Digital data representation

Topic for the next few weeks

Representing Numbers

Representing Texts

Representing Audio

Representing Images and graphics

Representing Videos

Representing Numbers

1 + 1 = 10 ?!?

Examples provided by
Jones & Bartlett Learning

Representing Numbers

Positional representation

- Decimal (base 10)
- Binary (base 2)
- Octal (base 8)
- Hexadecimal (base 16)
- Conversion between different bases
  - Binary, hex, oct to decimal (recap)
  - Binary to/from hex and octal
  - Decimal to binary, hex, octal

Signed binary representation – 2's complement

- 2's complement (binary) to decimal
- Decimal to 2's complement (binary)

Binary representation of fractions

- Binary to decimal
- Decimal to binary

Representing Numbers

Positional representation

- Decimal (base 10)
- Binary (base 2)
- Octal (base 8)
- Hexadecimal (base 16)
- Conversion between different bases
  - Binary, hex, oct to decimal (recap)
  - Binary to/from hex and octal
  - Decimal to binary, hex, octal

Signed binary representation – 2's complement

- 2's complement (binary) to decimal
- Decimal to 2's complement (binary)

Binary representation of fractions

- Binary to decimal
- Decimal to binary

Numbers

Natural numbers

Zero and any number obtained by repeatedly adding one to it.

Examples: 100, 0, 45645, 32

Negative whole numbers

Values less than 0, indicated with a “–” sign before the left-most digit

Examples: −24, −1, −45645, −32

> Side note: Hyphen -, Minus −, N-dash –, and M-dash — are four different typographical symbols; use them correctly in your writing
> See https://en.wikipedia.org/wiki/Dash for details

Numbers

Integers

Values that are either a natural number, or a negative whole number

Examples: 249, 0, −45645, −32

Rational Numbers

Values that are integers, or the quotient of two integers

Examples: −249, −1, 0, 3/7, −2/5

NOTE: π, e, $\sqrt{2}$, $\sqrt{5}$… are *irrational* numbers

Representing Numbers

- How many ways can we represent the numeric value “five”?

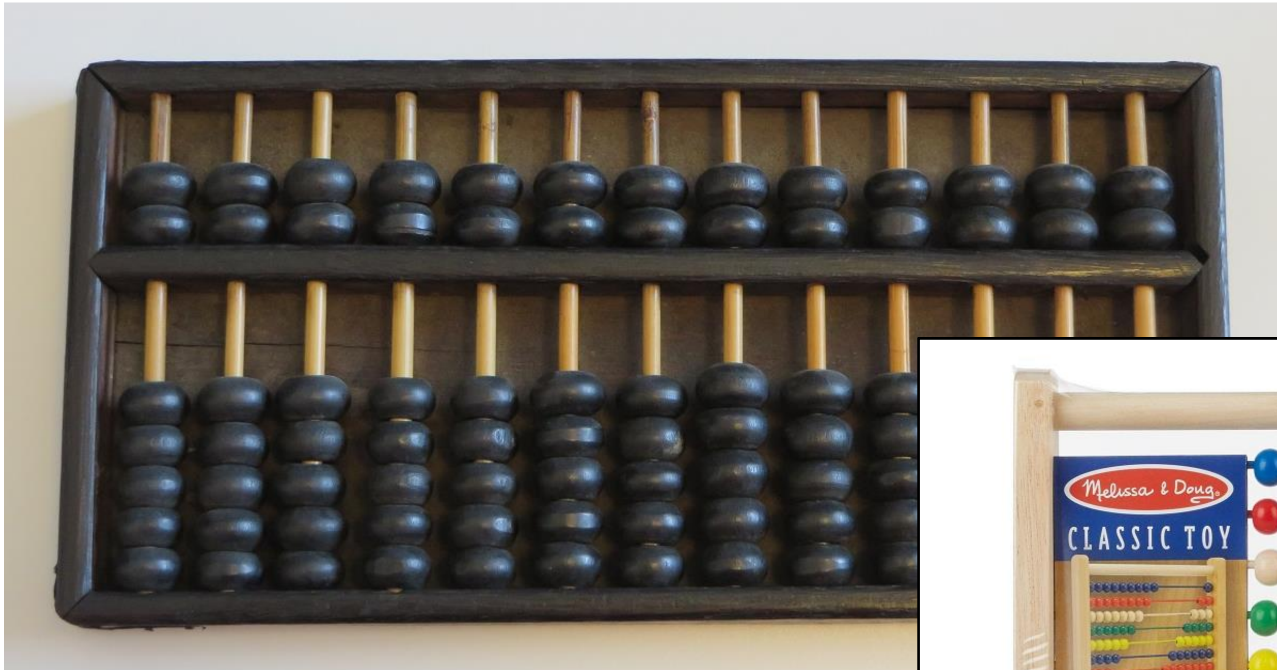
5

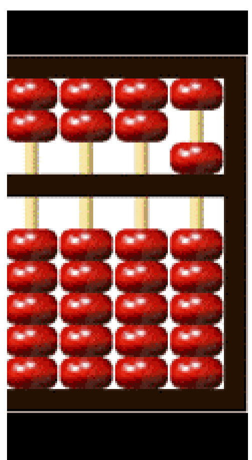
五

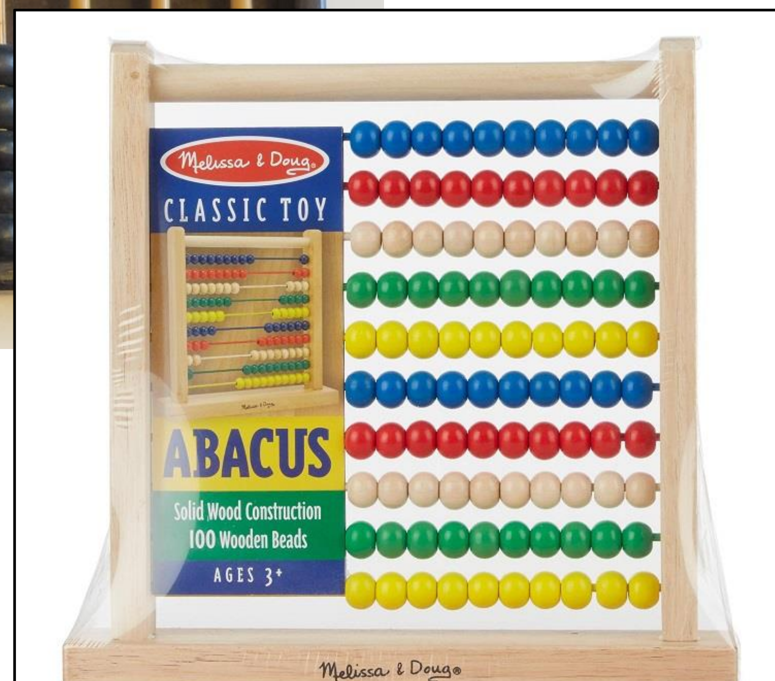
V

Tally marks used in most of Europe, Zimbabwe, Australia, New Zealand and North America.

Abacus







amazon.com

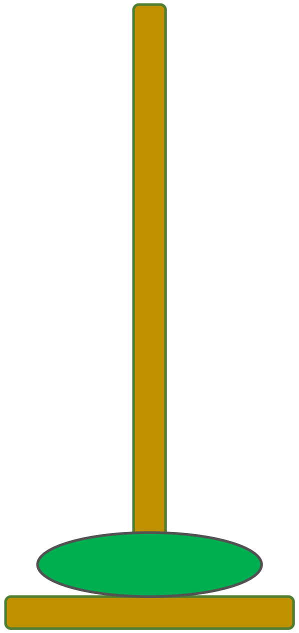
Positional Notation

428$ four hundreds and twenty-eight
248$ two hundreds and forty-eight

1820$ One thousand, eight hundreds and twenty
2108$ two thousand, one hundreds and eight

