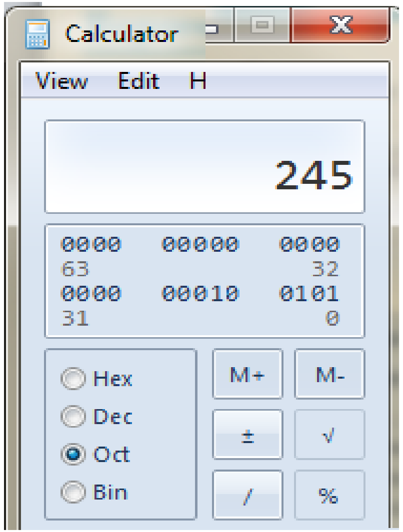
$$(1\,0101\,1111)_2 = 1 \cdot 2^8 + 0 \cdot 2^7 + 1 \cdot 2^6 + 0 \cdot 2^5 + 1 \cdot 2^4 + 1 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 351.$$

Representations of Integers

$$(245)_8 \text{ represents } 2 \cdot 8^2 + 4 \cdot 8 + 5 = 165.$$







$$(2AE0B)_{16} = 2 \cdot 16^4 + 10 \cdot 16^3 + 14 \cdot 16^2 + 0 \cdot 16 + 11 = 175627.$$

Algorithms for Integer Operations

ADDITION ALGORITHM

$$
\begin{array}{r}
1\ 1\ 1 \\
1\ 1\ 1\ 0 \\
+\ 1\ 0\ 1\ 1 \\
\hline
1\ 1\ 0\ 0\ 1
\end{array}
$$

MULTIPLICATION ALGORITHM

$$
\begin{array}{r}
1\ 1\ 0 \\
\times\ 1\ 0\ 1\ 1 \\
\hline
1\ 1\ 0 \\
1\ 1\ 0 \\
0\ 0\ 0 \\
1\ 1\ 0 \\
\hline
\end{array}
$$

Modular Exponentiation

3^{11} **mod** $m = 177{,}147$ **mod** m

$11 = (1011)_2$

b^n **mod** m $\qquad n = (a_{k-1} \ldots a_1 a_0)_2$

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

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

ALGORITHM Modular Exponentiation.

procedure *modular exponentiation*(b: integer,
  $n = (a_{k-1} a_{k-2} \ldots a_1 a_0)_2$, 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** $m\}$

Use Algorithm to find 3^{644} **mod** 645.

$$644 = (1010000100)_2$$

ALGORITHM Modular Exponentiation.

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

i	a_i	x	power
		1	3 = 3 % 645
0	0	–	9 = [(3 % 645) · (3 % 645)] % 645
1	0	–	81 = [(9 · 9) % 645]
2	1	81 = [(1 · 81) % 645]	111 = [(81 · 81) % 645]
3	0	–	66 = [(111 · 111) % 645]
4	0	–	486 = [(66 · 66) % 645]
5	0	–	126 = [(486 · 486) % 645]
6	0	–	396 = [(126 · 126) % 645]
7	1	471 = [(81 · 396) % 645]	81 = [(396 · 396) % 645]
8	0	–	111 = [(81 · 81) % 645]
9	1	**36** = [(471 · 111) % 645]	

Representations of Integers

CONVERSION BETWEEN BINARY, OCTAL, AND HEXADECIMAL EXPANSIONS

Conversion between binary and octal and between binary and hexadecimal expansions is extremely easy because each octal digit corresponds to a block of three binary digits and each hexadecimal digit corresponds to a block of four binary digits, with these correspondences shown in Table 1 without initial 0s shown.

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

4.2 Integer Representations and Algorithms

Representations of Integers

Find the octal and hexadecimal expansions of $(11\ 1110\ 1011\ 1100)_2$ and the binary expansions of $(765)_8$ and $(A8D)_{16}$.

Solution: To convert $(11\ 1110\ 1011\ 1100)_2$ into octal notation we group the binary digits into blocks of three, adding initial zeros at the start of the leftmost block if necessary. These blocks, from left to right, are 011, 111, 010, 111, and 100, corresponding to 3, 7, 2, 7, and 4, respectively. Consequently, $(11\ 1110\ 1011\ 1100)_2 = (37274)_8$. To convert $(11\ 1110\ 1011\ 1100)_2$ into hexadecimal notation we group the binary digits into blocks of four, adding initial zeros at the start of the leftmost block if necessary. These blocks, from left to right, are 0011, 1110, 1011, and 1100, corresponding to the hexadecimal digits 3, E, B, and C, respectively. Consequently, $(11\ 1110\ 1011\ 1100)_2 = (3EBC)_{16}$.

To convert $(765)_8$ into binary notation, we replace each octal digit by a block of three binary digits. These blocks are 111, 110, and 101. Hence, $(765)_8 = (1\ 1111\ 0101)_2$. To convert $(A8D)_{16}$ into binary notation, we replace each hexadecimal digit by a block of four binary digits. These blocks are 1010, 1000, and 1101. Hence, $(A8D)_{16} = (1010\ 1000\ 1101)_2$. ◀

Algorithms for Integer Operations

ALGORITHM Addition of Integers.

procedure *add*(a, b: positive integers)
{the binary expansions of a and b are $(a_{n-1}a_{n-2}\dots a_1a_0)_2$
  and $(b_{n-1}b_{n-2}\dots b_1b_0)_2$, 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 $(s_n s_{n-1} \dots s_0)_2$}

ALGORITHM Multiplication of Integers.

procedure *multiply*(a, b: positive integers)
{the binary expansions of a and b are $(a_{n-1}a_{n-2}\dots a_1a_0)_2$
  and $(b_{n-1}b_{n-2}\dots b_1b_0)_2$, respectively}
for $j := 0$ **to** $n - 1$
  if $b_j = 1$ **then** $c_j := a$ shifted j places
  else $c_j := 0$
{$c_0, c_1, \dots, c_{n-1}$ are the partial products}
$p := 0$
for $j := 0$ **to** $n - 1$
  $p := p + c_j$
return p {p is the value of ab}