Number Systems

Decimal System

ten symbols: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
called Arabic numerals

positional system = the value of a symbol depends on its position within the number

EX: 161
has value of 100
have value of 1

Sumerians were the first to use a positional system in 2100 BC

EX: Expand 161

$$161 = 1 \times 100 + 6 \times 10 + 1 \times 1$$
$$= 1 \times 10^2 + 6 \times 10^1 + 1 \times 10^0$$

powers of 10
∴ decimal system

Computers work with the **binary system**.

base (or **radix**) = the number of different symbols which can occur in each position in the number system

e.g. the decimal system is a base 10 system
(note there is no single symbol for the number 10)

Other number systems:

- Eskimos and North American Indians had a quinery system (base 5)
  ∴ had 5 symbols: 0, 1, 2, 3, 4
- Sumerians used a duodecimal system (base 12)
  we get 12 hour clock from them
  24 hr in a day
  360 days (or so they thought) in a year

12 symbols could be

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B

value of 10
value of 11

To Convert From **base r to base 10**:

expand out the number and simplify

EX: $15_{12} = ?_{10}$

one-five base 12

$12^1\ 12^0$

$$15_{12} = (1 \times 12^1 + 5 \times \cancel{12^0}^{\,1})_{10}$$
$$= (12 + 5)_{10}$$
$$= 17_{10}$$

EX: $A7B_{12} = ?_{10}$

$$A7B_{12} = (\overset{10}{A} \times \overset{144}{12^2} + 7 \times 12^1 + \overset{11}{B} \times \overset{1}{12^0})_{10}$$

$$= (1440 + 84 + 11)_{10}$$

$$= 1{,}535_{10}$$

Octal System (base 8)

8 symbols: 0, 1, 2, 3, 4, 5, 6, 7

EX: $100_8 = ?_{10}$

one-zero-zero octal

$$100_8 = (1 \times 8^2 + 0 \times 8^1 + 0 \times 8^0)_{10}$$

$$= (64 + 0 + 0)_{10} = 64_{10}$$

Binary Systems (base 2)

two symbols: 0, 1

computers have switches that can either be on or off

computers think of 1 as on and 0 as off

EX: Convert 342_5 to base 10.

$$342_5 = (3 \times 5^2 + 4 \times 5^1 + 2 \times 5^0)_{10}$$

$$= (3 \times 25 + 4 \times 5 + 2 \times 1)_{10}$$

$$= (75 + 20 + 2)_{10}$$

$$= 97_{10}$$

EX: Convert 1011_2 to base 10.

$$1011_2 = (1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0)_{10}$$

$$= (8 + 0 + 2 + 1)_{10}$$

$$= 11_{10}$$

systems w/ large bases have "compactness"

while systems with small bases have "expandedness" base 10

whole # part: 3 . fractional part: 273

$$3.273 = 3\times10^{0} + 2\times10^{-1} + 7\times10^{-2} + 3\times10^{-3}$$

decimal point b/c in base 10

We call it a radix point in other number systems.

EX: $111.111_2 = ?_{10}$

$$\overset{2^2\,2^1\,2^0}{111}.\overset{2^{-1}\,2^{-2}\,2^{-3}}{111}_2 = \left(1\times2^{2} + 1\times2^{1} + 1\times2^{0} + 1\times2^{-1} + 1\times2^{-2} + 1\times2^{-3}\right)_{10}$$

$$= \left(4 + 2 + 1 + \tfrac{1}{2} + \tfrac{1}{4} + \tfrac{1}{8}\right)_{10}$$

$$= (7 + .5 + .25 + .125)_{10}$$

$$= 7.875_{10}$$

Converting from base 10 to base r:

1. thinking/logical method

2. algorithm

EX: $53_{10} \rightarrow$ base 2

$\therefore 53_{10} = 110101_2$

logical thinking method

$$\begin{array}{r} 53 \\ -32 \\ \hline 21 \\ -16 \\ \hline 5 \\ -4 \\ \hline 1 \\ -1 \\ \hline 0 \end{array}$$

$\cancel{2^6}$	1	1	0	1	0	1
$\cancel{64}$	2^5	2^4	2^3	2^2	2^1	2^0
	32	16				

How many 32's in 53_{10}? 1

---

algorithm

$53_{10} \rightarrow$ base 2

base you want

whole # ÷ by 2	whole #	remainder
53/2	26	1
26/2	13	0
13/2	6	1
6/2	3	0
3/2	1	1
1/2	0	1

stop!

read # from remainders backwards

$\therefore 110101_2$

EX: $53_{10} \rightarrow$ base 5

logical

	2	0	3
5^3	5^2	5^1	5^0

$$\begin{array}{r} 53 \\ -50 \\ \hline 3 \end{array} \leftarrow 2\times25$$

$\therefore 53_{10} = 203_5$

$53_{10} \rightarrow$ base 5

algorithm

$53/5$	10	3
$10/5$	2	0
$2/5$	0	2

$\therefore 53_{10} = 203_5$

EX: $173_{10} \rightarrow$ base 12

logical

$\frac{1}{12^2} \ \frac{2}{12^1} \ \frac{5}{12^0}$

144

$\therefore 125_{12}$

$$\begin{array}{r} 173 \\ -144 \\ \hline 29 \\ 24 \\ \hline 5 \end{array}$$

algorithm

$173/12$	14	5
$14/12$	1	2
$1/12$	0	1

$\therefore 125_{12}$

Proof of why the algorithm works:

Let N_{10} be an arbitrary number in base 10

let r be the base we want

$$N_{10} = \left(d_n r^n + d_{n-1} r^{n-1} + \ldots + d_1 r + d_0 \right)_r$$

d_i = digit in i^{th} position

n = # of symbols/digits in N_{10}

First step in algorithm is divide by r:

$$\frac{N_{10}}{r} = \frac{\left(d_n r^n + d_{n-1} r^{n-1} + \ldots + d_1 r + d_0 \right)_r}{r}$$

$$= \left(\left(d_n r^{n-1} + d_{n-1} r^{n-2} + \ldots + d_1 \right) + \frac{d_0}{r} \right)_r$$

now take this # and ÷ by r

remainder AND rightmost digit/symbol of number we want

$$\frac{d_n r^{n-1} + d_{n-1} r^{n-2} + \ldots + d_1}{r}$$

$$= d_n r^{n-2} + d_{n-1} r^{n-3} + \ldots + \frac{d_1}{r}$$

remainder AND the next rightmost digit in # want

Keep repeating and the remainders will form the # we want (in reverse).