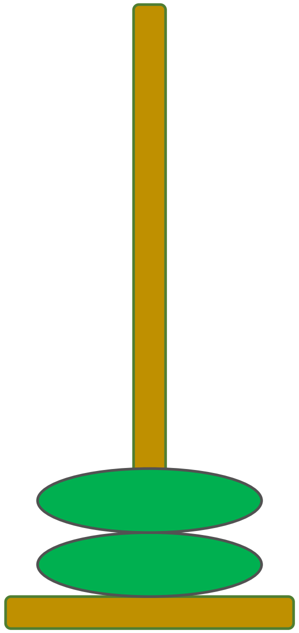
Decimal, how many letters

Positional Notation



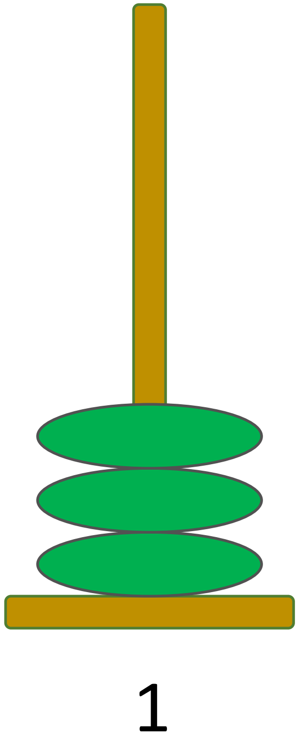
1

Positional Notation

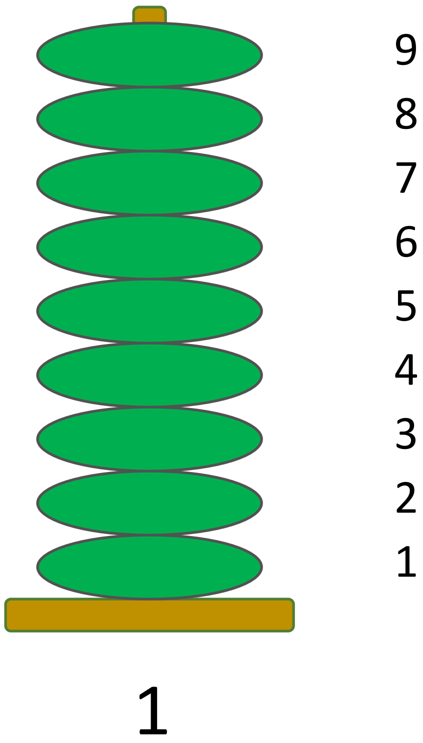


1

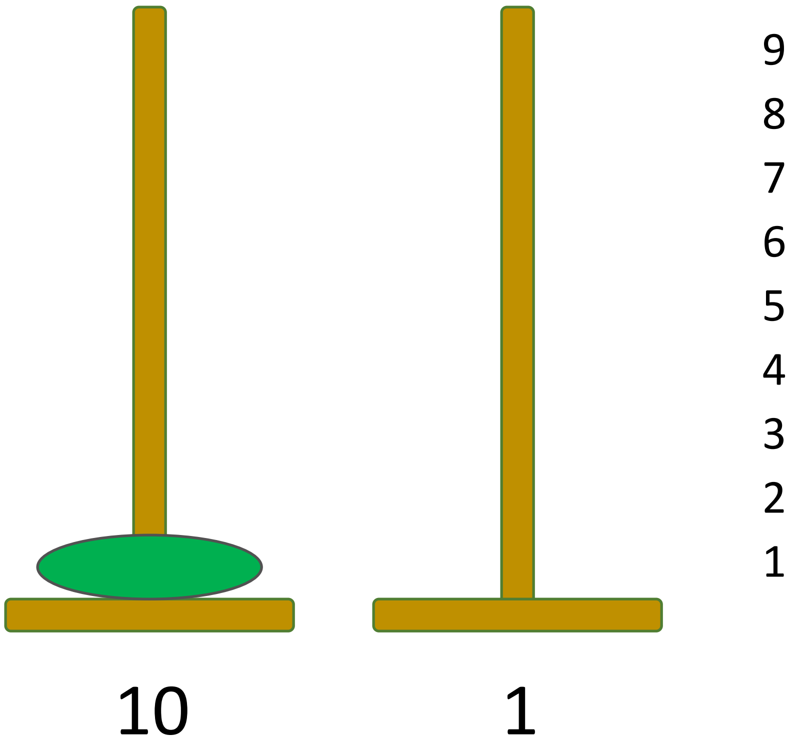
Positional Notation



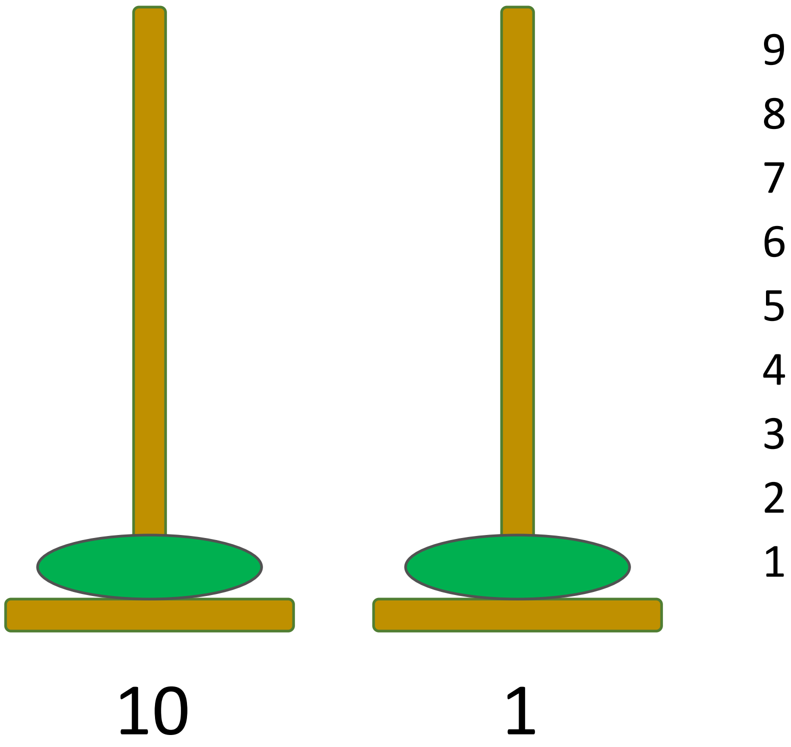
Positional Notation



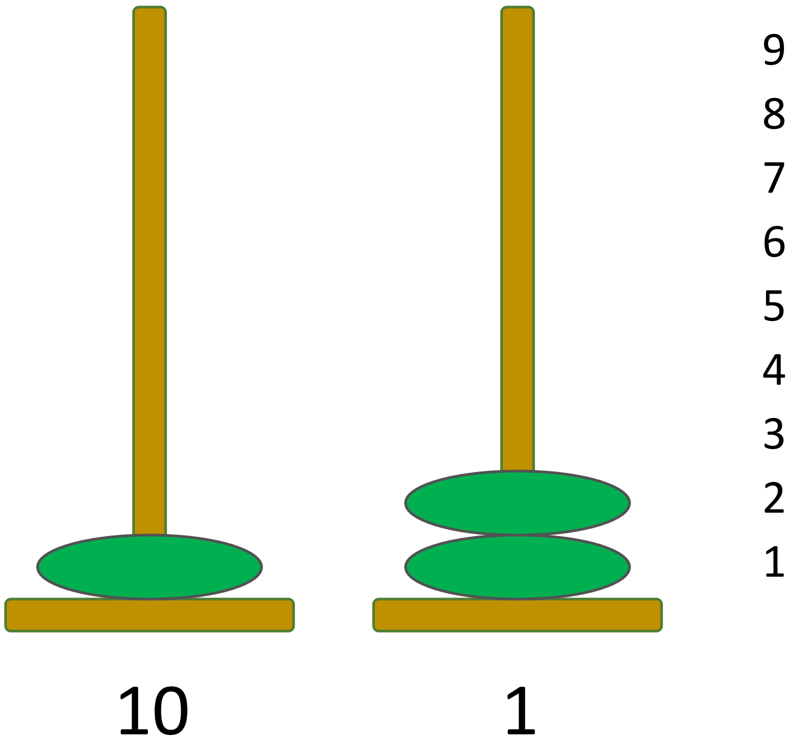
Positional Notation



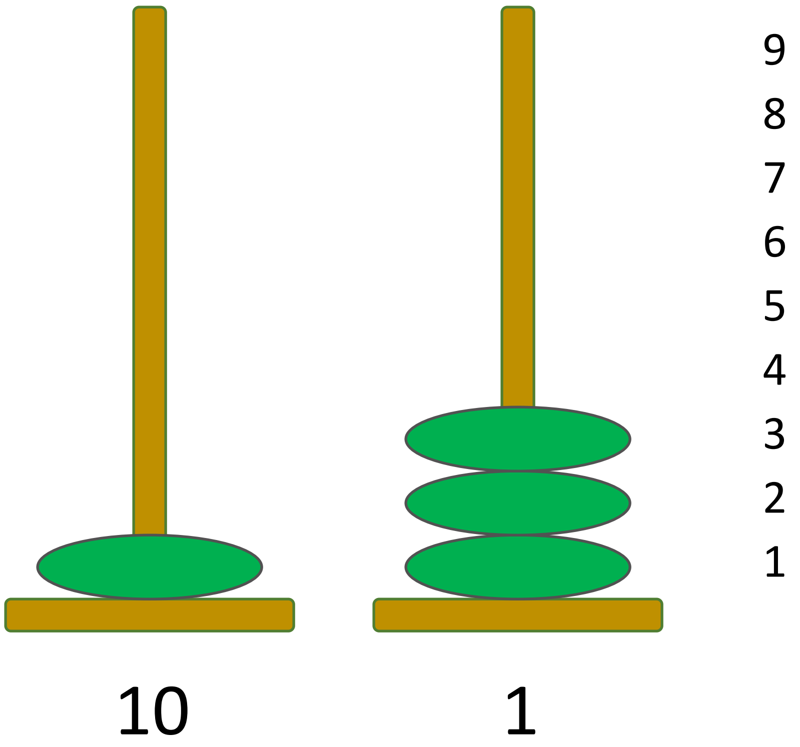
Positional Notation



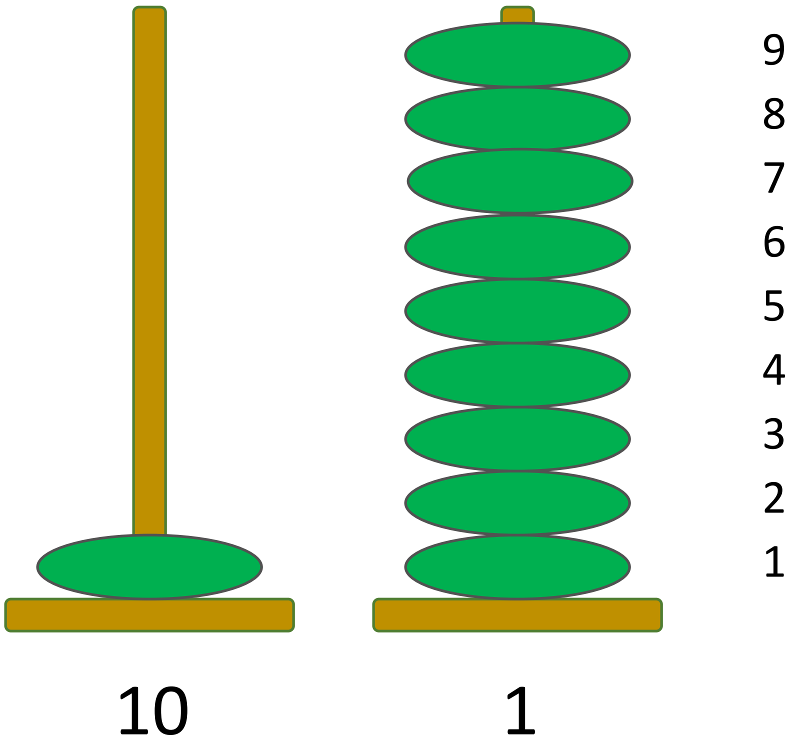
Positional Notation

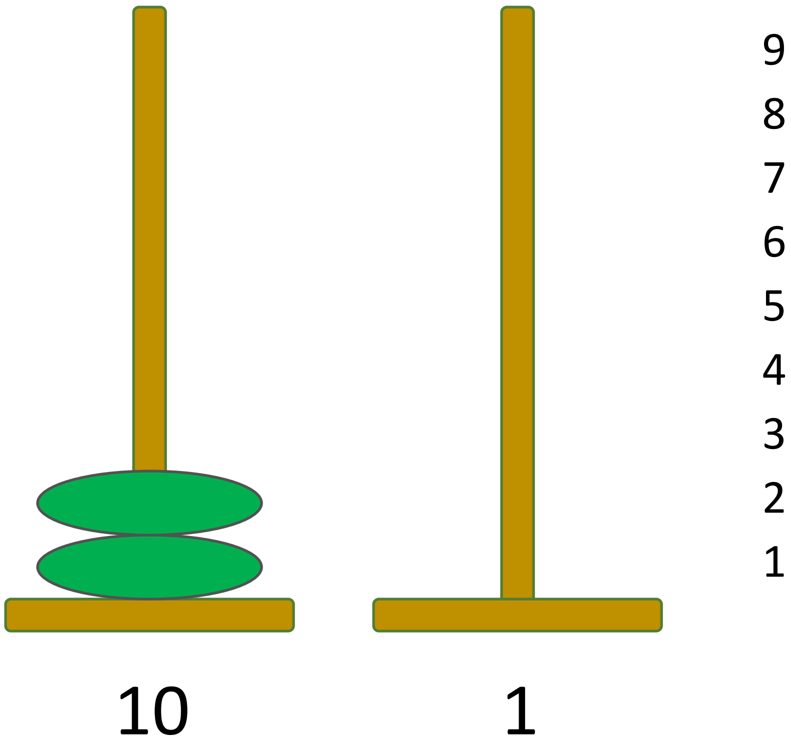


Positional Notation

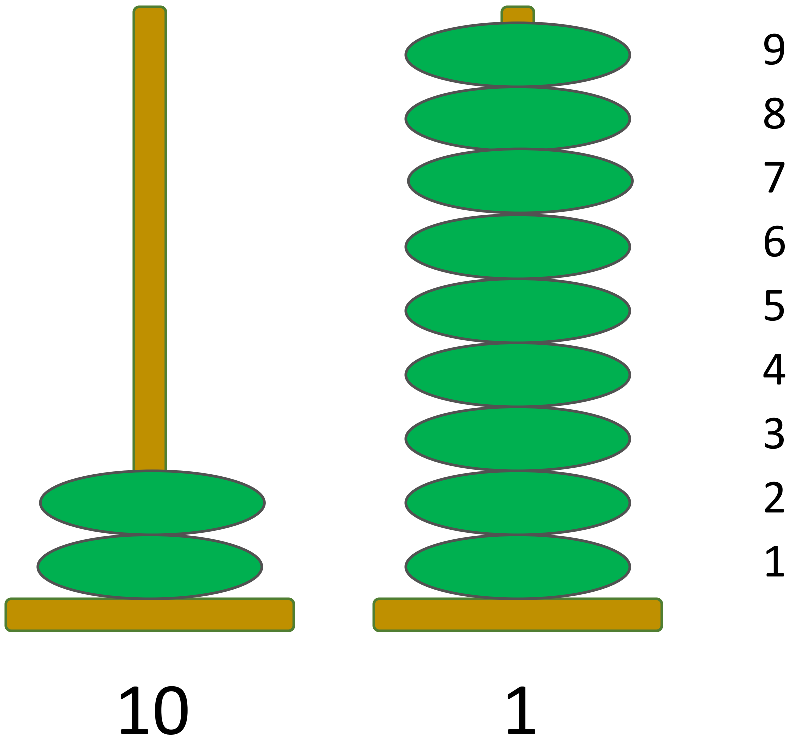


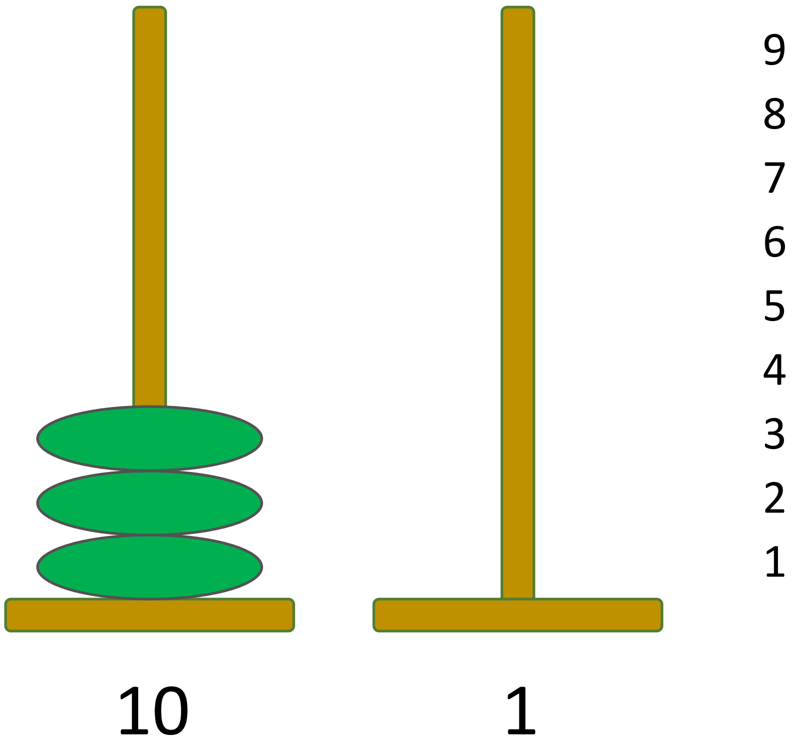
Positional Notation



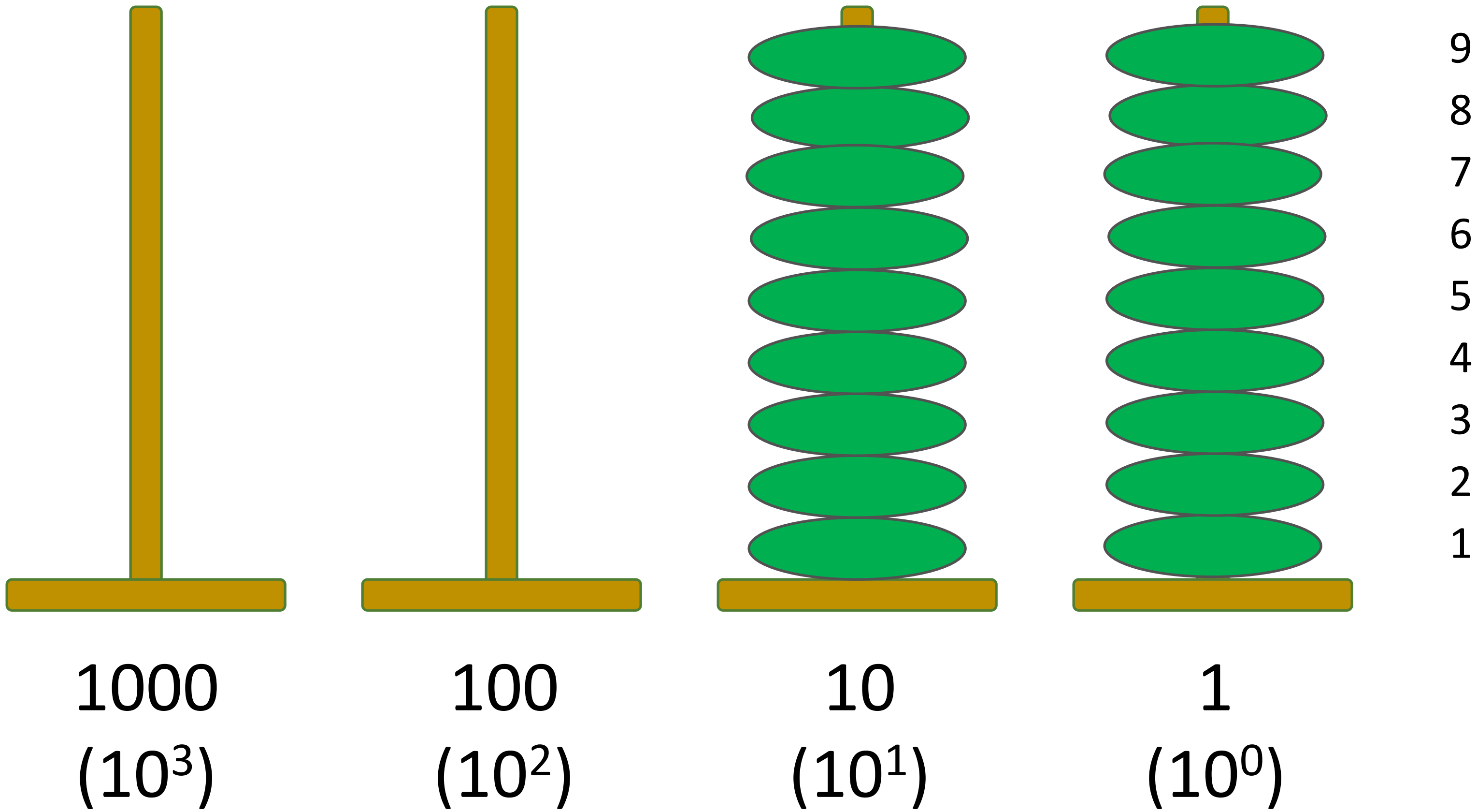

9
8
7
6
5
4
3
2
1
10
1

Positional Notation

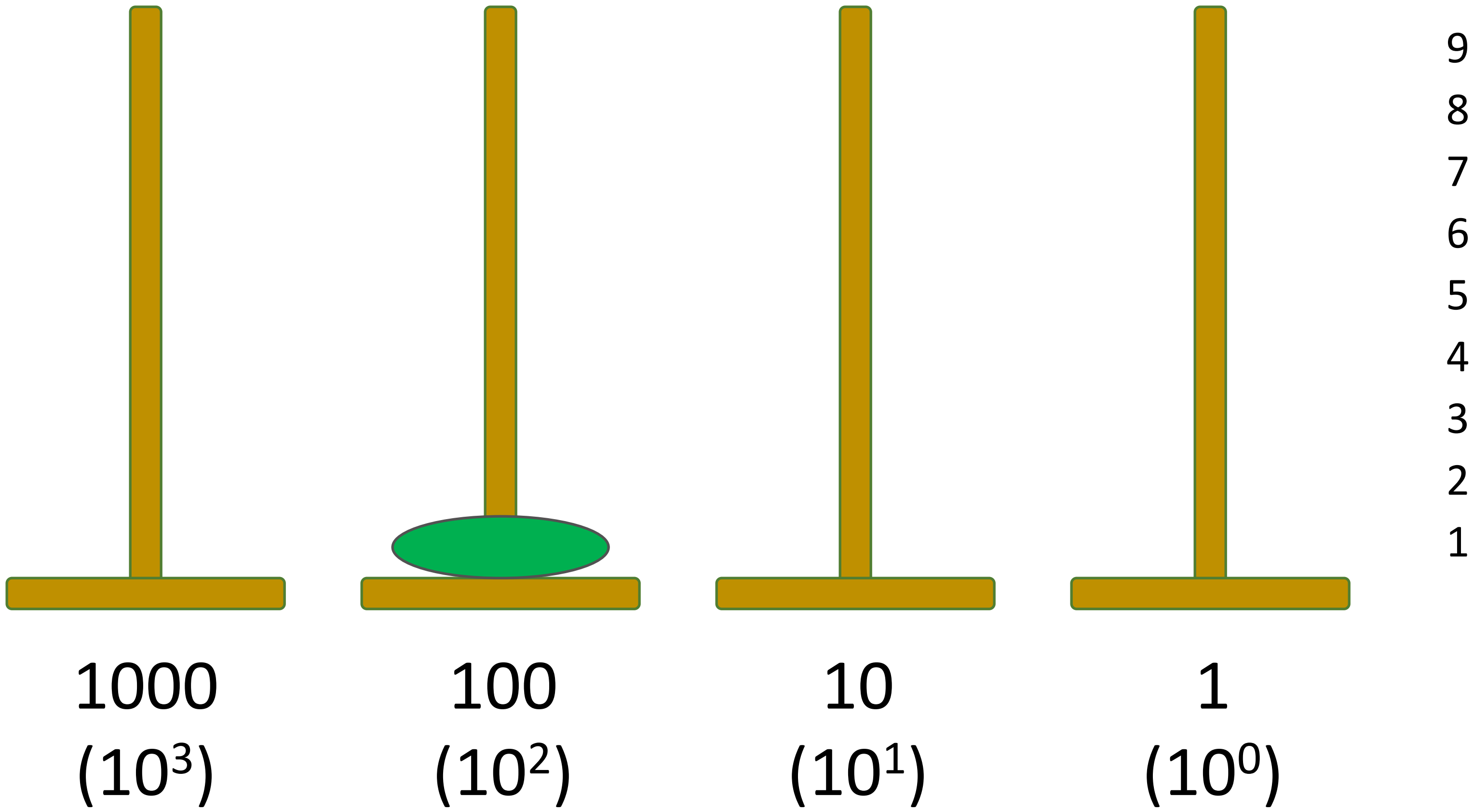



9
8
7
6
5
4
3
2
1
10
1

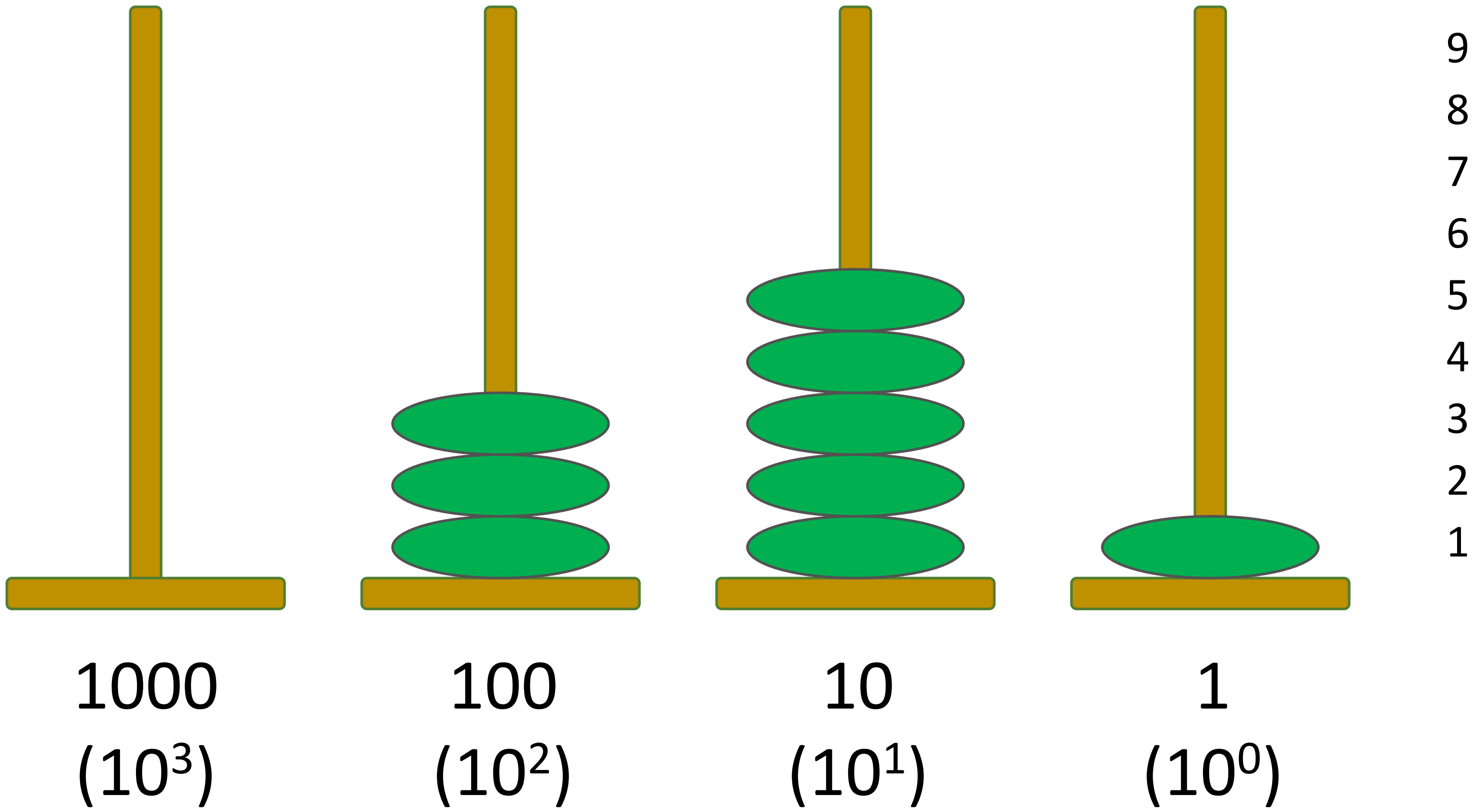
Positional Notation



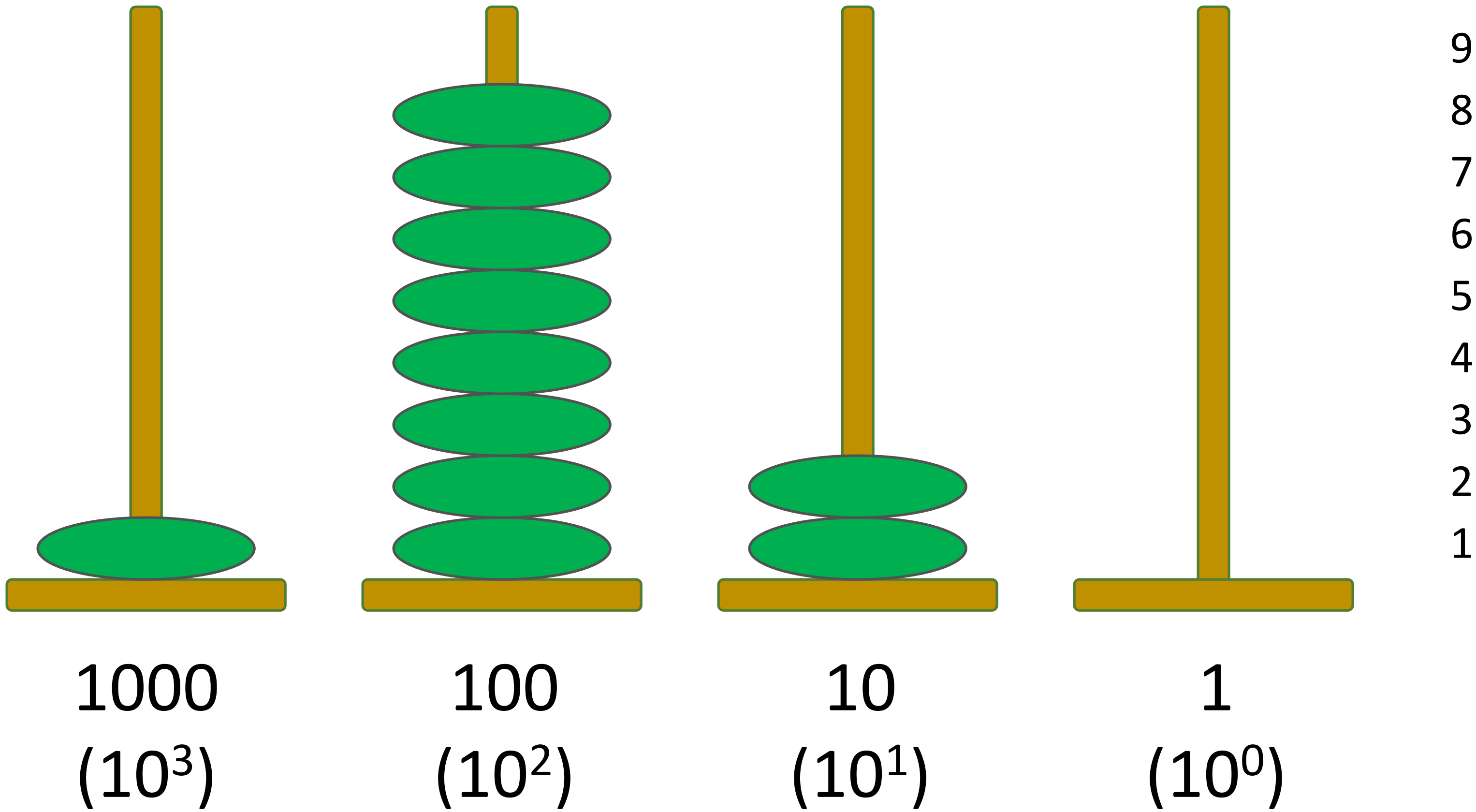
Positional Notation



Positional Notation



Positional Notation



1820

Positional Notation

4 3 2 1 0

- Allows us to count past 10 while using just 10 digits
- Each column of a number represents a power of the base. The exponent is the order of magnitude for the column

Positional Notation

$$10^4 \quad 10^3 \quad 10^2 \quad 10^1 \quad 10^0$$

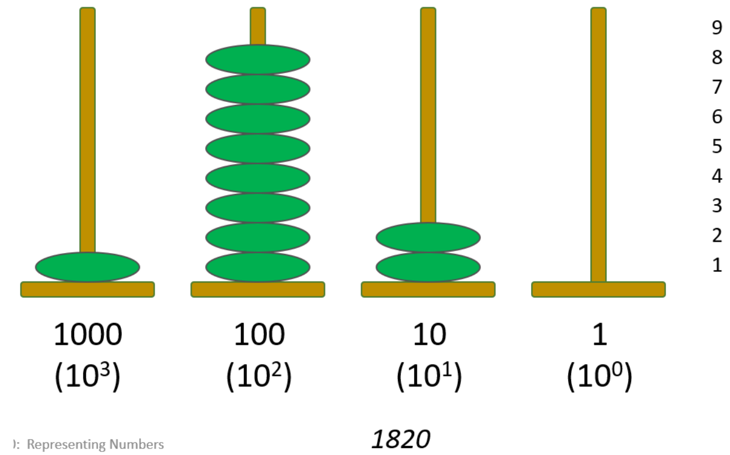
- The Decimal system is based on the number of digits we (usually) have

Positional Notation

…	10^4	10^3	10^2	10^1	10^0
…	10000	1000	100	10	1

- The magnitude of each column is the base, raised to its exponent

Positional Notation



10^4	10^3	10^2	10^1	10^0
10000	1000	100	10	1
	1	8	2	0
	1000	+800	+20	+0

=1820

- The magnitude of a number is determined by multiplying the magnitude of the column by the digit in the column and summing the products

Positional Notation

10^4	10^3	10^2	10^1	10^0
10000	1000	100	10	1
2	7	9	1	6
20000	+7000	+900	+10	+6

=27916

- The magnitude of a number is determined by multiplying the magnitude of the column by the digit in the column and summing the products

Another example

Representing Numbers

Positional representation

- Decimal (base 10)
- Binary (base 2)
- Octal (base 8)
- Hexadecimal (base 16)
- Conversion between different bases
  - Binary, hex, oct to decimal (recap)
  - Binary to/from hex and octal
  - Decimal to binary, hex, octal

Signed binary representation – 2's complement

- 2's complement (binary) to decimal
- Decimal to 2's complement (binary)

Binary representation of fractions

- Binary to decimal
- Decimal to binary

Non-Decimal Number Systems

Number can be in any base, 2, 3, 4, …, 8, 9,10,11… 16 …..

Decimal is base 10 and has 10 digit symbols:

0, 1, 2, 3, 4, 5, 6, 7, 8, 9 [0 ~ 10-1]

Binary is base 2 and has 2 digit symbols:

0,1 [0 ~ 2-1]

Octal is base 8 and has 8 digit symbols:

0,1,2,3,4,5,6,7 [0 ~ 8-1]

Hexdecimal is base 16 and has 16 digit symbols:

0,1,2,3,4,5,6,7,8,9, A,B,C,D,E,F [0 ~16-1]

For a value to exist in a given base, it can only contain the digits in that base, which range from 0 up to (but not including) the base.

[0 ~ base-1]

What bases can these values be in? 122, 198, 178, G1A4

Positional Notation

An n-digit unsigned integer in base b

(d_i is the digit in the i^{th} position in the number)

The number: $(d_{n-1}\ d_{n-2} \ldots d_i \ldots d_1\ d_0)_b$

Its value =

$$d_{n-1} \times b^{n-1} + d_{n-2} \times b^{n-2} + \ldots + d_i \times b^i \ldots + d_1 \times b^1 + d_0 \times b^0$$

Examples:

$(642)_{10} = 6 \times 10^2 + 4 \times 10^1 + 2 \times 10^0 = 642$

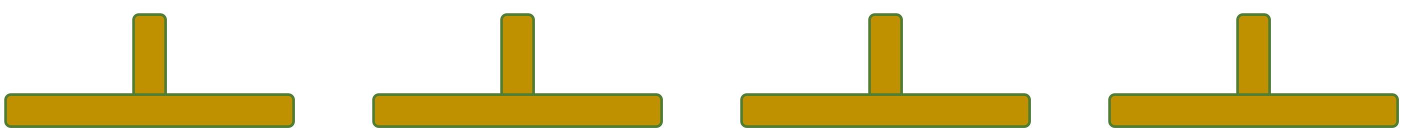
$(5073)_8 = 5 \times 8^3 + 0 \times 8^2 + 7 \times 8^1 + 3 \times 8^0 = 2619$

$(1011)_2 = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 11$

Binary

Binary is base 2 and has 2 digit symbols:

0,1

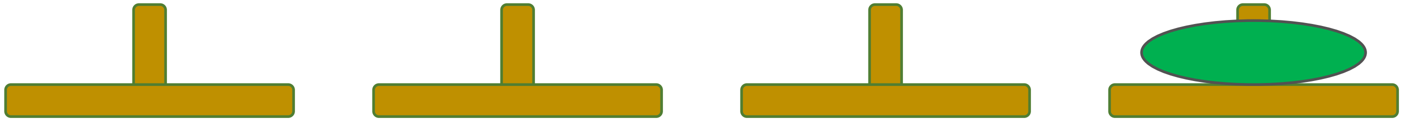


8	4	2	1
(2^3)	(2^2)	(2^1)	(2^0)

Binary

Binary is base 2 and has 2 digit symbols:

0,1

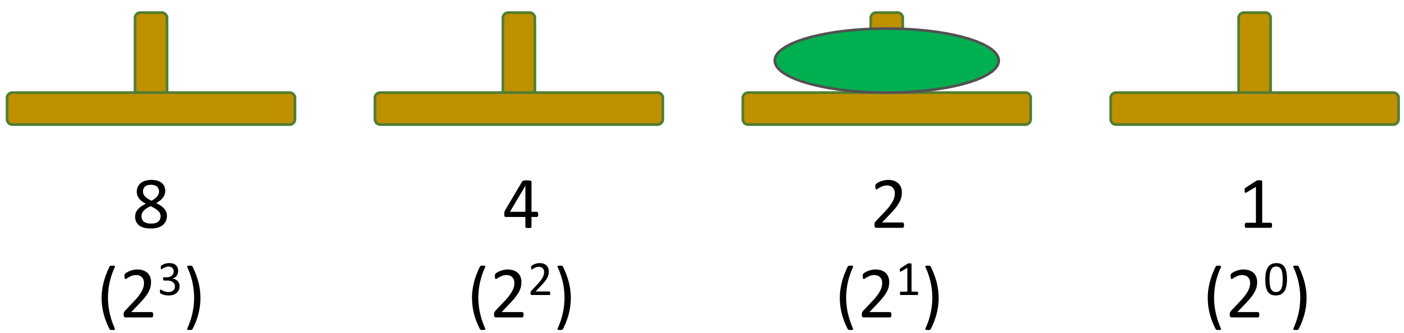


8	4	2	1
(2^3)	(2^2)	(2^1)	(2^0)

0001

Binary is base 2 and has 2 digit symbols:

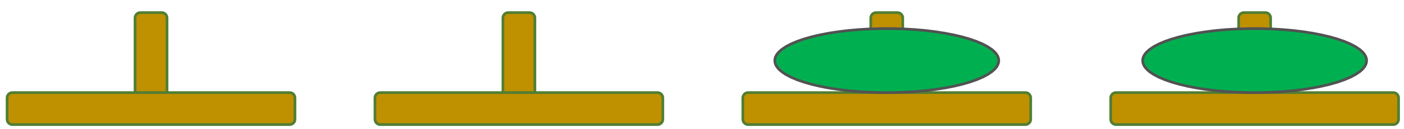
0,1



0010 Not 10 checkback

Binary is base 2 and has 2 digit symbols:

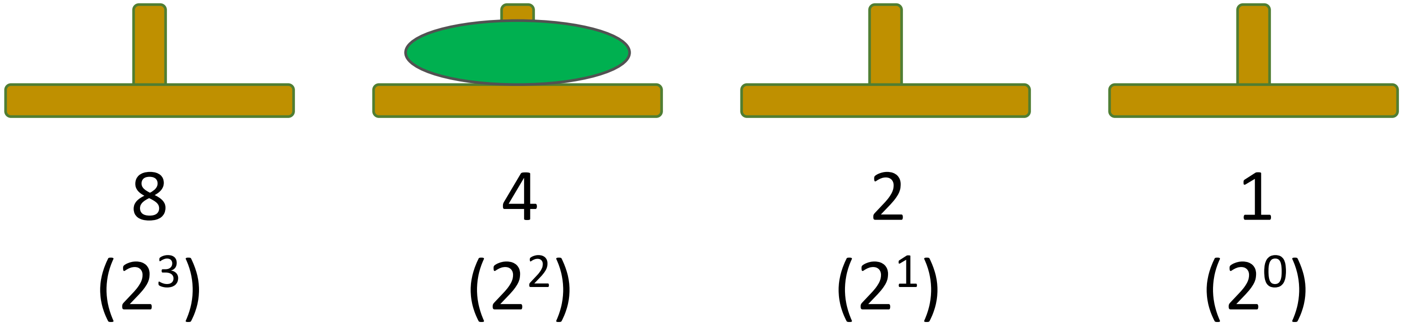
0,1



8	4	2	1
(2^3)	(2^2)	(2^1)	(2^0)

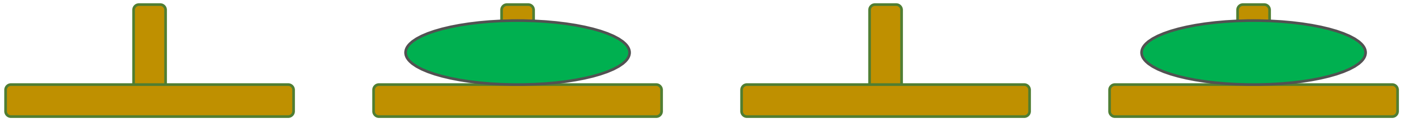
Binary is base 2 and has 2 digit symbols:

0,1



Binary is base 2 and has 2 digit symbols:

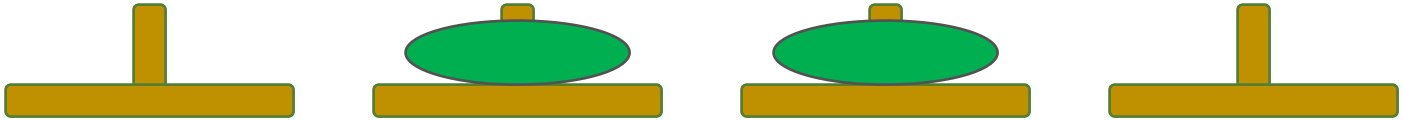
0,1



8	4	2	1
(2^3)	(2^2)	(2^1)	(2^0)

Binary is base 2 and has 2 digit symbols:

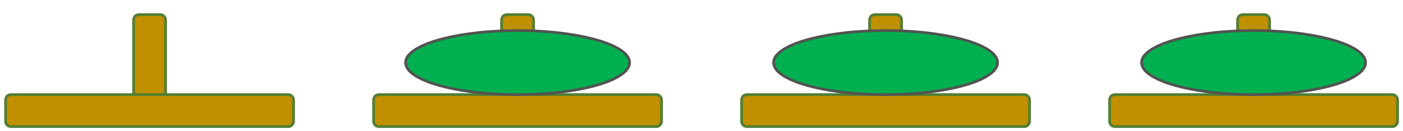
0,1



8	4	2	1
(2^3)	(2^2)	(2^1)	(2^0)

Binary is base 2 and has 2 digit symbols:

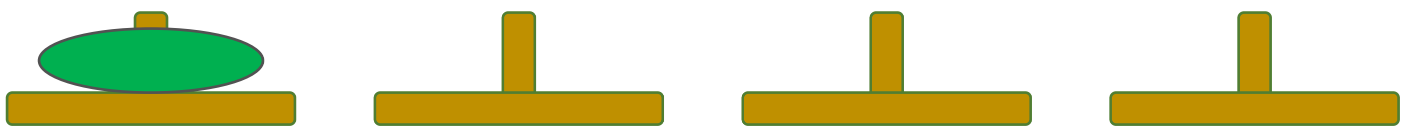
0,1



8	4	2	1
(2^3)	(2^2)	(2^1)	(2^0)

Binary is base 2 and has 2 digit symbols:

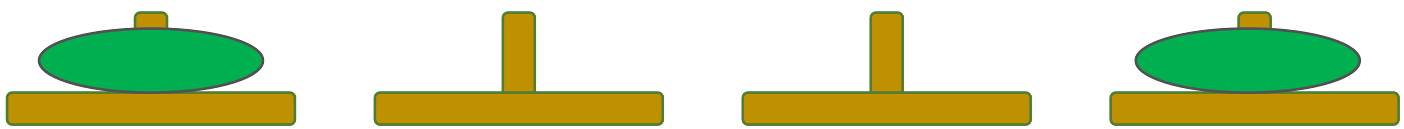
0,1



8	4	2	1
(2^3)	(2^2)	(2^1)	(2^0)

Binary is base 2 and has 2 digit symbols:

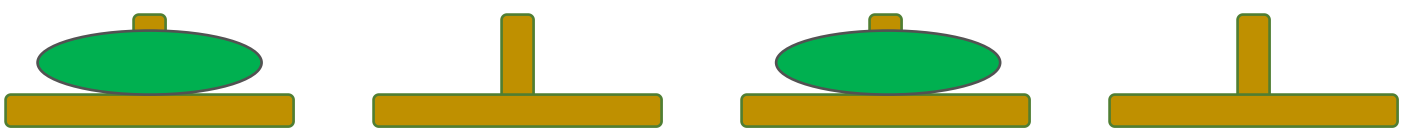
0,1



8	4	2	1
(2^3)	(2^2)	(2^1)	(2^0)

Binary is base 2 and has 2 digit symbols:

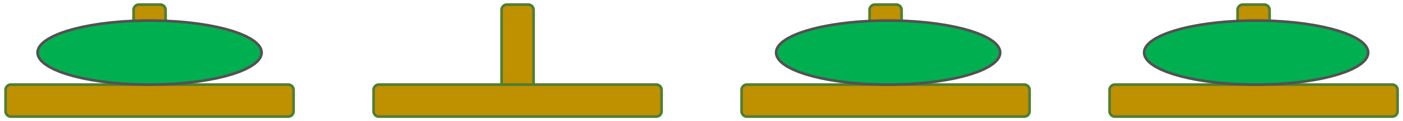
0,1



8 (2^3) | 4 (2^2) | 2 (2^1) | 1 (2^0)

Binary is base 2 and has 2 digit symbols:

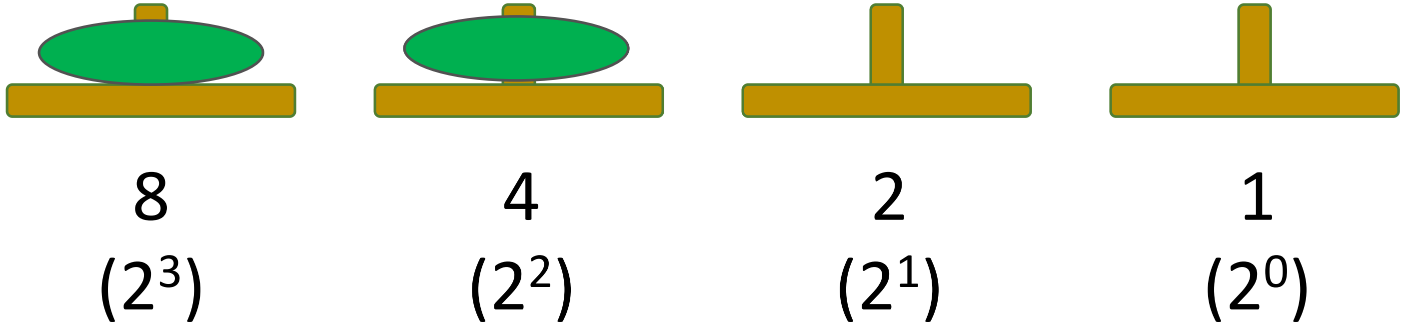
0,1



8 (2^3) 4 (2^2) 2 (2^1) 1 (2^0)

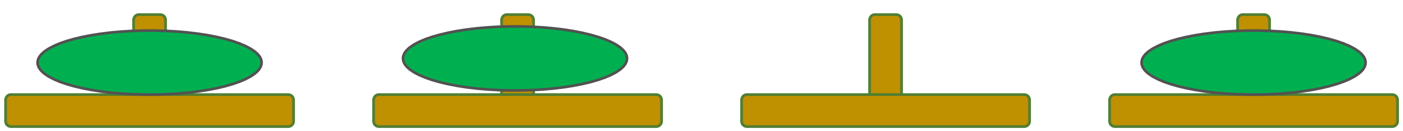
Binary is base 2 and has 2 digit symbols:

0,1



Binary is base 2 and has 2 digit symbols:

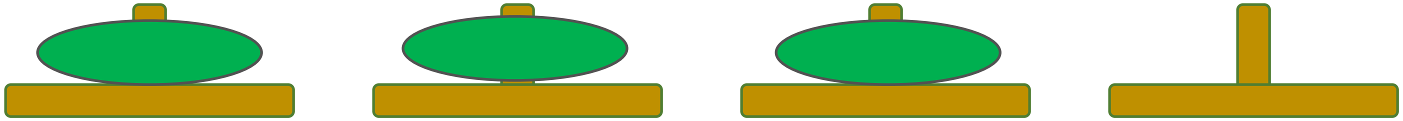
0,1



8	4	2	1
(2^3)	(2^2)	(2^1)	(2^0)

Binary is base 2 and has 2 digit symbols:

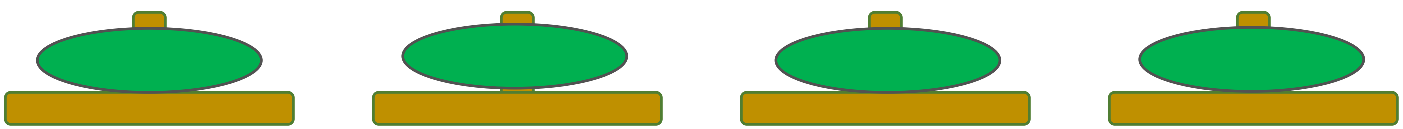
0,1



8 (2^3) 4 (2^2) 2 (2^1) 1 (2^0)

Binary is base 2 and has 2 digit symbols:

0,1



8	4	2	1
(2^3)	(2^2)	(2^1)	(2^0)

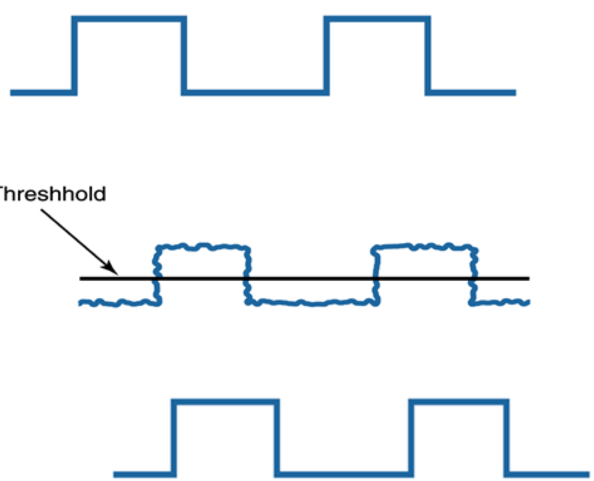
Binary Numbers and Computers

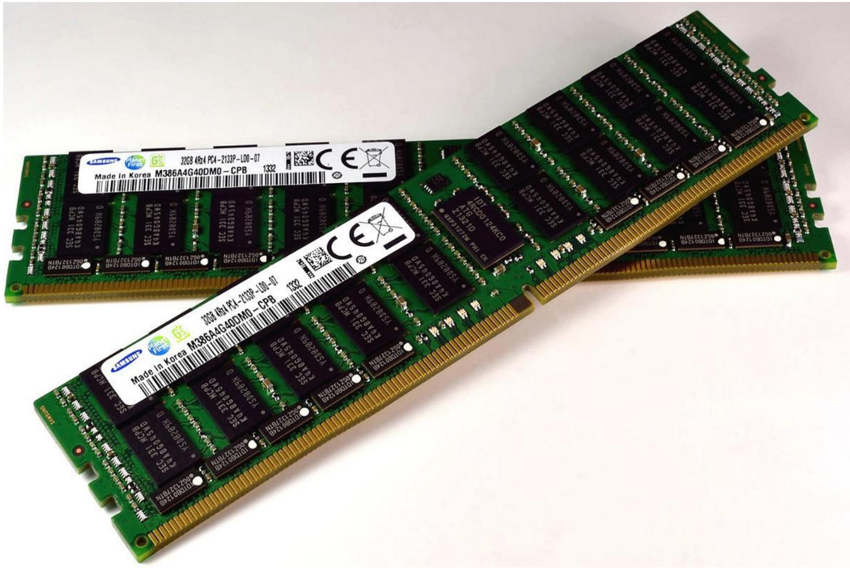
Computers have storage units called binary digits or bits

for instance,

Low Voltage = 0
High Voltage = 1

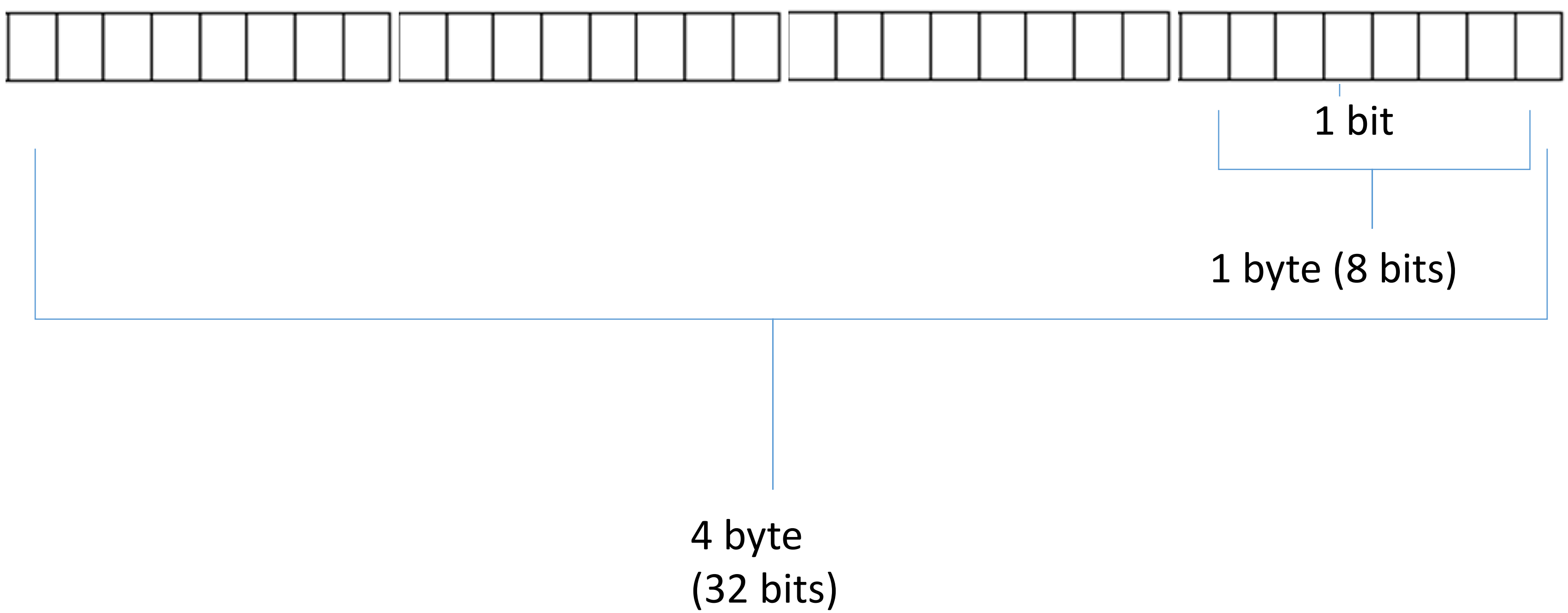
all bits are 0 or 1





- Bit/byte/K/M/G/T

- int x;



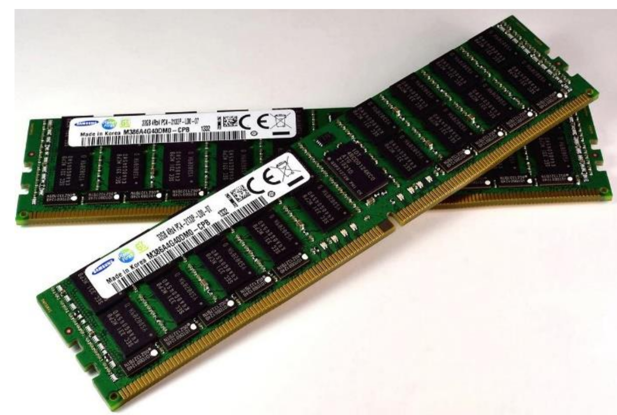
Decimal Notation

- base 10 or radix 10 ... uses 10 symbols

  $\boxed{0, 1, 2, 3, 4, 5, 6, 7, 8, 9}$
- Position represents powers of 10
- 5473_{10} or 5473

  $(5 * 10^3) + (4 * 10^2) + (7 * 10^1) + (3 * 10^0)$

Binary Notation



- base 2 ... uses only 2 symbols

  $\boxed{0, 1}$
- Position represents powers of 2
- 11010_2

  $(1 * 2^4) + (1 * 2^3) + (0 * 2^2) + (1 * 2^1) + (0 * 2^0)$ **= 26**

A Single Byte

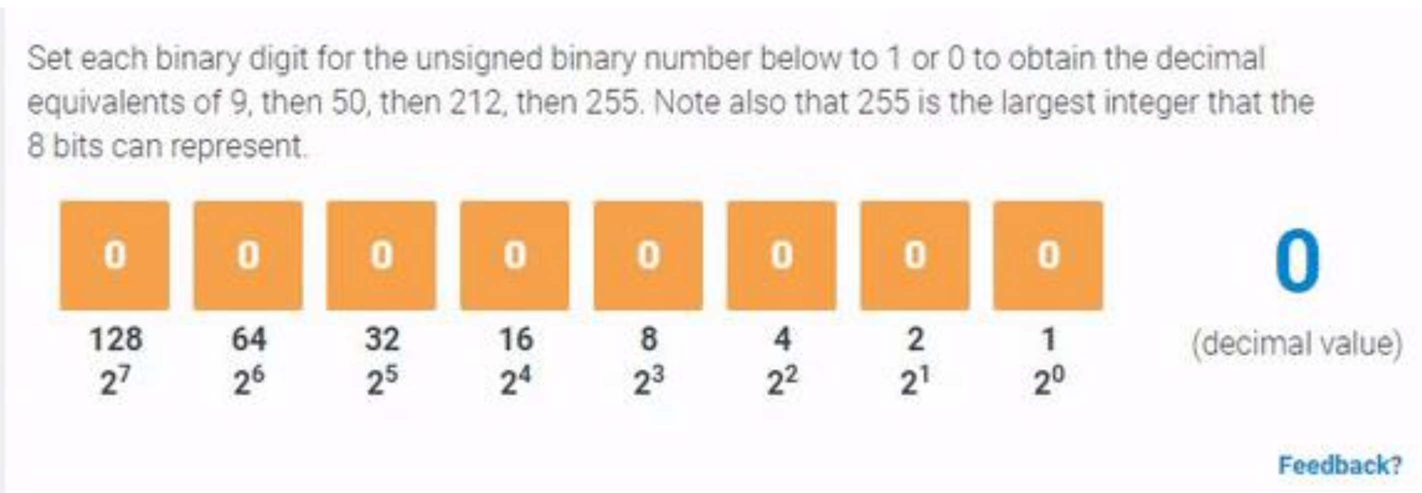
7	6	5	4	3	2	1	0

A Single Byte

2^7 2^6 2^5 2^4 2^3 2^2 2^1 2^0

A Single Byte

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
128	64	32	16	8	4	2	1



A Single Byte

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
128	64	32	16	8	4	2	1
0	0	0	0	0	0	0	0
0	+0	+0	+0	+0	+0	+ 0	+ 0

=0

There is also a representation for zero, giving 256 (2^8) possible combinations of 0 and 1 in 8 bits

1 bit: 2 values (0, 1); **2** bits: 4 values (00, 01, 10, 11); …

Every one **bit** added *doubles* the range of values

A Single Byte

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
128	64	32	16	8	4	2	1
1	1	1	1	1	1	1	1

128 +64 +32 +16 +8 +4 + 2 + 1

=255

255 is the largest value (in decimal) that can be expressed using 8 bits

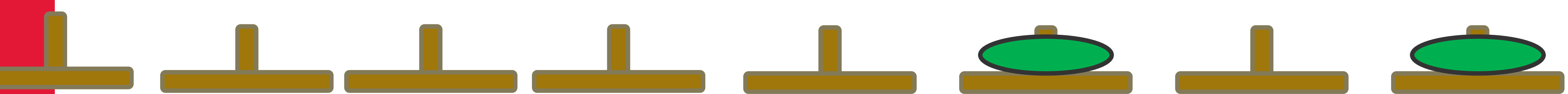
Max using 2 bits?

Max using 4 bits?

- 3 2 8 → $3*10^2 + 2*10^1 + 8*10^0 =$ 300 + 20 + 8 = 328 Decimal 328

  10^2 10^1 10^0

Binary representations



… 8 (2^3) 4 (2^2) 2 (2^1) 1 (2^0)

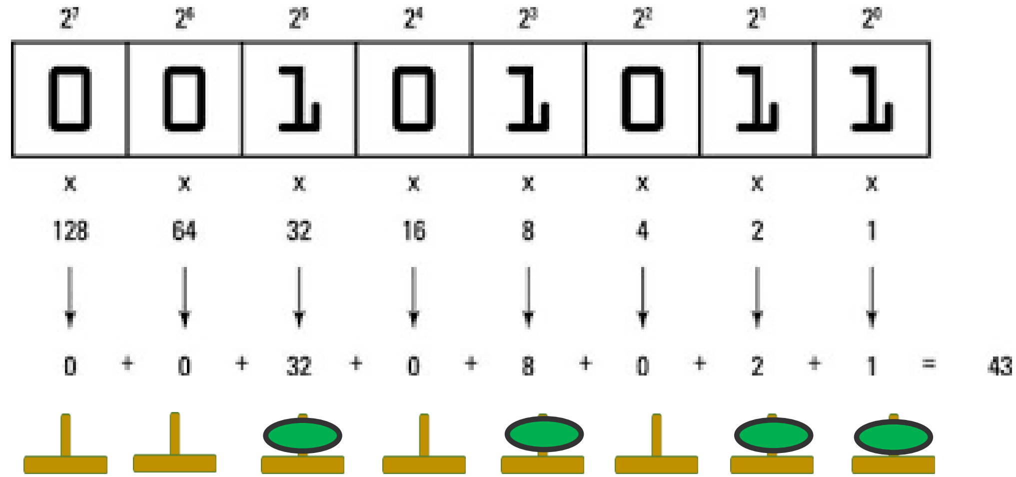
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

… 2^6 2^5 2^4 2^3 2^2 2^1 2^0

Binary

$1*2^2 + 0*2^1 + 1*2^0 =$

4 + 0 + 1 = 5



Binary Value				Decimal Representation				Decimal Value
				8	4	2	1	
0	0	0	0	0 +	0 +	0 +	0	0
0	0	0	1	0 +	0 +	0 +	1	1
0	0	1	0	0 +	0 +	2 +	0	2
0	0	1	1	0 +	0 +	2 +	1	3
0	1	0	0	0 +	4 +	0 +	0	4
0	1	0	1	0 +	4 +	0 +	1	5
0	1	1	0	0 +	4 +	2 +	0	6
0	1	1	1	0 +	4 +	2 +	1	7
1	0	0	0	8 +	0 +	0 +	0	8
1	0	0	1	8 +	0 +	0 +	1	9
1	0	1	0	8 +	0 +	2 +	0	10

8	4	2	1

Binary, Oct and Hex to Decimal

Give the Decimal equivalent of the following

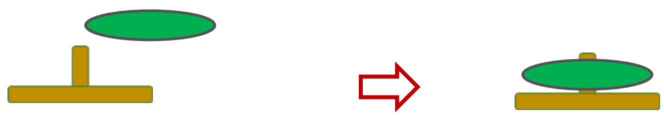
- Binary $000\ldots010_2$ 2
- Binary $000\ldots..0111_2$ 7
- Binary $000\ldots..1010_2$ 10
- Binary $000\ldots11010_2$ 26
- Binary $000\ldots1010111_2$ 87
- Binary $000\ldots10100010_2$ 162

Arithmetic in Binary

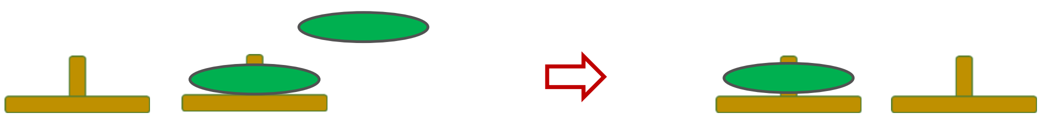
Remember that there are only 2 digit symbols in binary, 0 and 1

0 + 0 is 0

0 + 1 is 1



1 + 1 is 0 with a carry



Carry Values

```
 1 0 1 1 1 1 1
   1 0 1 0 1 1 1
 + 1 0 0 1 0 1 1
 ---------------
 1 0 1 0 0 0 1 0
```

Binary representations, arithmetic

- 3 + 4 ?

```
 00…0 0 1 1
 00…0 1 0 0
 ----------
 00…0 1 1 1     7
```

- 3 + 1 ?

```
 0 0…0 0 1 1
 0 0…0 0 0 1
 -----------
 0 0 …01 0 0    4
```

- 3 + 5 ?

```
 0 0…0 0 1 1
 0 0…0 1 0 1
 -----------
 0 0…1 0 0 0    8
```

- 3 + 7 ?

```
 0 0…0 0 1 1
 0 0…0 1 1 1
 -----------
 0 0…1 0 1 0    10
```

Remember that there are only 2 digit symbols in binary, 0 and 1

1 + 1 is 0 with a carry
1+1+1 is 1 with a carry

```
 1 0 1 1 1 1 1
   1 0 1 0 1 1 1     87
  +1 0 0 1 0 1 1     75
 ---------------
 1 0 1 0 0 0 1 0    162
```

Representing Numbers

Positional representation

- Decimal (base 10)
- Binary (base 2)
- Octal (base 8)
- Hexadecimal (base 16)
- Conversion between different bases
  - Binary, hex, oct to decimal (recap)
  - Binary to/from hex and octal
  - Decimal to binary, hex, octal

Signed binary representation – 2’s complement

- 2’s complement (binary) to decimal
- Decimal to 2’s complement (binary)

Binary representation of fractions

- Binary to decimal
- Decimal to binary

Octal Notation

For base B, B digit symbols [0~B-1]

- base 8 ... uses 8 symbols

  0, 1, 2, 3, 4, 5, 6, 7

- Position represents power of 8
- 1523_8

  $(1 * 8^3) + (5 * 8^2) + (2 * 8^1) + (3 * 8^0)$ = 851

Hexadecimal Notation

- base 16 or ‘hex’ ... uses 16 symbols

  0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

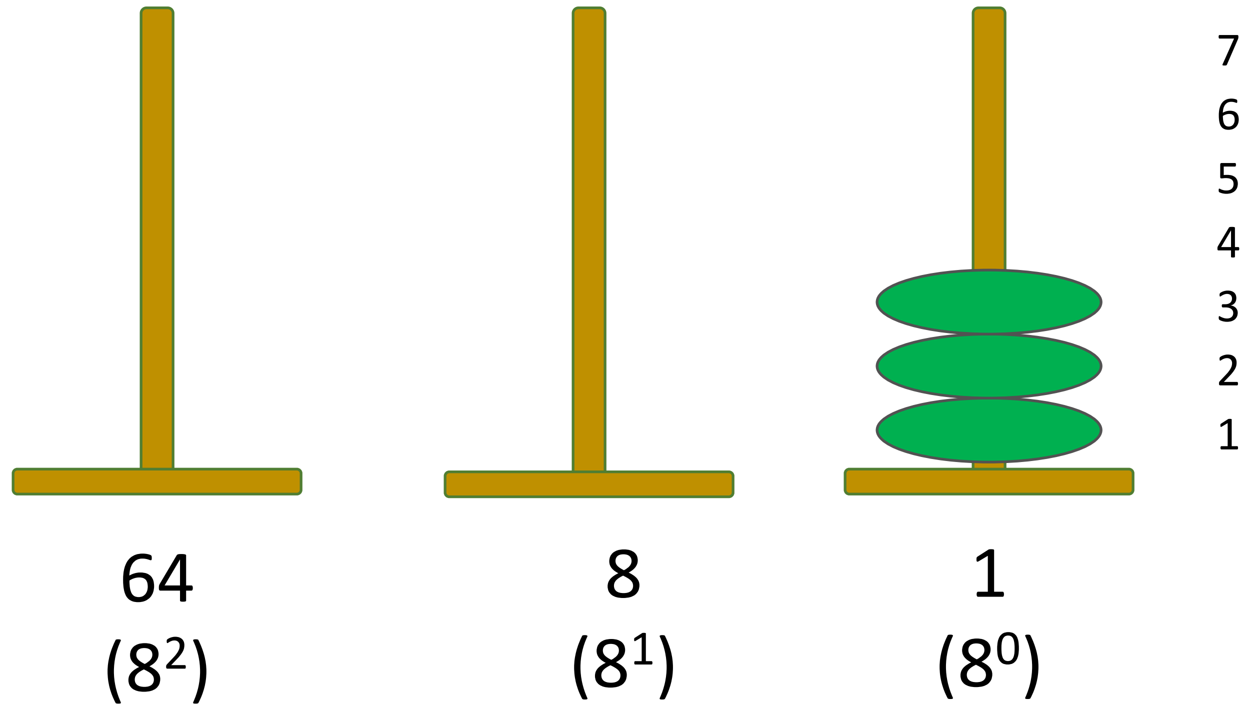
- Position represents powers of 16
- $B65F_{16}$ or 0xB65F

  $(11 * 16^3) + (6 * 16^2) + (5 * 16^1) + (15 * 16^0)$ = 46687

Positional Notation

Octal is base 8 and has 8 digit symbols:

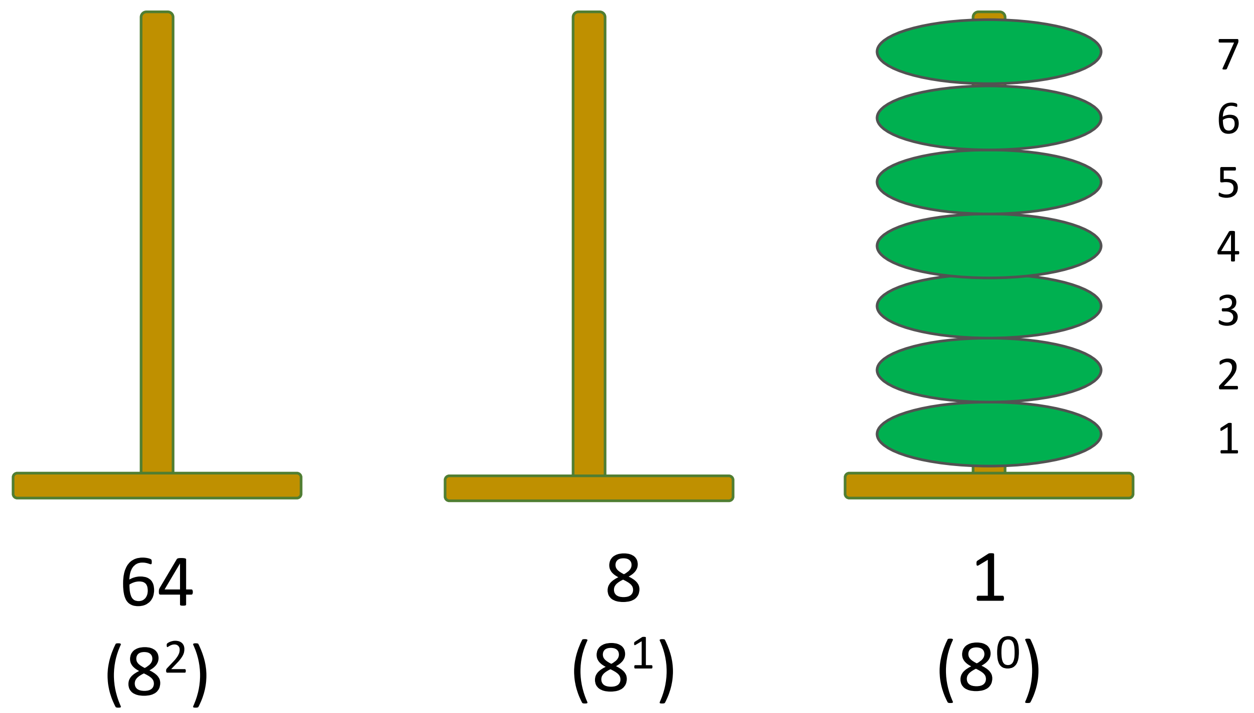
0,1,2,3,4,5,6,7



Positional Notation

Octal is base 8 and has 8 digit symbols:

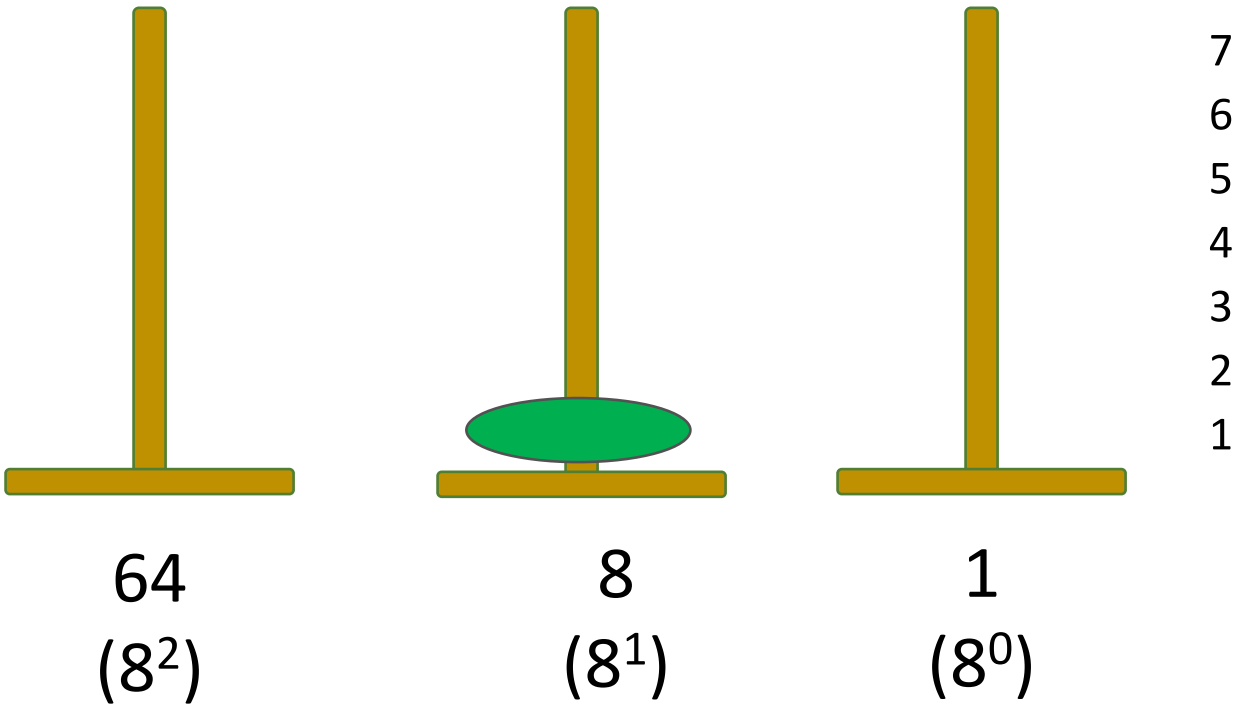
0,1,2,3,4,5,6,7



Positional Notation

Octal is base 8 and has 8 digit symbols:

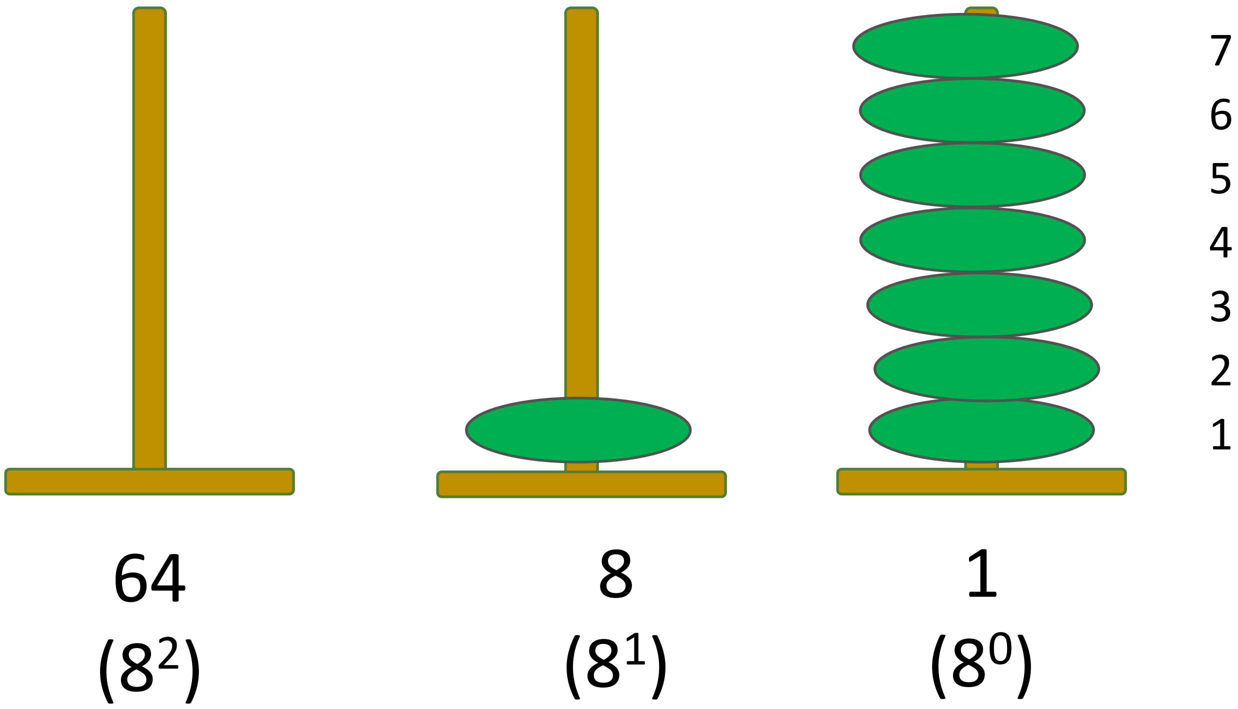
0,1,2,3,4,5,6,7



Positional Notation

Octal is base 8 and has 8 digit symbols:

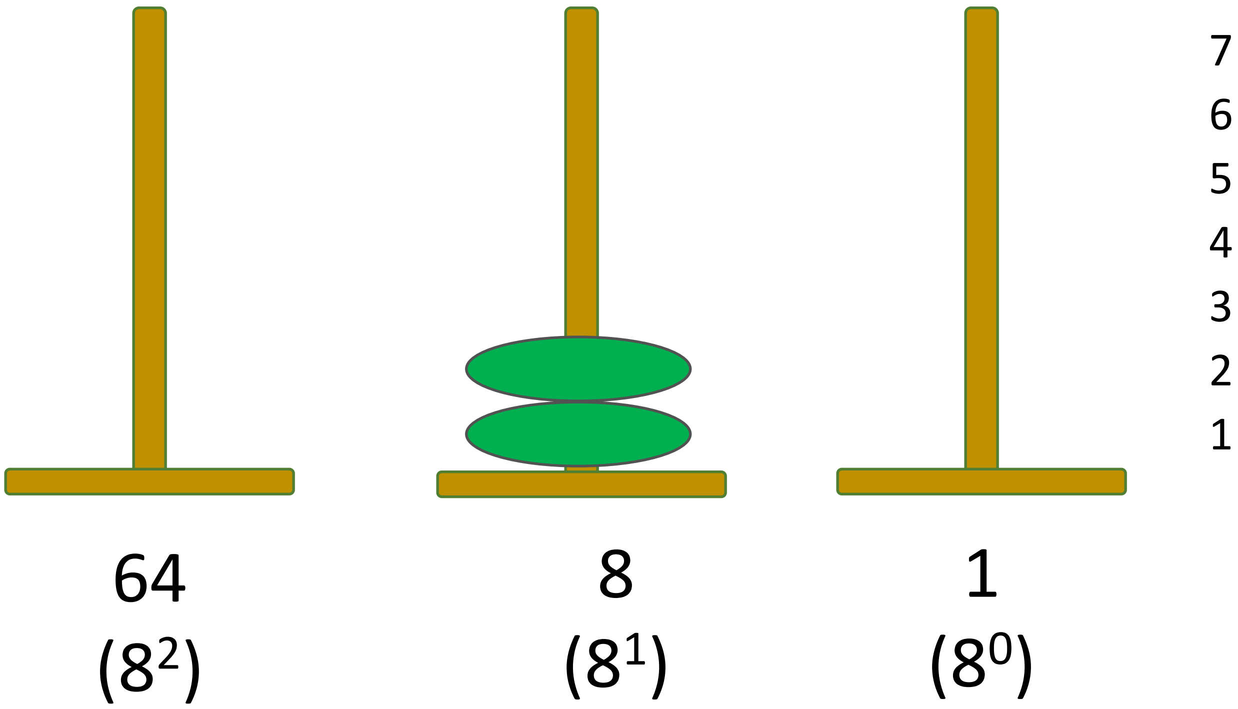
0,1,2,3,4,5,6,7



Positional Notation

Octal is base 8 and has 8 digit symbols:

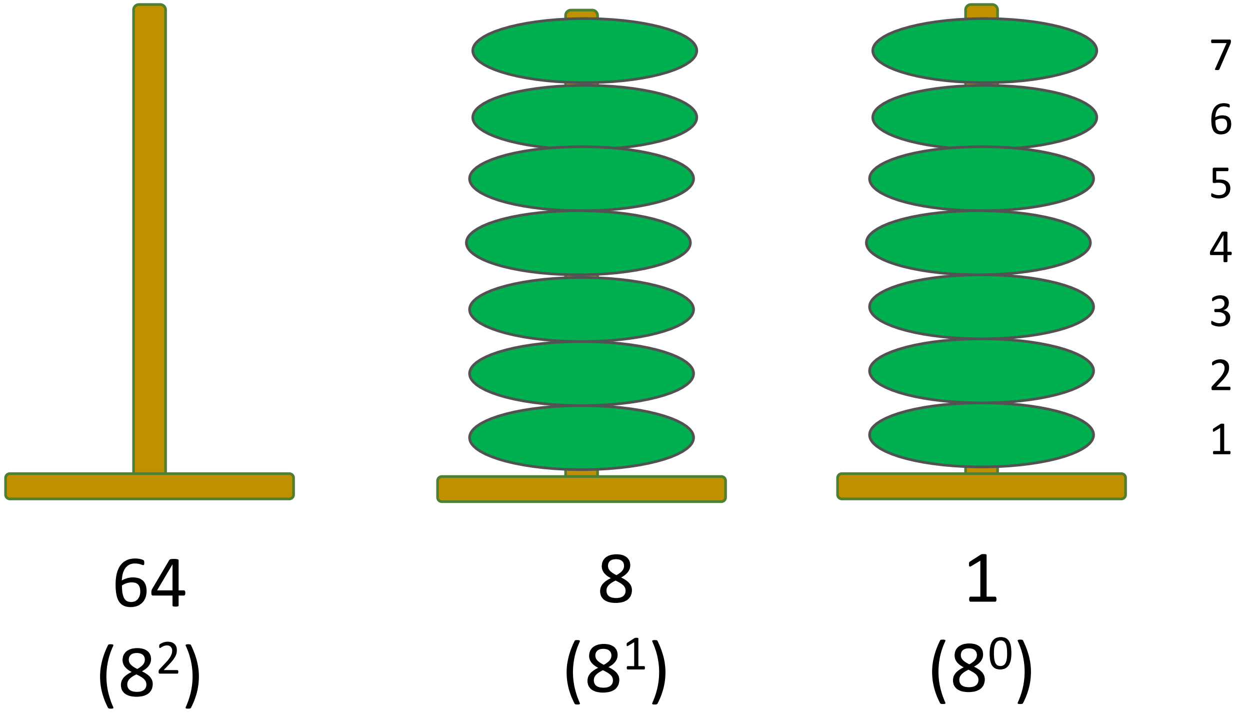
0,1,2,3,4,5,6,7



Positional Notation

Octal is base 8 and has 8 digit symbols:

0,1,2,3,4,5,6,7

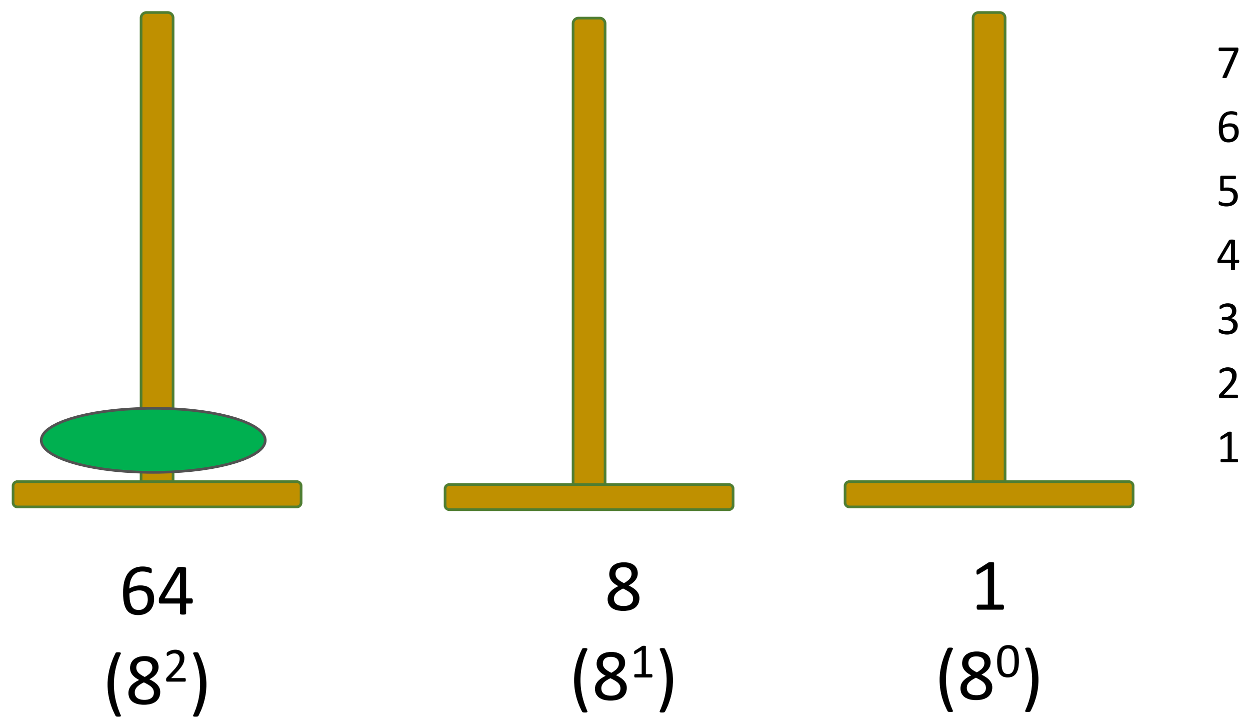


Must be 63, why?

Positional Notation

Octal is base 8 and has 8 digit symbols:

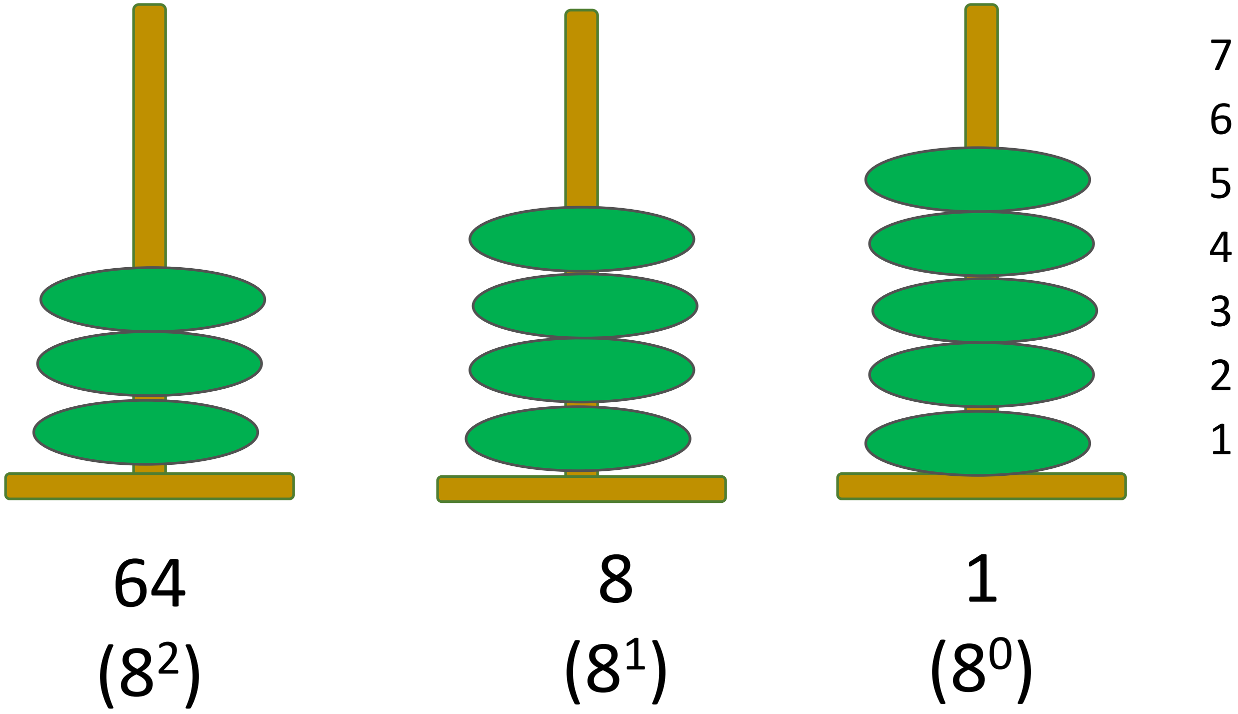
0,1,2,3,4,5,6,7



Positional Notation

Octal is base 8 and has 8 digit symbols:

0,1,2,3,4,5,6,7



Ask to do

Others To decimal

- 3 4 5

10^2 10^1 10^0

→ $3*10^2 + 4*10^1 + 5*10^0 =$
$300 + 40 + 5 = 345$

Decimal 328

- 1 0 1

2^2 2^1 2^0

→ $1*2^2 + 0*2^1 + 1*2^0 =$
$4 + 0 + 1 = 5$

0000000000000101

Binary

- 3 4 5

8^2 8^1 8^0

→ $3*8^2 + 4*8^1 + 5*8^0 =$
$192 + 32 + 5 = 229$

Octal 345

- 3 8 5
- 3 4 9

→ 

Octal

You should know these conversions.

Octal Notation

- base 8 ... uses 8 symbols

  0, 1, 2, 3, 4, 5, 6, 7
- Position represents power of 8
- 1523_8

  $(1 * 8^3) + (5 * 8^2) + (2 * 8^1) + (3 * 8^0)$ = 851

Hexadecimal Notation

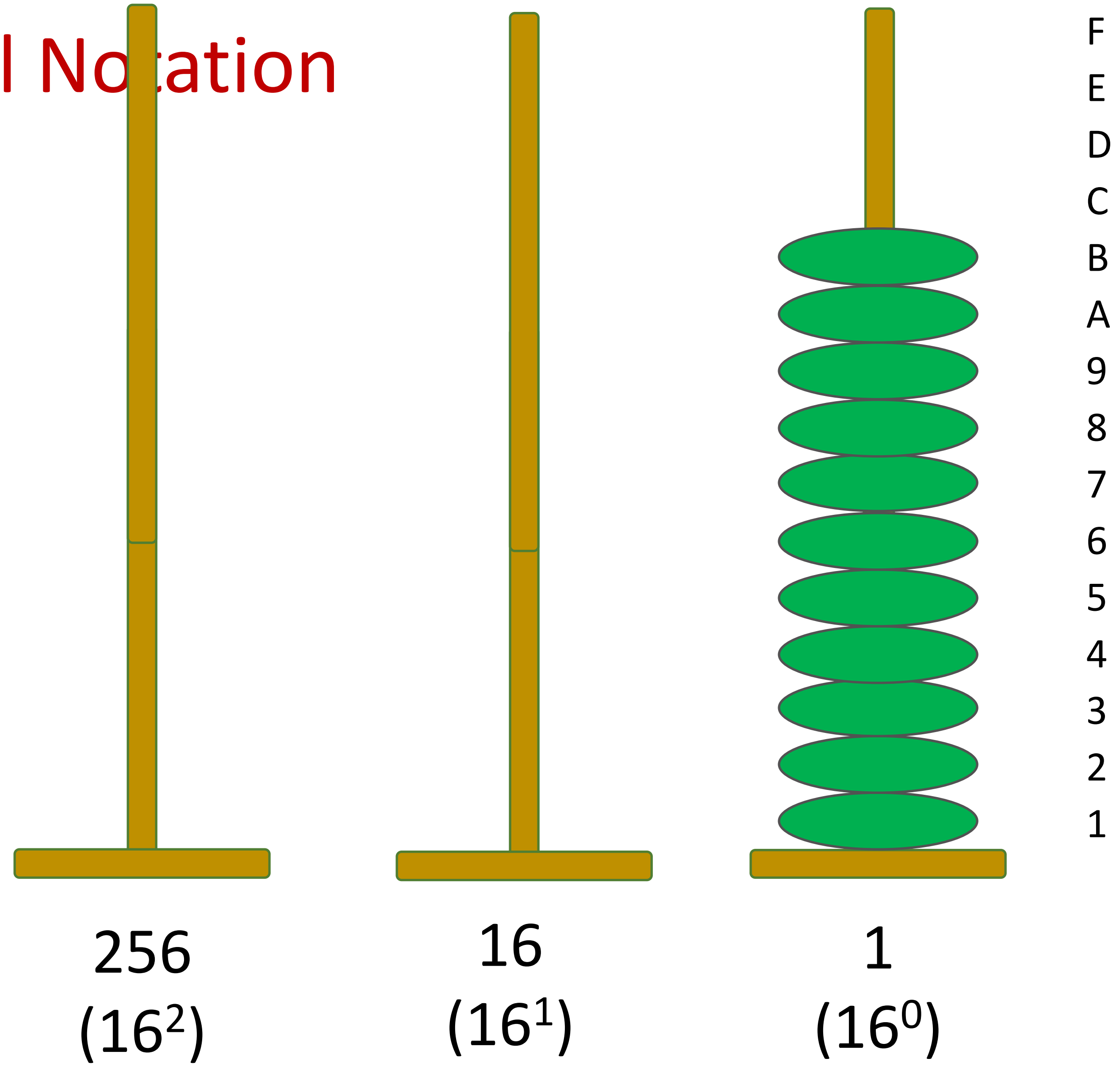
- base 16 or 'hex' ... uses 16 symbols

  0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

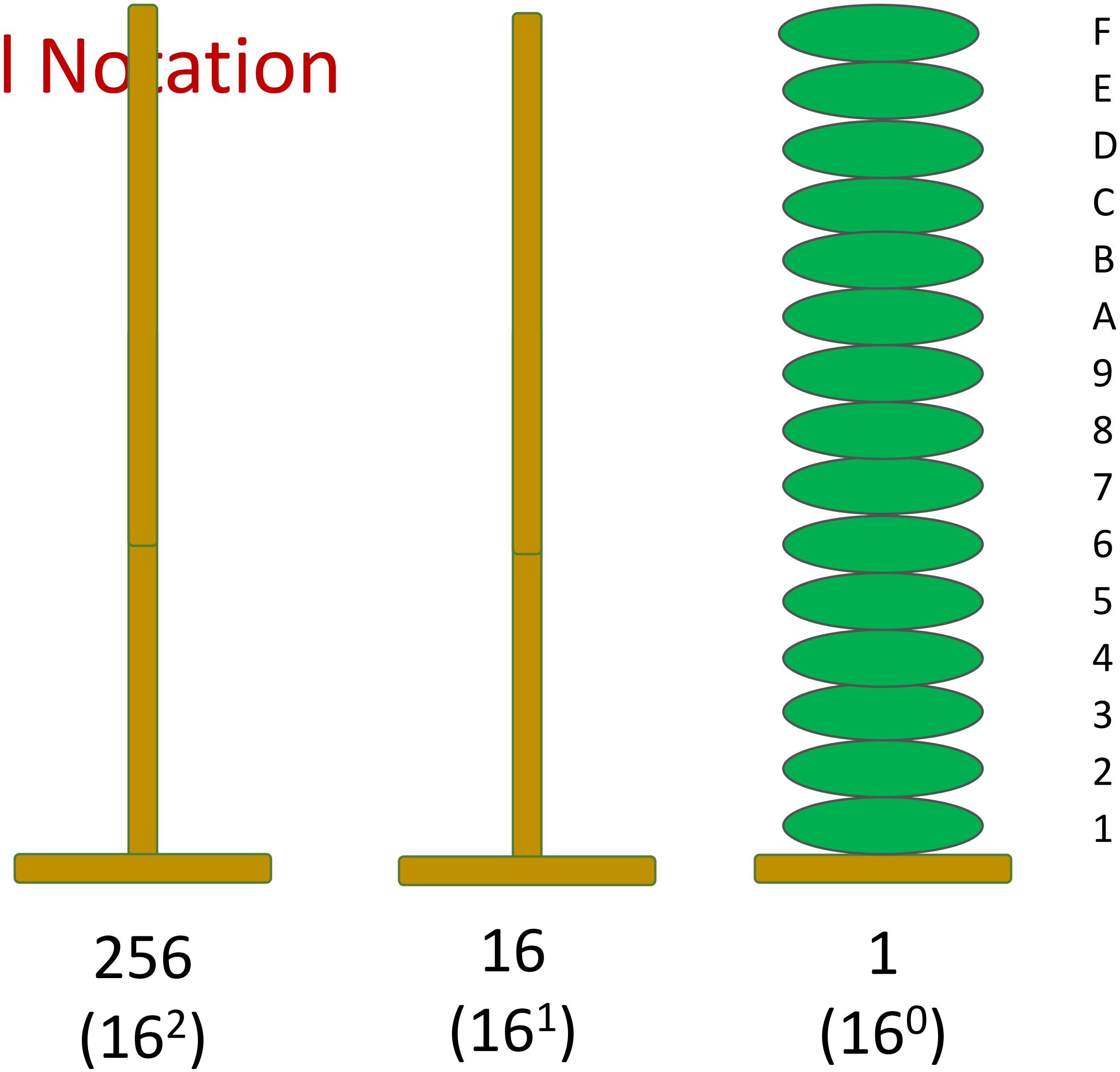
  10 11 12 13 14 15
- Position represents powers of 16
- $B65F_{16}$ or 0xB65F

  $(11 * 16^3) + (6 * 16^2) + (5 * 16^1) + (15 * 16^0)$ = 46687

Positional Notation

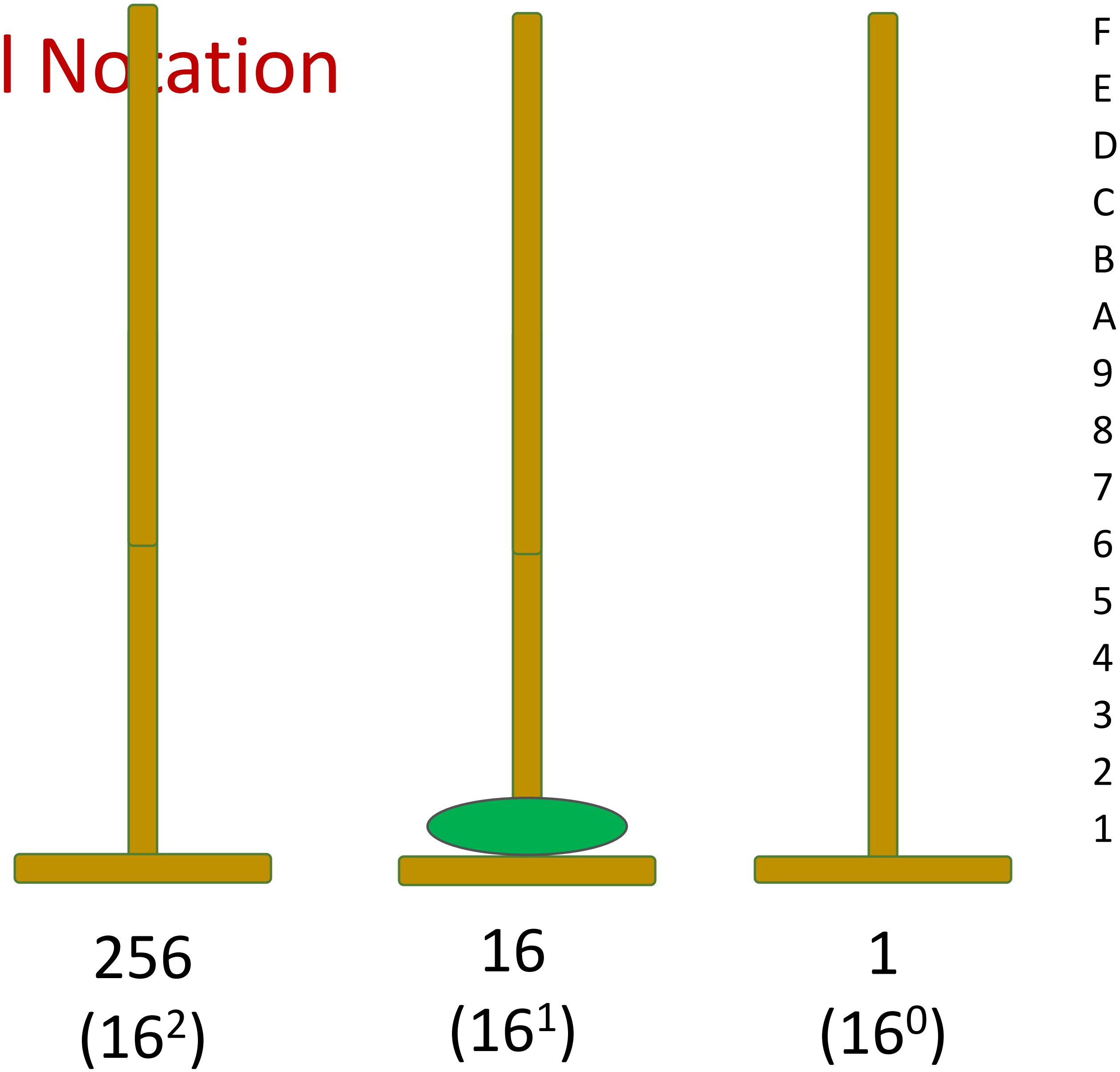


Positional Notation

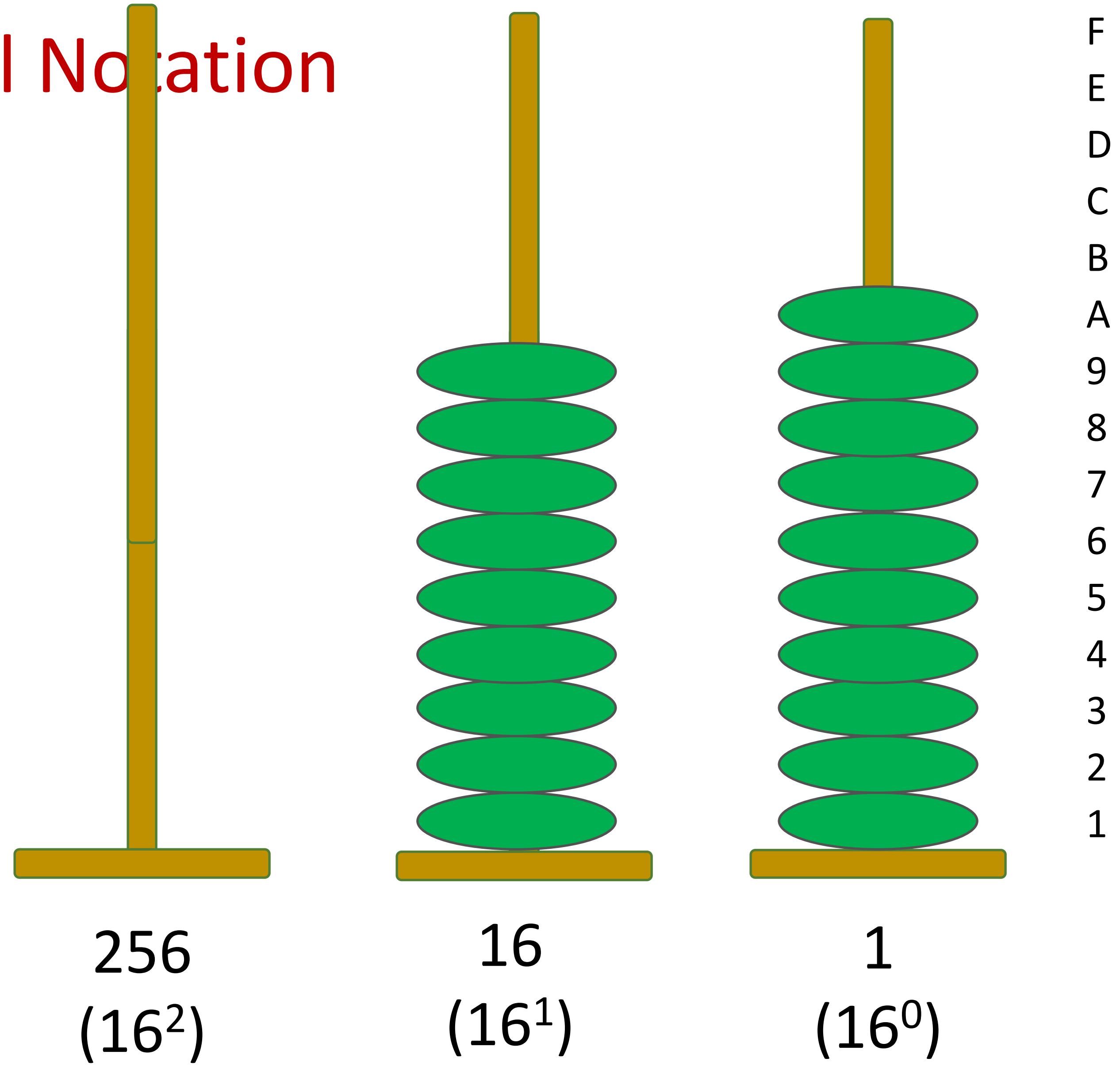


BTW, not valid for base 8

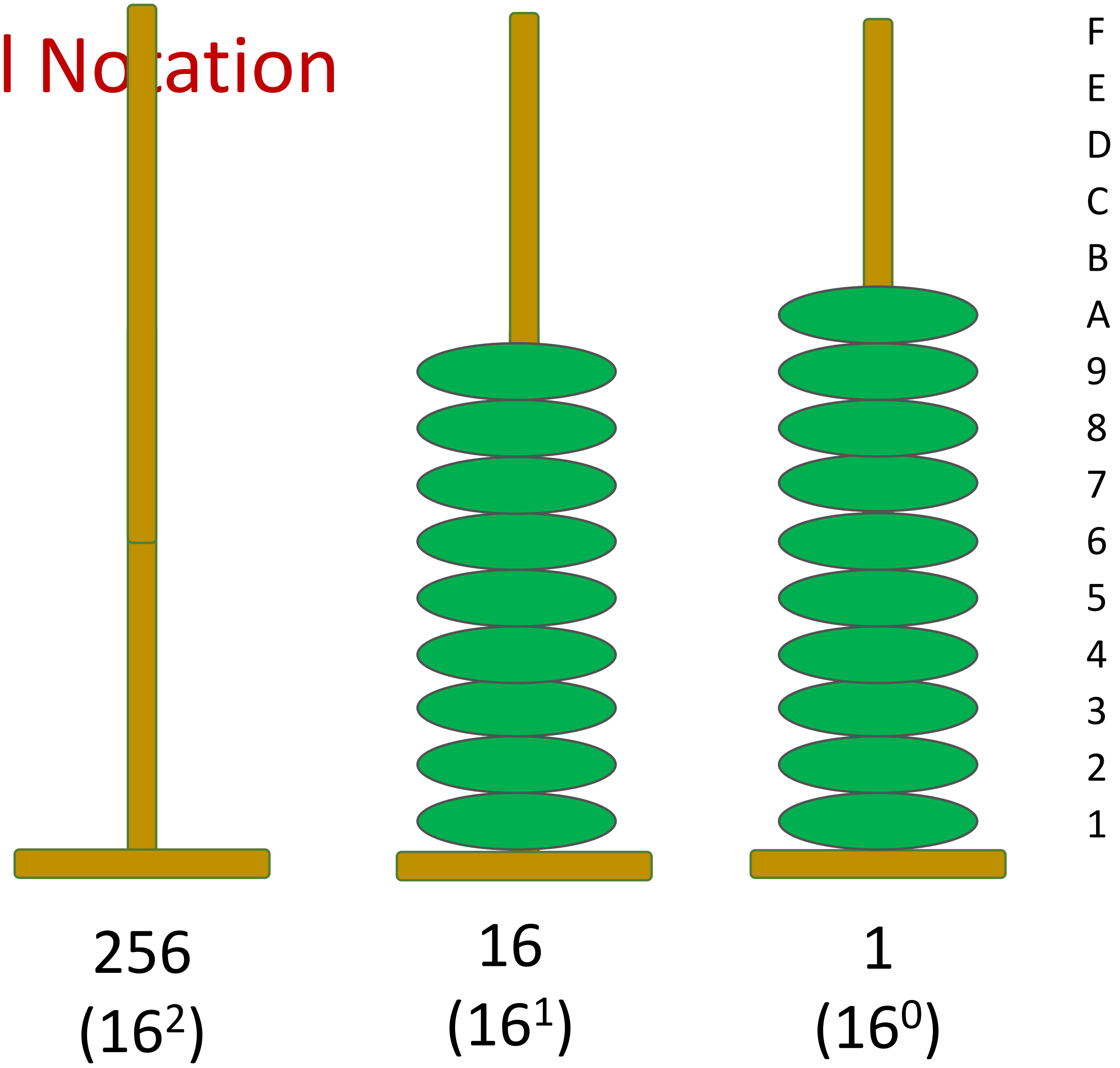
Positional Notation



Positional Notation

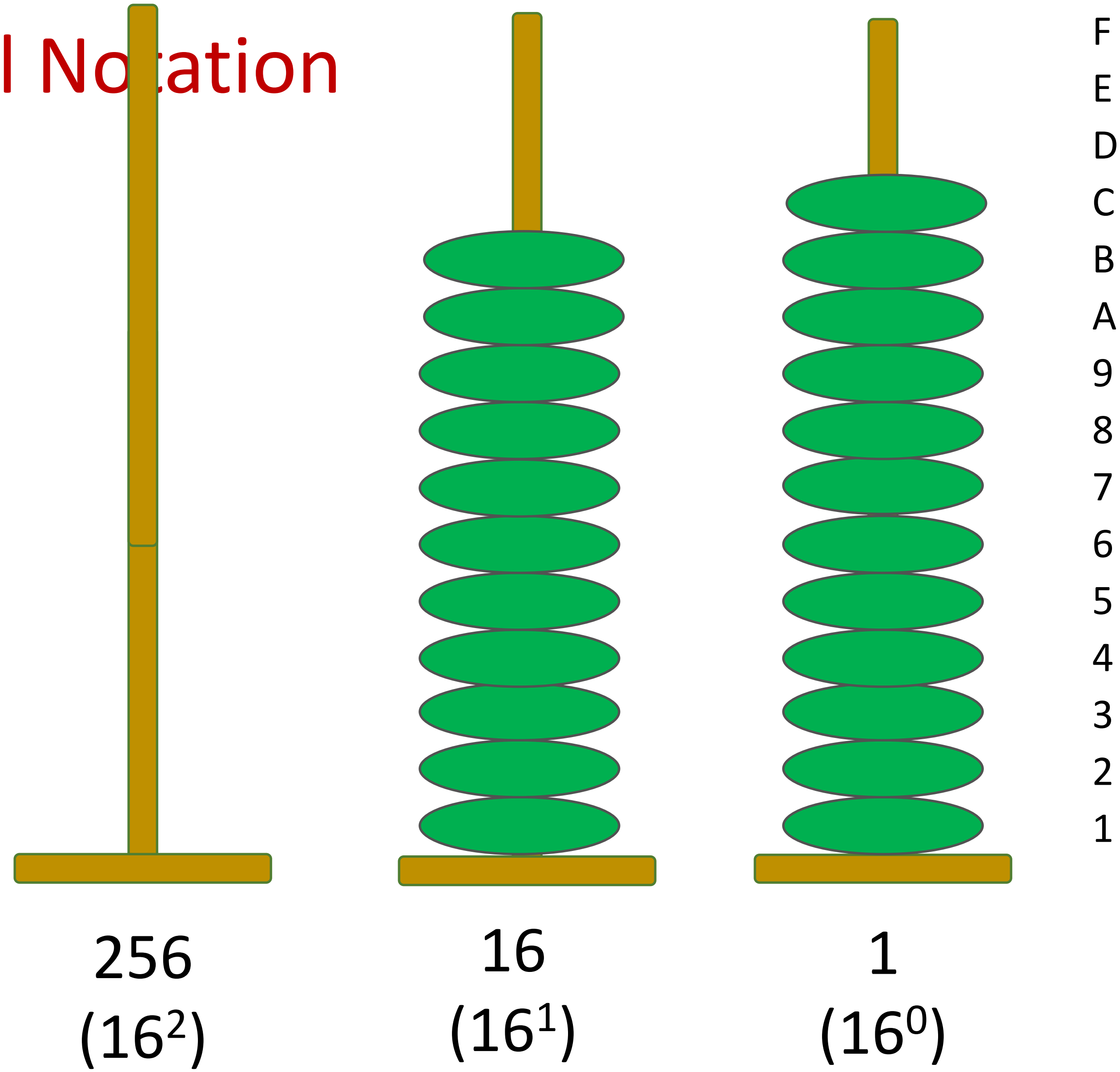


Positional Notation



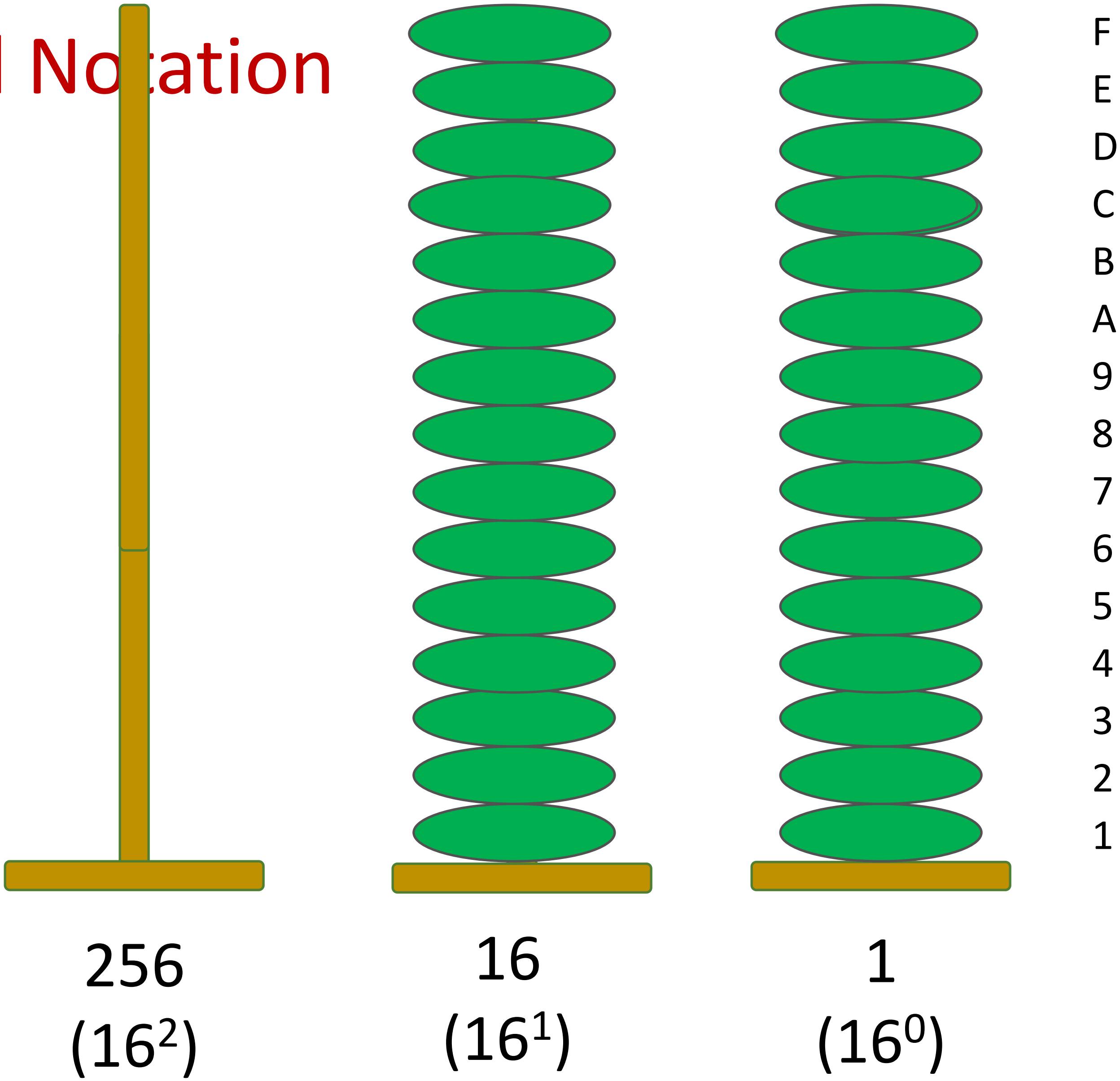
Check with cleavebook

Positional Notation



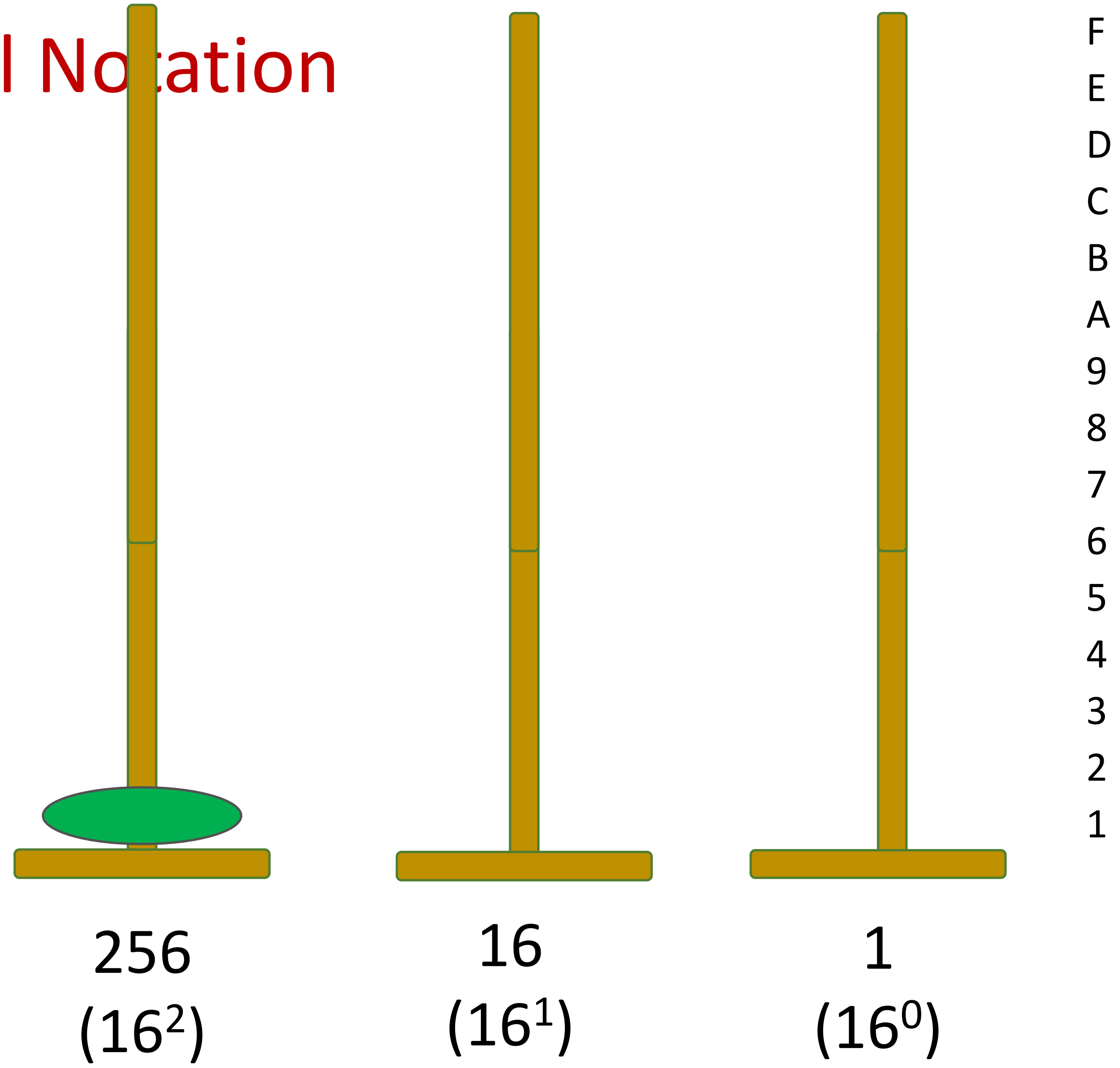
Check with cleavebook

Positional Notation

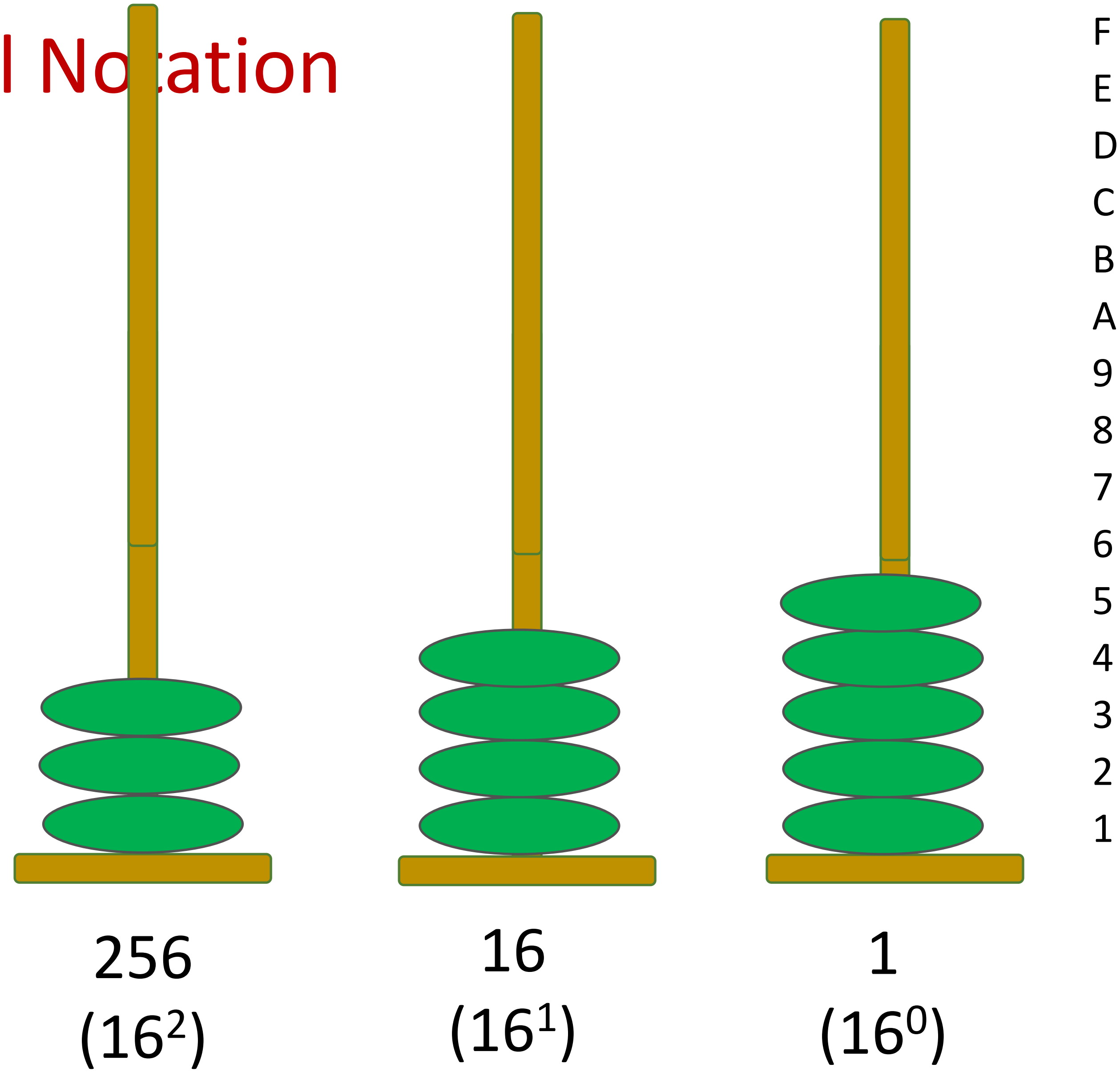


Must be 255

Positional Notation



Positional Notation



Others To decimal

- 3 4 5

10^2 10^1 10^0

→ $3*10^2 + 4*10^1 + 5*10^0 =$
300 + 40 + 5= 345

Decimal 328

- 1 0 1

2^2 2^1 2^0

→ $1*2^2 + 0*2^1 + 1*2^0 =$
4 + 0 + 1 = 5

0000000000101

Binary

- 3 4 5

8^2 8^1 8^0

→ $3*8^2 + 4*8^1 + 5*8^0 =$
192 + 32 + 5 = 229

Octal 345

- 3 4 5

16^2 16^1 16^0

→ $3*16^2 + 4*16^1 + 5*16^0 =$
3*256 + 4*16 + 5 *1 =
768 + 64 + 5 = 837

Hex 345

You should know these conversions.

- 3 H
- 3 G 5

Non-Decimal Number Systems

Number can be in any base, 2, 3, 4, …8, 10, 13, 15, 16 …..

Decimal is base 10 and has 10 digit symbols:

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

Binary is base 2 and has 2 digit symbols:

0,1

Octal is base 8 and has 8 digit symbols:

0,1,2,3,4,5,6,7

Hexdecimal is base 16 and has 16 digit symbols:

0,1,2,3,4,5,6,7,8,9, A,B,C,D,E,F

base 5 and has ? digit symbols

0,1,2,3,4

base 7 and has ? digit symbols 0,1,2,3,4,5,6

For a value to exist in a given base, it can only contain the digits in that base, which range from 0 up to (but not including) the base.

[0 ~ base-1]

What bases can these values be in? 122, 198, 178, G1A4

Representing Numbers

Positional representation

- Decimal (base 10)
- Binary (base 2)
- Octal (base 8)
- Hexadecimal (base 16)
- Conversion between different bases
  - Binary, hex, oct to decimal (recap)
  - Binary to/from hex and octal
  - Decimal to binary, hex, octal

Signed binary representation – 2’s complement

- 2’s complement (binary) to decimal
- Decimal to 2’s complement (binary)

Binary representation of fractions

- Binary to decimal
- Decimal to binary

Positional Notation

An n-digit unsigned integer in base b

(d_i is the digit in the i^{th} position in the number)

The number: $(\, d_{n-1} \;\; d_{n-2} \ldots d_i \ldots \;\; d_1 \;\; d_0 \,)_b$

Its value =

$$d_{n-1} \times b^{n-1} + d_{n-2} \times b^{n-2} + \ldots + d_i \times b^{i} \ldots + d_1 \times b^{1} + d_0 \times b^{0}$$

Converting Binary to Decimal

What is the decimal equivalent of the binary number 1101110?

$$
\begin{aligned}
& 1 \times 2^6 = 1 \times 64 = 64 \\
+\ & 1 \times 2^5 = 1 \times 32 = 32 \\
+\ & 0 \times 2^4 = 0 \times 16 = 0 \\
+\ & 1 \times 2^3 = 1 \times 8 = 8 \\
+\ & 1 \times 2^2 = 1 \times 4 = 4 \\
+\ & 1 \times 2^1 = 1 \times 2 = 2 \\
+\ & 0 \times 2^0 = 0 \times 1 = 0 \\
& = 110 \text{ in base 10}
\end{aligned}
$$

Converting Octal to Decimal

What is the decimal equivalent of the octal number 642?

642_8

$$
\begin{aligned}
& 6 \times 8^2 = 6 \times 64 = 384 \\
+ & 4 \times 8^1 = 4 \times 8 = 32 \\
+ & 2 \times 8^0 = 2 \times 1 = 2 \\
& = 418 \text{ in base } 10
\end{aligned}
$$

Converting Hexadecimal to Decimal

What is the decimal equivalent of the hexadecimal number DEF?

DEF_{16}

$$\begin{aligned} & D \times 16^2 = 13 \times 256 = 3328 \\ + & E \times 16^1 = 14 \times 16 = 224 \\ + & F \times 16^0 = 15 \times 1 = 15 \\ & = 3567 \text{ in base } 10 \end{aligned}$$

Remember, the digit symbols in base 16 are
0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F

No cal in test, but simpler…

Decimal number	Binary representation	Octal representation	Hexadecimal representation
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F
16	10000	20	10

$a=16 \quad a=10000_2 \quad a=20_8 \quad a=10_{16}$

$a=76 \quad a=1001100_2 \quad a=114_8 \quad a=4C_{16}$

Positional Notation

An n-digit unsigned integer in base b

(d_i is the digit in the i^{th} position in the number)

The number: $(d_{n-1}\ d_{n-2} \ldots d_i \ldots d_1\ d_0)_b$

Its value =

$$d_{n-1} \times b^{n-1} + d_{n-2} \times b^{n-2} + \ldots + d_i \times b^i \ldots + d_1 \times b^1 + d_0 \times b^0$$

Examples:

$(642)_{10} = 6 \times 10^2 + 4 \times 10^1 + 2 \times 10^0 = 642$

$(5073)_8 = 5 \times 8^3 + 0 \times 8^2 + 7 \times 8^1 + 3 \times 8^0 = 2619$

$(1011)_2 = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 11$

$(3214)_5 = \qquad = 434$

Representing Numbers

Positional representation

- Decimal (base 10)
- Binary (base 2)
- Octal (base 8)
- Hexadecimal (base 16)
- Conversion between different bases
  - Binary, hex, oct to decimal (recap)
  - Binary to/from hex and octal
  - Decimal to binary, hex, octal

Signed binary representation – 2’s complement

- 2’s complement (binary) to decimal
- Decimal to 2’s complement (binary)

Binary representation of fractions

- Binary to decimal
- Decimal to binary

Large Numbers

255 is the largest integer that can be represented in 8 bits, but larger values can be represented with more bits.

For example, 27916 is represented by:

0110110100001100

Can you remember the Binary representation?

Short Forms for Binary

Because large numbers require long strings of Binary digits, short forms have been developed to help deal with them.

An early system used was called Octal

It's based on the 8 patterns in 3 bits

Later system used was called Hexadecimal,

It's based on the 16 patterns in 4 bits

Binary to/from others -- why Hex and Octal

I want an int with representation 0011011010000110 , how to denote it?

Inconvenient to write 0011011010000110

- 1 1 0 1 1 0 1 0 0 0 0 1 1 0

2^{13} 2^{12} 2^{11} 2^{10} 2^9 2^8 2^7 2^6 2^5 2^4 2^3 2^2 2^1 2^0

8192 4096 1024 512 128 4 2 = 13958 Decimal

Easier way?

Short Forms for Binary - Octal

111	7
110	6
101	5
100	4
011	3
010	2
001	1
000	0

0 0 1 1 0 1 1 0 1 0 0 0 0 1 1 0

Can be shortened by dividing the number into 3-bit chunks, starting from the least significant bit and replacing each with a single Octal digit

Short Forms for Binary - Octal

111	7
110	6
101	5
100	4
011	3
010	2
001	1
000	0

000011011010000110

0 3 3 2 0 6

Inserted to make a complete group of 3 bits

0 3 3 2 0 6

Binary to Octal Examples

Try these:

111100_2

100101_2

111001_2

1100101_2

Hint: when the leftmost group has fewer than three digits, fill with zeroes from the left:

1100101 = 1 100 101 = 001 100 101

110011101

Binary to Octal Examples

Try these:

$111100_2 = 74_8$

$100101_2 = 45_8$

$111001_2 = 71_8$

$1100101_2 = 145_8$

Hint: when the leftmost group has fewer than three digits, fill with zeroes from the left:

1100101 = 1 100 101 = 001 100 101

$110011101_2 = 635_8$

Short Forms for Binary - Hexadecimal

0111	7	1111	F
0110	6	1110	E
0101	5	1101	D
0100	4	1100	C
0011	3	1011	B
0010	2	1010	A
0001	1	1001	9
0000	0	1000	8

It was later determined that using base 16 and 4 bit patterns would be more efficient

2 hex digits = 8 bits (1 byte)

Since there are only 10 numerical digits, 6 letters were added to complete the set of hexadecimal digits

Short Forms for Binary - Hexadecimal

0111	7	1111	F
0110	6	1110	E
0101	5	1101	D
0100	4	1100	C
0011	3	1011	B
0010	2	1010	A
0001	1	1001	9
0000	0	1000	8

0 0 1 1 0 1 1 0 1 0 0 0 0 1 1 0

can be short-formed by dividing the number into 4-bit chunks -- starting from the least significant bit (LSB) -- and replacing each with a single Hexadecimal digit.

0 0 1 1 0 1 1 0 1 0 0 0 0 1 1 0

Short Forms for Binary - Hexadecimal

0111	7	1111	F
0110	6	1110	E
0101	5	1101	D
0100	4	1100	C
0011	3	1011	B
0010	2	1010	A
0001	1	1001	9
0000	0	1000	8

0011 0110 1000 0110

3 6 8 6

Number base notations:

3686_{16}

$(3686)_{16}$

Used in this course, math literature

3686h

3686_{hex}

Used in computer science articles

0x3686

Some prog. languages

Binary to/from others -- why Hex and Octal

"I want an int with representation `0011011010000110`, how to code it in C?"

Inconvenient to write **11011010000110**

- **1 1 0 1 1 0 1 0 0 0 0 1 1 0**

  2^{13} 2^{12} 2^{11} 2^{10} 2^{9} 2^{8} 2^{7} 2^{6} 2^{5} 2^{4} 2^{3} 2^{2} 2^{1} 2^{0}

  8192+4096+1024+512+128+4+2= 13958

  → Decimal 13958_{10}

- $\underbrace{\mathtt{1\ 1}}_{3}\ \underbrace{\mathtt{0\ 1\ 1}}_{3}\ \underbrace{\mathtt{0\ 1\ 0}}_{2}\ \underbrace{\mathtt{0\ 0\ 0}}_{0}\ \underbrace{\mathtt{1\ 1\ 0}}_{6}$

  ↔ Octal 33206_{8} easier

- $\underbrace{\mathtt{1\ 1}}_{3}\ \underbrace{\mathtt{0\ 1\ 1\ 0}}_{6}\ \underbrace{\mathtt{1\ 0\ 0\ 0}}_{8}\ \underbrace{\mathtt{0\ 1\ 1\ 0}}_{6}$

  ↔ Hex 3686_{16} easier

You should know these conversions (both ways).

Binary to/from others -- why Hex and Octal

I want an int with representation 01001100, how to denote it?

Inconvenient to write **`01001100`**

- 0 1 0 0 1 1 0 0 ⟷ $1*2^6 + 1*2^3 + 1*2^2 =$ Decimal

  $2^7\ 2^6\ 2^5\ 2^4\ 2^3\ 2^2\ 2^1\ 2^0$ ⟷ $64 + 8 + 4$ **76**

- $\underbrace{0\ 1}_{1}\ \underbrace{0\ 0\ 1}_{1}\ \underbrace{1\ 0\ 0}_{4}$ ⟷ $\mathbf{114}_8$ Octal

- $\underbrace{0\ 1\ 0\ 0}_{4}\ \underbrace{1\ 1\ 0\ 0}_{12\ --\ C}$ ⟷ $\mathbf{4C}_{16}$ $\mathbf{4c}_{16}$ Hex

You should know these conversions (both ways).

Binary to Hexadecimal Examples

Try these:

11011100_2

10110101_2

10011001_2

110110101_2

Hint: when the leftmost group has fewer than four digits, fill with zeroes on the left:

110110101 = 1 1011 0101 = 0001 1011 0101

1101001011101_2

Binary to Hexadecimal Examples

Try these:

$11011100_2 = DC_{16}$

$10110101_2 = B5_{16}$

$10011001_2 = 99_{16}$

$110110101_2 = 1B5_{16}$

Hint: when the leftmost group has fewer than four digits, fill with zeroes on the left:

110110101 = 1 1011 0101 = 0001 1011 0101

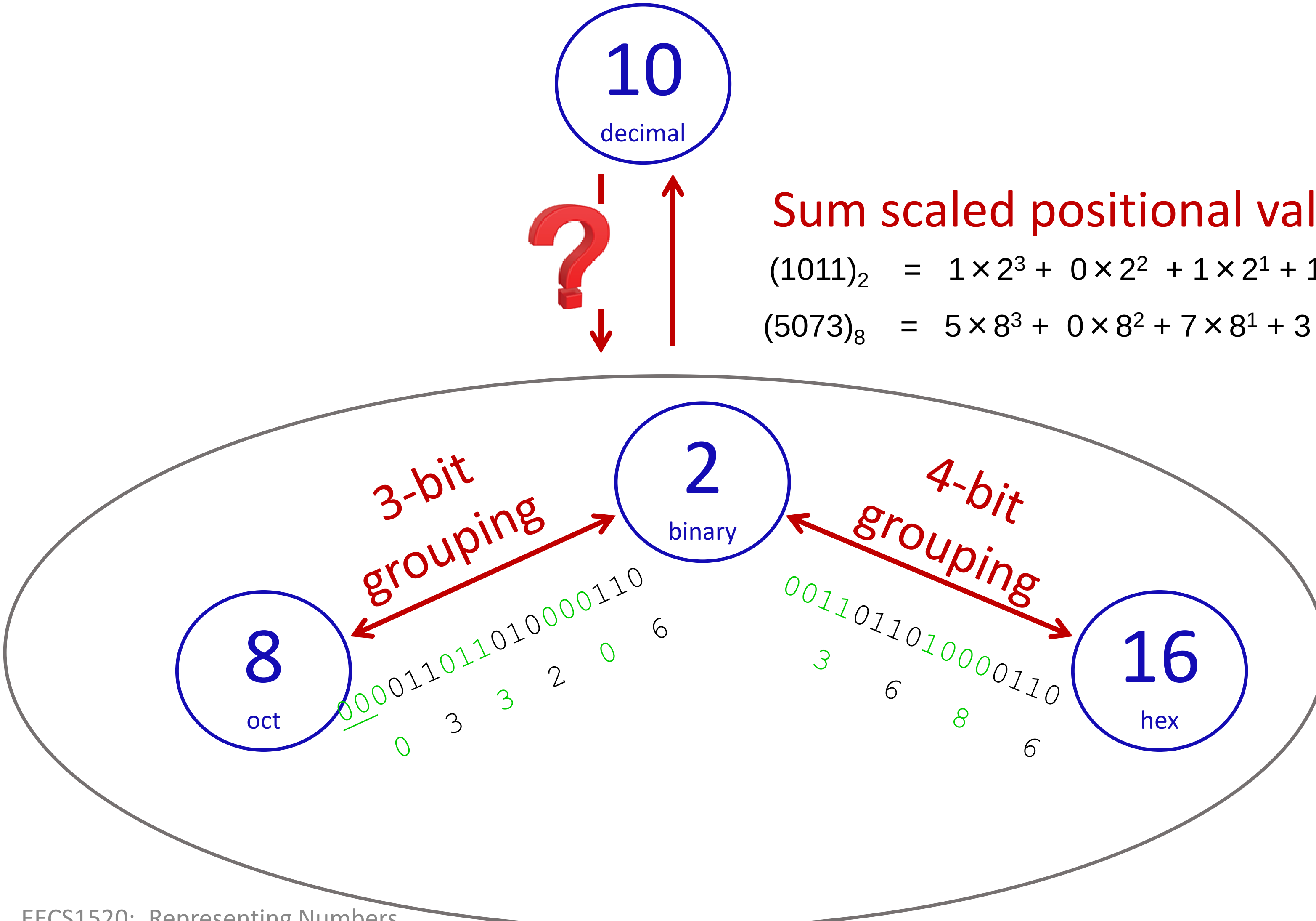
$1101001011101_2 = 1A5D_{16}$

Summary: base conversion methods

Sum scaled positional values

$$(1011)_2 = 1\times 2^3 + 0\times 2^2 + 1\times 2^1 + 1\times 2^0 = 11$$

$$(5073)_8 = 5\times 8^3 + 0\times 8^2 + 7\times 8^1 + 3\times 8^0 = 2619$$



Representing Numbers

Positional representation

- Decimal (base 10)
- Binary (base 2)
- Octal (base 8)
- Hexadecimal (base 16)
- Conversion between different bases
  - Binary, hex, oct to decimal (recap)
  - Binary to/from hex and octal
  - Decimal to binary, hex, octal

Signed binary representation – 2's complement

- 2's complement (binary) to decimal
- Decimal to 2's complement (binary)

Binary representation of fractions

- Binary to decimal
- Decimal to binary

How to extract the Least Significant Digit

$$(d_{n-1}d_{n-2}\cdots d_1\, d_0)_b$$

$$= d_{n-1} * b^{n-1} + d_{n-2} * b^{n-2} + \cdots + d_1 * b^1 + d_0 * b^0$$

$$= (d_{n-1} * b^{n-2} + d_{n-2} * b^{n-3} + \cdots + d_1 * b^0) * b + d_0$$

$$= (d_{n-1}d_{n-2}\cdots d_1\,)_b \;\; * \;\; b \;\; + \;\; d_0$$

One less digit

Example: $(402213)_5 \;=\; (40221)_5 \;\times 5 \;\; + \;\; 3$

Example: $(402213)_{10} \;=\; (40221)_{10} \times 10 \;\; + \;\; 3$

Example: $(40221)_{10} \;=\; (4022)_{10} \times 10 \;\; + 1$

Divide base, remainder gives the right most digit.

(quotient) Divide base, remainder gives the 2nd right most digit.

(new quotient) Divide base, remainder gives the 3rd right most digit.

....

Converting Decimal to Other Bases

Divide base, remainder gives the right most digit.
Divide base, remainder gives the 2nd right most digit.
Divide base, remainder gives the 3rd right most digit.
....

Algorithm for converting number in base 10 to other bases (e.g., base b)

While (the quotient is not zero)
    Divide the decimal number by the new base b.
    Make the remainder the next digit to the left in the answer.
    Replace the original decimal number with the quotient.

$$(d_{n-1}d_{n-2}\cdots d_1\, d_0)_b \qquad \text{(divide by b)}$$

$$= \underbrace{(d_{n-1}d_{n-2}\cdots d_1)_b}_{\textit{Quotient}} \times b + \underbrace{d_0}_{\textit{Remainder}}$$

Converting Decimal to Binary

What is 9 (base 10) in binary (base 2)?

Try it!

What is 37 (base 10) in binary (base 2)?

Try it!

Converting Decimal to Binary

Quotient	*Remainder*	*(division by 2)*
9		(9 = 2*4 + 1)
4	*1*	
2	*0*	
1	*0*	
0	*1*	

1 0 0 1

$(9)_{10} = (1\ 0\ 0\ 1)_2$

Let's check by converting back

Converting Decimal to Binary

Quotient	*Remainder*	*(division by 2)*
37		(37 = 2*18 + 1)
18	*1*	
9	*0*	
4	*1*	
2	*0*	
1	*0*	
0	*1*	

1 0 0 1 0 1

$$(37)_{10} = (100101)_2$$

Let's check by converting back

Converting Decimal to Octal

What is 43 (base 10) in base 8?

Try it!

What is 1988 (base 10) in base 8?

Try it!

Converting Decimal to Octal

Quotient	*Remainder*	*(division by 8)*
43		(43 = 8*5 + 3)
5	*3*	
0	*5*	

5 3

$(43)_{10} = (53)_8$

Let's check by converting back

Converting Decimal to Octal

Quotient	*Remainder*	*(division by 8)*
1988		(1988 = 8*248 + 4)
248	*4*	
31	*0*	
3	*7*	
0	*3*	

3 7 0 4

$(1988)_{10} = (3704)_8$

Let's check online

Converting Decimal to Hexadecimal

What is 43 (base 10) in base 16?

Try it!

What is 1988 (base 10) in base 16?

Try it!

What is 3567 (base 10) in base 16?

Try it!

Converting Decimal to Hexadecimal

Quotient	*Remainder*	*(division by 16)*
43		(43 = 16*2 + 11)
2	*11 = B*	
0	*2*	

2 B

(43) $_{10}$ = (2B) $_{16}$

Let's check by converting back

Converting Decimal to Hexadecimal

Quotient	*Remainder*	*(division by 16)*
1988		(1988 = 16*124 + 4)
124	*4*	
7	*12 = C*	
0	*7*	

7 C 4

$(1988)_{10} = (7C4)_{16}$

Let's check online

Converting Decimal to Hexadecimal

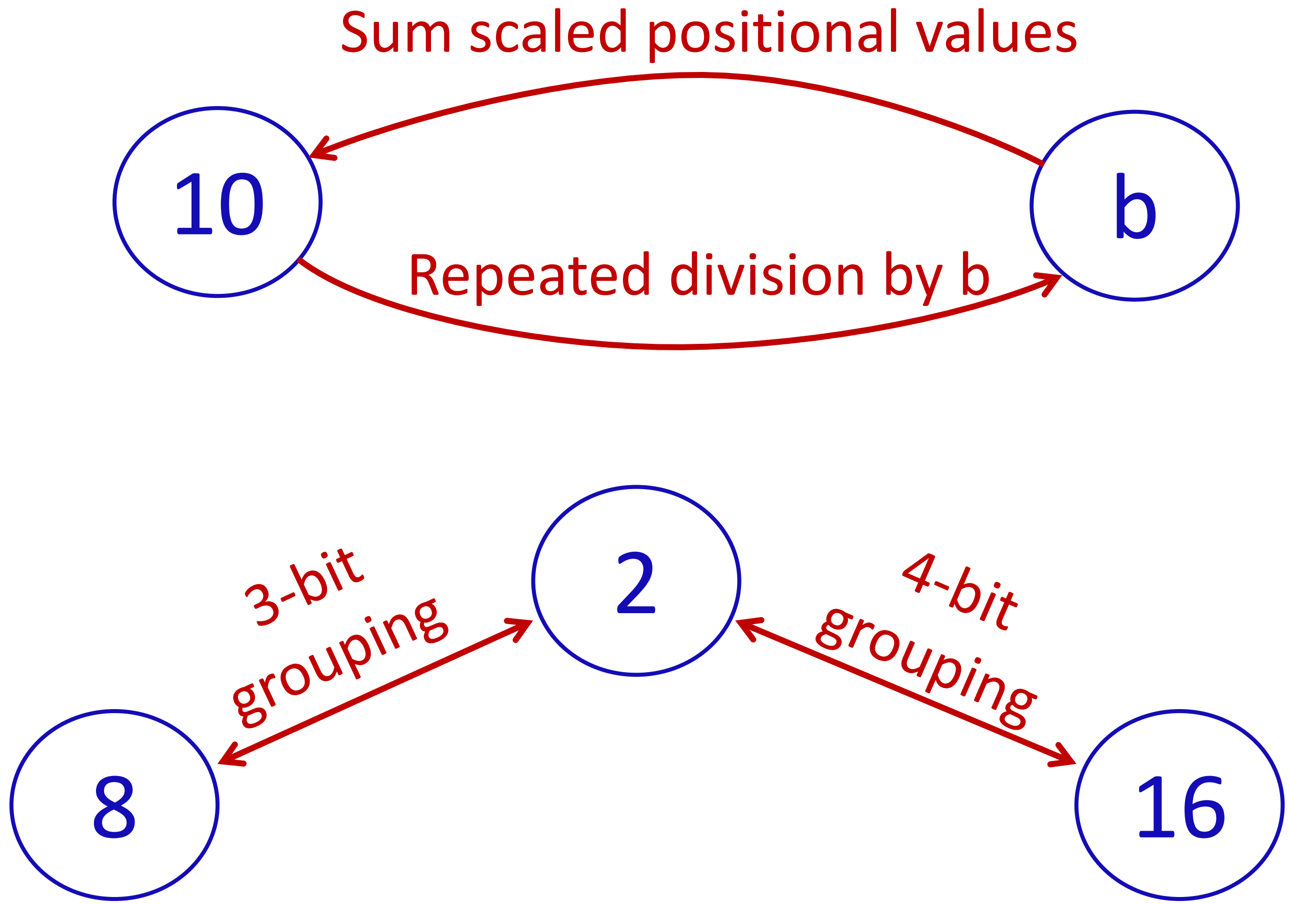
Quotient	*Remainder*	*(division by 16)*
3567		(3567 = 16*222 + 15)
222	*15 = F*	
13	*14 = E*	
0	*13 = D*	

D E F

$$(3567)_{10} = (\text{DEF})_{16}$$

Did earlier other way
Exercise for you
Let's check online

Summary: base conversion methods



Summary: base conversion methods

10
decimal

Repeated division by base
List remainders ← ← ←

Sum scaled positional values

$(5073)_8 = 5 \times 8^3 + 0 \times 8^2 + 7 \times 8^1 + 3 \times 8^0 = 2619$

$(1011)_2 = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 11$

2
binary

3-bit grouping

8
oct

000011011010000110

0 3 3 2 0 6

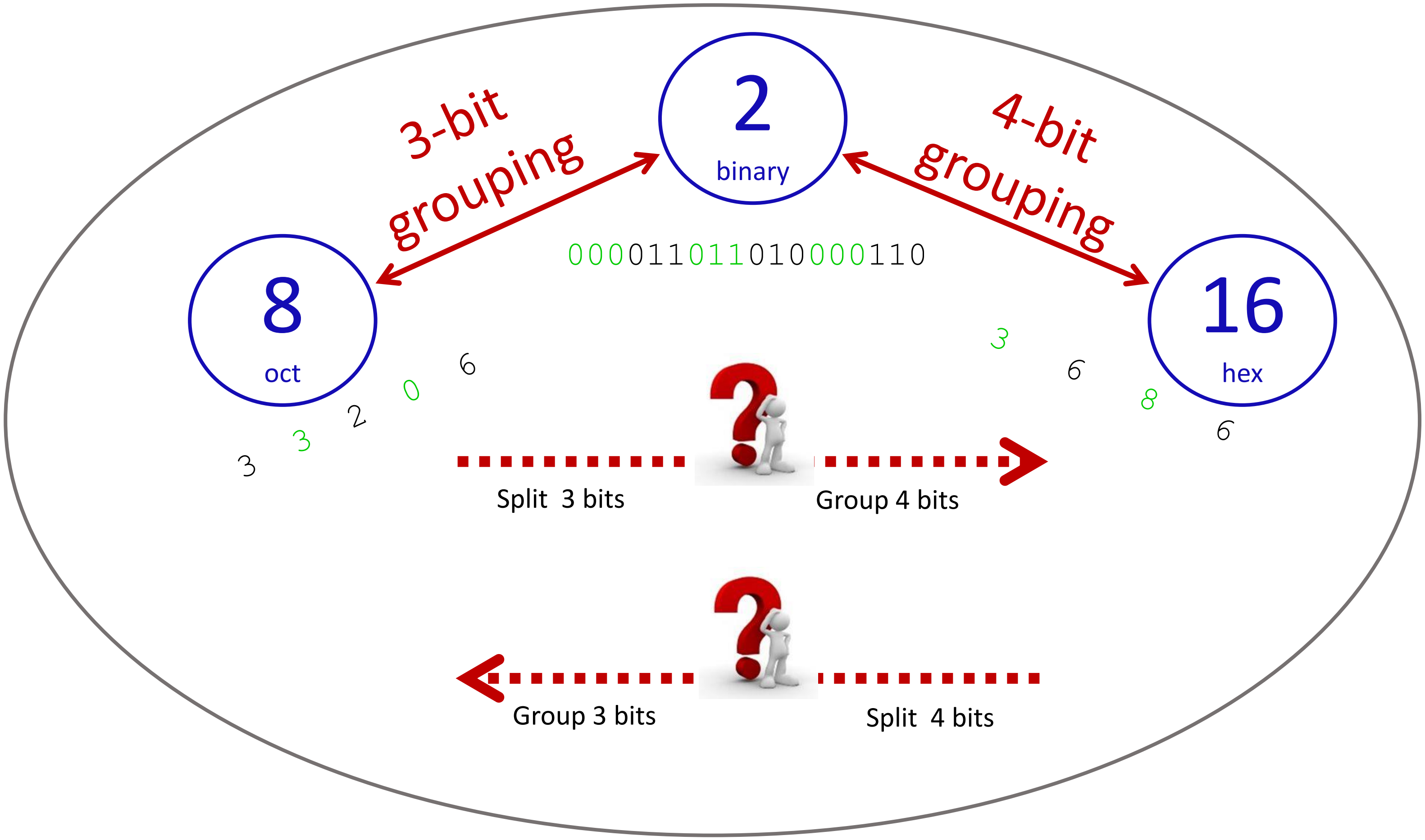
4-bit grouping

16
hex

0011011010000110

3 6 8 6

Extra: Oct <-> Hex



Representing Numbers

Positional representation

- Decimal (base 10)
- Binary (base 2)
- Octal (base 8)
- Hexadecimal (base 16)
- Conversion between different bases
  - Binary, hex, oct to decimal (recap)
  - Binary to/from hex and octal
  - Decimal to binary, hex, octal

Signed binary representation – 2’s complement

- 2’s complement (binary) to decimal
- Decimal to 2’s complement (binary)

Binary representation of fractions

- Binary to decimal
- Decimal to binary

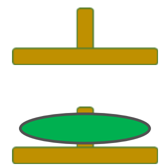
- So far, assume all bits are values. For n bits/poles, how many different values? Max and min value?

total min max

- One bit/pole

0

1



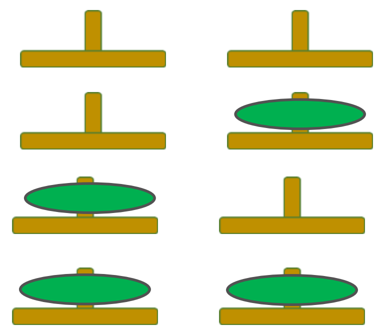
- Two bits/poles

0 0

0 1

1 0

1 1

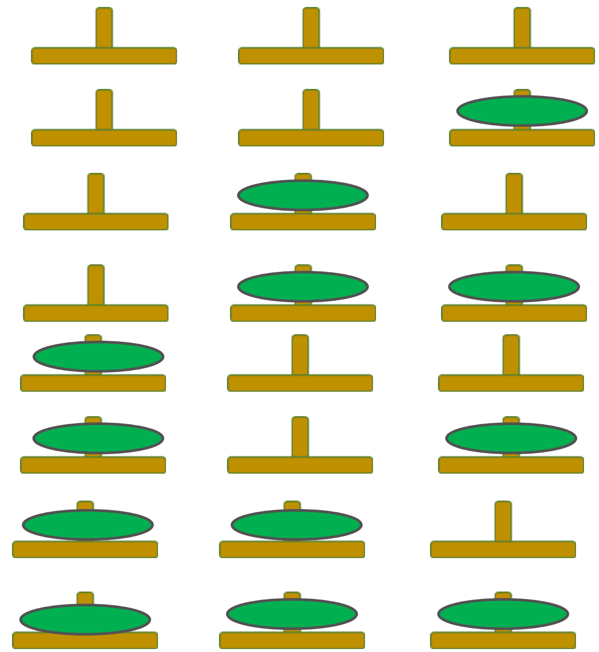


- 3 bits/poles?

- So far, assume all bits are values. For n bits/poles, how many different values? Max and min value?

- 3 bits/poles

0 0 0
0 0 1
0 1 0
0 1 1
1 0 0
1 0 1
1 1 0
1 1 1



total	min	max
8	0	7

- 7 bits/poles?
- 8 bits/poles?
- 32 bits/poles?

- So far, assume all bits are values. For n bits/poles, how many different values? Max and min value?

	total	min	max	
• 7 bits/poles	2^7 = 128 values.	0000000	1111111	0~127
• 8 bits/poles	2^8 = 256 values.	00000000	11111111	0~255
• 32 bits/poles	2^{32} = 4,294,967,296 values.	000…0	111…1	0~4,294,967,295
• n bits/poles	2^n values.	0	~ 2^n -1	

How To Create a Negative Number

In digital electronics you cannot simply put a minus sign in front of a number to make it negative.

You must represent a negative number in a *fixed-length* binary number system. All signed arithmetic must be performed in a *fixed-length* number system.

A physical *fixed-length* device (usually memory) contains a fixed number of bits (usually 4-bits, 8-bits, 16-bits) to hold the number.

Integers: Ordinary Signed Representation

- Leftmost bit represents sign (0 = "+", 1 = "–").
  The remaining bits represent an ordinary binary number.

sign bit

$\boxed{0}$ 1 0 1 = + 5

1 1 0 1 = – 5

0 0 0 0 = + 0

1 0 0 0 = – 0

"1's complement"

- 0 has two different representations.
- How to eliminate this wasted redundancy and ambiguity?

Integers: Ordinary Signed Representation

- Let's find -7 and -12 using Ordinary Signed Representation in an 8-bit binary system

8-bit binary system

sign bit	2^6	2^5	2^4	2^3	2^2	2^1	2^0	
0	0	0	0	0	1	1	1	= + 7
1	0	0	0	0	1	1	1	= - 7
0	0	0	0	1	1	0	0	= + 12
1	0	0	0	1	1	0	0	= - 12

Integers: Ordinary Signed Representation

- Now let's add +7 and -7 using Ordinary Signed Representation in an 8-bit binary system

$$
\begin{array}{rcccccccc}
 & & & & 1 & 1 & 1 & & \\
 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\
+ & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\
\hline
 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0
\end{array}
\qquad\qquad
\begin{array}{rr}
 & 7 \\
+ & (-7) \\
\hline
 & 0
\end{array}
$$

As you can see the result is not zero and that cannot be correct

- ✗ This method is not accurate if we need to perform operations such as addition or subtraction
- ✗ 0 has two different representations

How to eliminate this wasted redundancy, inaccuracy and ambiguity?

2’s Complement Notation

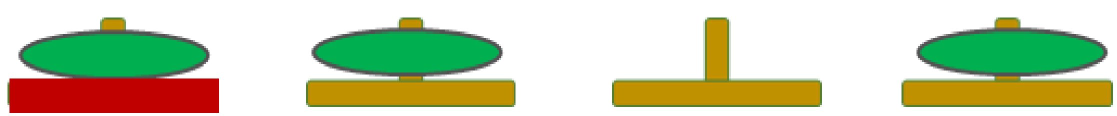
- 2’s complement is one possible solution.
- The leftmost bit represents not only sign, but **negative** positional value.

3	2	1	0	(bit positions)
-2^3	2^2	2^1	2^0	(positional values)
1	**1**	**0**	**1**	**(the number)**

$$= -2^3 * \mathbf{1} + 2^2 * \mathbf{1} + 2^1 * \mathbf{0} + 2^0 * \mathbf{1}$$

$$= -8 + 4 + 0 + 1$$

$$= -3$$

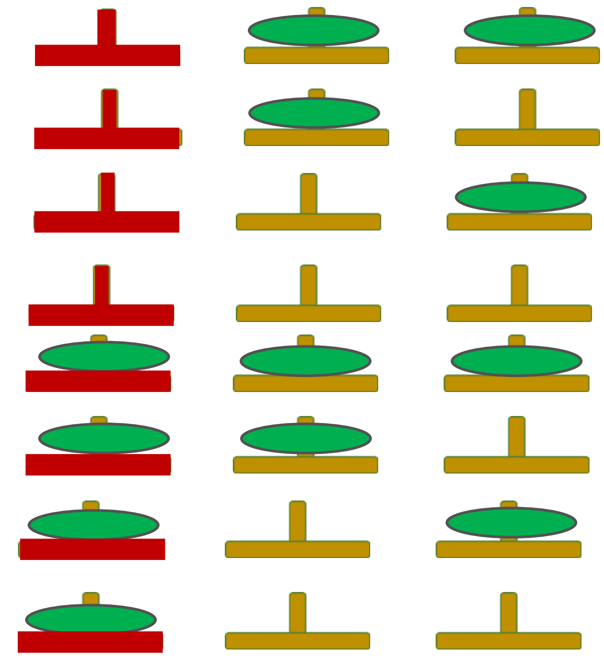


2's Complement Notation

n bits

e.g., $n = 2$

MAX	011	=	+3	$2^{n-1} - 1$
	010	=	+2	
	001	=	+1	4 values
	000	=	0	
	111	=	− 1	
	110	=	− 2	
	101	=	− 3	4 values
MIN	100	=	− 4	-2^{n-1}



- 3 bits/poles 2^3 values. $0 \sim 2^3-1$
- 3 bits --2's complement 2^3 values. $-2^{3-1} \sim 2^{3-1}-1$

2's Complement Notation

n bits   e.g., $n = 8$

MAX	01111111	$=$	$+2^{n-1} - 1$	$= +127$
	01111110			
	…			
	00000010	$=$	$+2$	
	00000001	$=$	$+1$	
	00000000	$=$	0	
	11111111	$=$	-1	
	11111110	$=$	-2	
	…			
MIN	10000000	$=$	-2^{n-1}	$= -128$

128 values (00000000 to 01111111)

128 values (11111111 to 10000000)

- 8 bits/poles 2^8 values. $0 \sim 2^8 - 1$
- 8 bits --2's complement 2^8 values. $-2^{8-1} \sim 2^{8-1} - 1$

Signed vs. unsigned, summary

Assume all n bits are magnitudes.

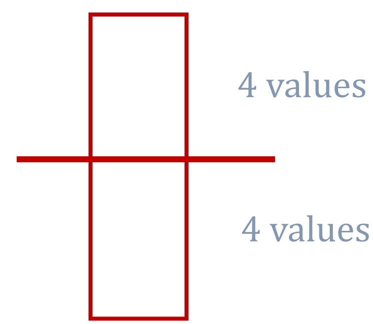
- Max value: 1111111….11111
- Min value: 0000000….00000
- How many values: 2^n
- Range: $0 \sim 2^n-1$

Assume 2's complement.

- Max value: 0111111….11111
- Min value: 1000000….00000
- How many values: 2^n
- Range: $-2^{n-1} \sim 0 \sim 2^{n-1}-1$

half half

3 bits, 2's complement, max and min value?



4 bits ? 8 bits ?

2's Complement (binary) → Decimal

Be given the # of bits

- If MSB is 0 → decimal value is positive
  - Convert from binary to decimal as normal   0110 →
- If MSB is 1 → decimal value is negative   1101 →
  - Negative value definition   (approach 1)
  - Take the remaining (or all) bits
  - Flip the bits (i.e., 0→1, 1→0)   (approach 2)
  - Add 1
  - Convert from binary to decimal as normal, remember the sign
- For example (assume 4 bits):
  - 0110 → 6
  - 1101 → −((0010) +1) → −(0011) → −3
  - 1001 →
  - 11100110 →

Decimal → 2's Complement (binary)

- If decimal is positive, convert to binary (26, how?)
- If decimal is negative
  - Convert absolute value to binary (26, how?)
  - Flip the bits (i.e., 0→1, 1→0)
  - Add 1
  - Set the MSB to be 1

$1010\ ? \xrightarrow{\text{unsigned}} 10$

$1010\ ? \xrightarrow{\text{signed}} -6$

Be given
the # of bits

- For example:
  - 6 → 0110 assume 4 bits
  - −6 → (6→0110) → 1001+1 → 1010 assume 4 bits

    Not 10
  - -3 (3→0011) → 1101 assume 4 bits
  - -7 (7→0111) → 1001
  - -17 assume 8 bits (17→00010001) 11101111
  - -26 assume 8 bits (26→ 00011010) 11100110
  - -37 assume 8 bits (37→00100101) 11011011

2's Complement Examples assume 8 bits

Example #1 -5 ?

```
 5 = 00000101
     ↓↓↓↓↓↓↓↓
     11111010
           +1
     --------
-5 = 11111011
```

Complement Digits

Add 1

Example #2 -13?

```
 13 = 00001101
      ↓↓↓↓↓↓↓↓
      11110010
            +1
      --------
-13 = 11110011
```

Complement Digits

Add 1

Interpretations of Binary Patterns

4 bits, max min?

unsigned | signed

Binary	Decimal (All values)	Hexadecimal	2's Complement (left most is sign)
1111	15	F	−1
1110	14	E	−2
1101	13	D	−3
1100	12	C	−4
1011	11	B	−5
1010	10	A	−6
1001	9	9	−7
1000	8	8	−8
0111	7	7	7
0110	6	6	6
0101	5	5	5
0100	4	4	4
0011	3	3	3
0010	2	2	2
0001	1	1	1
0000	0	0	0

8

8

Arithmetic in 2's Complement
(remember it's a *fixed length* system)

00 + 00 = 00

00 + 01 = 01

01 + 00 = 01

01 + 01 = 10

−3	in 2's complement		11111101
+1	in 2's complement	+	00000001
-2			11111110

10 + -3 (111101) -5 (111011) + 2 -127 (10…1) +1

29 (00011101) - 23 (0010111) 23-29

Arithmetic in 2's Complement
(remember it's a *fixed length* system)

00 + 00 = 00

00 + 01 = 01

01 + 00 = 01

01 + 01 = 10

-2	in 2's complement		11111110
$+5$	in 2's complement	+	00000101
3	discard the carry bit	1	00000011

10 + -3 (111101) -5 (111011) + 2 -127 (10…1) +1

29 (00011101) - 23 (0010111) 23-29

Arithmetic in 2's Complement

(remember it's a *fixed length* system)

00 + 00 = 00

00 + 01 = 01

01 + 00 = 01

01 + 01 = 10

−1	in 2's complement		11111111
+1	in 2's complement	+	00000001
0	discard the carry bit	1	00000000

10 + -3 (111101)   -5 (111011) + 2   -127 (10…1) +1

29 (00011101) - 23 (0010111)   23-29

POS + NEG → POS Answer

Take the 2’s complement of the negative number and use regular binary addition.

$$\begin{array}{r} 9 \\ +\ (-5) \\ \hline 4 \end{array} \qquad \begin{array}{r} 00001001 \\ +\ 11111011 \\ \hline \cancel{1}]00000100 \end{array}$$

8th Bit = 0: Answer is Positive
Disregard 9th Bit

$$\left.\begin{array}{r} 00000101 \\ \downarrow\downarrow\downarrow\downarrow\downarrow\downarrow\downarrow\downarrow \\ 11111010 \\ +1 \\ \hline 11111011 \end{array}\right\}$$

2’s Complement Process

POS + NEG → NEG Answer

Take the 2’s complement of the negative number and use regular binary addition.

```
   (-9)         11110111
+    5        + 00000101
-------       ----------
   -4           11111100
```

$\boxed{4}$

8^{th} Bit = 1: Answer is Negative

To Check: Perform 2’s Complement On Answer

```
  11111100
  00000011
        +1
----------
  00000100
```

2’s Complement Process

```
  00001001
  11110110
        +1
----------
  11110111
```

NEG + NEG → NEG Answer

Take the 2's complement of both negative numbers and use regular binary addition.

$$\begin{array}{rcr} (-9) & \longrightarrow & 11110111 \\ +\ (-5) & \longrightarrow & +\ 11111011 \\ \hline -\boxed{14} & \longleftarrow & \cancel{1}]\,11110010 \end{array}$$

2's Complement Numbers, See Conversion Process In Previous Slides

8^{th} Bit = 1: Answer is Negative
Disregard 9^{th} Bit

To Check: Perform 2's Complement On Answer

$$\begin{array}{r} 11110010 \\ \downarrow\downarrow\downarrow\downarrow\downarrow\downarrow\downarrow\downarrow \\ 00001101 \\ +1 \\ \hline 00001110 \end{array}$$

POS + NEG → POS Answer

Take the 2’s complement of the negative number and use regular binary addition.

$$\begin{array}{r} +7 \\ +\ (-7) \\ \hline 0 \end{array} \qquad \begin{array}{r} 00000111 \\ +\ 11111001 \\ \hline \cancel{1}]00000000 \end{array}$$

Disregard 9th Bit

$$\left.\begin{array}{r} 00000111 \\ \downarrow\downarrow\downarrow\downarrow\downarrow\downarrow\downarrow\downarrow \\ 11111000 \\ +1 \\ \hline 11111001 \end{array}\right\}$$

2’s Complement Process

As you can see with using 2’s complement, the result is zero which is correct

Number Overflow

If each value is stored using 8 bits, then 126 + 3 overflows:

Max value? -128 ~ 127

```
   01111110
+  00000011
   10000001
```

Apparently, 126 + 3 is −127. This overflow error is the result of trying to represent an infinite range in a finite one

Even present-day 64-bit computers cannot operate with all integers because only finitely many of them can be represented as 64-bit binary numbers

Representing Numbers

Positional representation

- Decimal (base 10)
- Binary (base 2)
- Octal (base 8)
- Hexadecimal (base 16)
- Conversion between different bases
  - Binary, hex, oct to decimal (recap)
  - Binary to/from hex and octal
  - Decimal to binary, hex, octal

Signed binary representation – 2's complement

- 2's complement (binary) to decimal
- Decimal to 2's complement (binary)

Binary representation of fractions

- Binary to decimal
- Decimal to binary

Positional Notation extended to fractions

The number: $(\underbrace{d_{n-1}\ d_{n-2}\ \ldots\ d_1\ d_0}_{\text{integral/whole part}}\ .\ \underbrace{d_{-1}\ d_{-2}\ d_{-3}\ldots}_{\text{fractional part}})_b$

decimal/radix point

Its value =

$d_{n-1} \times b^{n-1} + d_{n-2} \times b^{n-2} + \ldots + d_1 \times b^1 + d_0 \times b^0$
$+\ d_{-1} \times b^{-1} + d_{-2} \times b^{-2} \times\ d_{-3} \times b^{-3}\ \ldots$

Examples:

$(64.27)_{10} = 6 \times 10^1 + 4 \times 10^0 + 2 \times 10^{-1} + 7 \times 10^{-2} = 64.27$

$(50.73)_8 = 5 \times 8^1 + 0 \times 8^0 + 7 \times \underbrace{8^{-1}}_{0.125} + 3 \times \underbrace{8^{-2}}_{0.015625} = 40.921875$

$(10.11)_2 = 1 \times 2^1 + 0 \times 2^0 + 1 \times \underbrace{2^{-1}}_{0.5} + 1 \times \underbrace{2^{-2}}_{0.25} = 2.75$

Fractions: Binary $\longrightarrow$ Decimal

$$(1\quad 0\quad .\quad 1\quad 1\,)_2 = 2^1 + 2^{-1} + 2^{-2} = 2.75$$

$$\begin{array}{cccc} \uparrow & \uparrow & \uparrow & \uparrow \\ 2^1 & 2^0 & 2^{-1} & 2^{-2} \\ 2 & 1 & 0.5 & 0.25 \end{array}$$

Note:

Integer/whole part: $(1\quad 0\,)_2 = 2$

Fractional part: $(.\ 1\quad 1\,)_2 = .75$

These can each be converted separately.

$(1\,0\,0\,1\;.\;0\;1\,)_2 = 9.25$

$(1\,0\,1\,0\;.\;1\;1\;1)_2 = 10.875$

$(1\,0\,1\,1\;.\;1\;0\;1)_2 = 11.625$

Fractions: Binary ⟵ Decimal

Example 1:

$(2\,.\,7\,5)_{10} \quad = \quad (\,?\,)_2$

$(2\,)_{10} \quad = \quad (\,1\,0\,)_2$ [repeated division by 2, described earlier]

$(.\,7\,5)_{10} \quad = \quad (\,.\;1\;\;1\;)_2$ [repeated multiplication by 2, described next]

.75 * 2 = **1** . 5 [extract integer part: 0/1, repeat on **fraction**]

.5 * 2 = **1** . 0 [stop when fraction becomes 0,
or you run out of available bit positions]

$\mathbf{(2\,.\,7\,5)_{10} \quad = \quad (\,1\,0\,.\;\;1\;\;1\;)_2}$

Decimal to binary

37.75 100101.11

Integral part

Keep on: ÷ base

List: Remainder ← ←

New: Quotient

Stop: when quotient is 0

Fractional part

Keep on: × base

List: Integral part → →

New: Fractional part

Stop: when fractional is 0
or, run out of bits

20.25 10100.01

20.625

Fractions: Decimal to Binary

Example 2: $(37.25)_{10} = (??.??)_2$

37/2 = 18 **1**
18/2 = 9 **0**
9/2 = 4 **1**
4/2 = 2 **0**
2/2 = 1 **0**
1/2 = 0 **1**

[repeated division by 2]

1 0 01 0 1

.25 * 2 = **0** .5
.5 * 2 = **1**.0

[repeated multiplication by 2

0 1

100101.01

Verify: $(100101.01)_2 = 32 + 4 + 1 + 0 * 2^{-1} + 1 * 2^{-2} = 37 + 0.25$

37.75 10.875 11.625

Fractions: Decimal to Binary

Example 3: $(.3285)_{10} = (\,.\,??????\,)_2$

.3285 * 2 = **0** .657 → **0**
.657 * 2 = **1** .314 → **1**
.314 * 2 = **0** . 628 → **0**
.628 * 2 = **1** . 256 → **1**
.256 * 2 = **0** . 512 → **0**
.512 * 2 = **1** . 024 → **1**

fraction .024 not 0 yet → **may round up**

$(.3285)_{10}$ is between $(\,.010101\,)_2$ and $(\,.010110\,)_2$

Verify: $(\,.010101\,)_2 = 2^{-2} + 2^{-4} + 2^{-6} = .328125\ (closer!)$
$(\,.010110\,)_2 = 2^{-2} + 2^{-4} + 2^{-5} = .34375$

2.3 .???? assume 4 bits 10.0100 (2.25)
10.0101 2.3175

Scientific Notation

- Very large and very small numbers are often represented such that their order of magnitude can be compared.
  - E.g., an electron's mass is about 0.00000000000000000000000000000091093826 kg.
    - In scientific notation, this is written $9.1093826 \times 10^{-31}$ kg.
  - E.g., Earth's mass is about 5,973,600,000,000,000,000,000,000 kg.
    - In scientific notation, this is written 5.9736×10^{24} kg.

- In general, any number can be expressed as follows:

$$a \times b^{e}$$

where e is an integer, and a is a real number, and b is a natural number > 1

0.00137	1.37×10^{-3}
15237	1.5237×10^{4}
59000005	5.9000005×10^{7}
123.025	1.23025×10^{2}
0.00005025	5.025×10^{-5}

Representing Real Numbers in Decimal

For your information

A real value in base 10 can be defined by the following formula where the exponent is an integer

$$\text{sign} \times \text{mantissa} \times 10^{\text{exponent}}$$

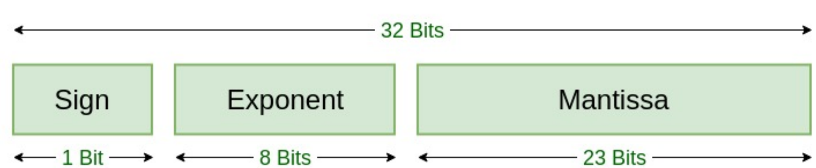
This representation is called **floating point** because the radix point "floats" left or right, depending on the value of *exponent*

Representing floating point in computers is considered beyond this course. If you are interested, search IEEE 754 in the internet

- In a 64-bit floating point number, only 53 bits are used for mantissa
- Lower precision if you use f.p. numbers when integer ones could also work

The idea of these slides was to show you that all different types of numbers can be represented in computers by using only 0 and 1

yet in some cases not as accurate as they are in real world



How many bits is enough?

For your information

- Recording Earth coordinates with 1 cm precision
  - $\log_2$(40 000 000 m *100 cm/m) ~ 32 bits per coordinate

log2(40 000 000 * 100) =
31.897352854

- Jeff Bezos' wealth (2019) with $0.01 precision
  - $\log_2$($112.4 billion * 100) ~ 44 bits

log2(112.4 billion * 100) =
43.3537072691

- Human mass with 100 *g* precision
  - $\log_2$(635 * 10) ~ 13 bits
  - 635 *kg* was the highest mass of a person ever, according to Google

log2(635 * 10) =
12.6325408765

Outside air temperature with 0.1 degree precision
  - $\log_2$(10*(58−(−88)) ~ 11 bits

- Generally, you need $\log_2 10$ (~3.32) binary digits for every decimal one

Representing Numbers

Representing Texts

Representing Audio

Representing Images and graphics

Representing Videos

Representing Numbers

LET'S RECAP...

Positional representation

- Decimal (base 10)
- Binary (base 2)
- Octal (base 8)
- Hexadecimal (base 16)
- Conversion between different bases
  - Binary, hex, oct to decimal
  - Binary to/from hex and octal (special relations)
  - Decimal to binary, hex, octal

Signed binary representation – 2's complement

- 2's complement (binary) to decimal
- Decimal to 2's complement (binary)

Binary representation of fractions

- Binary to decimal
- Decimal to binary

Binary, Oct and Hex → Decimal

Give the Decimal equivalent of the following

- Binary $00111010_2 = (\qquad)_{10}$
- Octal $137_8 = (\qquad)_{10}$
- Hex $13C_{16} = (\qquad)_{10}$
- $132_4 = (\qquad)_{10}$
- $122_3 = (\qquad)_{10}$

Binary → Oct and Hex

Given binary representation 00111010_2, how to represent it in Oct and Hex?

- Octal ($)_8$
- Hex ($)_{16}$
- ($)_4$

Oct and Hex → Binary

Given Oct representation 73_8 , what is the binary representation?

00111011

Given Hex representation 73_{16}, what is the binary representation?

01110011

Given Hex representation $4B_{16}$, what is the binary representation?

01001011

Given representation 322_4, what is the binary representation

Decimal → Binary, Oct and Hex

Given decimal 37_{10}, how to represent it in Binary, Oct and Hex? How many base2, base8, base16 digits

- Binary
- Octal
- Hex
- Base 4
- Base 5

2's complement (binary) ←→ decimal

- 0110 → 6   assume 4 bits 2's complement
- 1001 → −((0110) +1) → −(0111) → −7   assume 4 bits 2's complement
- 1101 → −((0010) +1) → −(0011) → −3   assume 4 bits 2's complement
- 11100110 → - (00011001 +1) → - (00011010) → -26   assume 8 bits

---

- 5 → 0101   assume 4 bits
- −6 → (6→0110) → 1001+1 → 1010   assume 4 bits
- -3   (3→0011) →   assume 4 bits
- -7   (7→0111) →
- -17   assume 8 bits
- -26   assume 8 bits
- -37   assume 8 bits

- Assume 8 bits, do the following calculation in binary

3 + 4

3+1

3+5

3+7

10 - 3 (11111101)

- 5 (11111011) + 2

- 127 (10…1) +1

29 (00011101) - 23 (00010111)

23-29

Factions ←→ decimal

$(\mathbf{10.111})_{\mathbf{2}} \;=\; (\;.\;??????\;)_{\mathbf{10}}$ 2.875

$(\mathbf{100101.11})_{\mathbf{2}} \;=\; (\;.\;??????\;)_{\mathbf{10}}$ 37.75

$(\mathbf{37.75})_{\mathbf{10}} \;=\; (\;.\;??????\;)_{\mathbf{2}}$ 100101.11

$(\mathbf{20.25})_{\mathbf{10}} \;=\; (\;.\;??????\;)_{\mathbf{2}}$ 10100.01

$(\mathbf{20.625})_{\mathbf{10}} \;=\; (\;.\;??????\;)_{\mathbf{2}}$ 10100 .101

Valid or not?

- $(1234)_{10}$
- $(87120)_{8}$
- $(12AB)_{16}$
- $(3AB)_{12}$
- $(525)_{5}$
- $(1010101)_{10}$

Signed vs. unsigned, summary

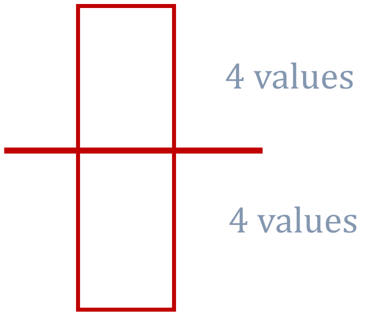
Assume all n bits are magnitudes.

- Max value: 1111111….11111
- Min value: 0000000….00000
- How many values: 2^n
- Range: $0 \sim 2^n - 1$

Assume 2's complement.

- Max value: 0111111….11111
- Min value: 1000000….00000
- How many values: 2^n
- Range: $-2^{n-1} \sim 0 \sim 2^{n-1} - 1$

3 bits, 2's complement, max and min value?



4 bits ? 8 bits ?

Representing Numbers

Representing Texts

Representing Audio

Representing Images and graphics

Representing Videos