Data representation

How do we represent "stuff"?

Example: file my.c

```
%emacs my.c
main () {
   float f = 3.14;
   printf ("%f\n", f);
}
%cc my.c
%a.out
```

What is the representation of "3.14"?

- in my.c: character (ASCII)
- in a.out (file or RAM): floating point (IEEE 754 standard)

What is the representation of "\n"?

- in my.c: 2 characters (ASCII)
- in a.out (file or RAM): integer
- in output: cursor moves to next line

In general, it's all bits!

- What they mean depends on the interpretation

Data representation

"All your base are belong to us."
- The Computer Organization

Number bases

Positional representation of a number

Base (radix) k, digits d_i
Digit values: 0, 1, . . . (k-1)
Number of digits: n

weight	$k^{(n-1)}$		k^2	k^1	k^0
digits	$d_{(n-1)}$	…	d_2	d_1	d_0

$$\text{number} = \sum_{i=0}^{i=n-1} (d_i * k^i)$$

Focus on integers first, then extend to fractions

Number bases

Human number system: base 10

Base (radix) k = 10, digits d_i
Assume number of digits: 5

weight	$10^{(n-1)}$	10^3	10^2	10^1	10^0
digits	3	7	2	9	4

$$\text{number} = \sum_{i=0}^{i=n-1} (d_i * k^i)$$

$$= (3 * 10000) + (7 * 1000) + (2 * 100) + (9 * 10) + (4 * 1)$$

$$= 37294_{ten}$$

Converting number bases

Computer number system: base 2 (binary)

Base (radix) k = 2, digits d_i = (0, 1)

Given: 13_{ten}

Find: binary representation

	b_5	b_4	b_3	b_2	b_1	b_0
weight	2^5	2^4	2^3	2^2	2^1	2^0
digit	0	0	1	1	0	1

8 + 4 + 1 = 13_{ten}

	result	remainder		try it!
start	13			
13/2	6	1	b_0	
6/2	3	0	b_1	
3/2	1	1	b_2	
1/2	0	1	b_3	
0/2	0	0	b_4	
0/2	0	0	b_5	

Why does this work?

$13_{ten} = b_5 * 32 + b_4 * 16 + b_3 * 8 + b_2 * 4 + b_1 * 2 + b_0$

$(2 * 6 + 1)_{ten} = 2 * (b_5 * 16 + b_4 * 8 + b_3 * 4 + b_2 * 2 + b_1) + b_0$

Therefore, dividing by 2 gives a result of the expression in parentheses and a remainder of b_0

$6_{ten} = b_5 * 16 + b_4 * 8 + b_3 * 4 + b_2 * 2 + b_1$

$= 2 * (b_5 * 8 + b_4 * 4 + b_3 * 2 + b_2) + b_1$

Dividing by 2 again gives a remainder of b_1 and so forth.

3_{ten} = $b_5 * 8 + b_4 * 4 + b_3 * 2 + b_2$

$= 2 * (b_5 * 4 + b_4 * 2 + b_3) + b_2$

1_{ten} = $b_5 * 4 + b_4 * 2 + b_3$

$= 2 * (b_5 * 2 + b_4) + b_3$

0_{ten} $= b_5 * 2 + b_4$

What do all higher bits have to be?

Converting number bases

Given: 50_{ten}
Find: base 3 representation

	d_5	d_4	d_3	d_2	d_1	d_0
weight	243	81	27	9	3	1
digit	0	0	1	2	1	2

	result	remainder
start	50	
50/3	16	2
16/3	5	1
5/3	1	2
1/3	0	1
	0	0
	0	0

Converting number bases

Given: 325_{ten}
Find: base 16 (hexadecimal) representation

	d_3	d_2	d_1	d_0
weight	4096	256	16	1
digit	0	1	4	5

	result	remainder
start	325	
325/16	20	5
20/16	1	4
1/16	0	1
	0	0

Converting number bases

Given: binary representation
Find: base 8 (octal) representation

value: 101010111010100110$_{two}$

Could convert to base 10, then convert to base 8
However, what is the pattern of digit values?

binary:	**64**	**32**	**16**	**8**	**4**	**2**	**1**
octal:	**64**			**8**			**1**

Group binary digits:

101	010	111	010	100	110
5	2	7	2	4	6

Converting number bases: fractions

Given: decimal fraction representation
Find: binary representation

value: 0.375

		b_{-1}	b_{-2}	b_{-3}	b_{-4}	b_{-5}
weight		0.5	0.25	0.125	0.0625	0.03125
digit	0.	0	1	1	0	0

To convert a whole number, we successively divided by 2
What should we do with a fraction?

$0.375_{ten} = b_{-1} * 2^{-1} + b_{-2} * 2^{-2} + b_{-3} * 2^{-3} + b_{-4} * 2^{-4} + b_{-5} * 2^{-5}$

If we multiply by 2,

$0.750 = b_{-1} * 2^{0} + b_{-2} * 2^{-1} + b_{-3} * 2^{-2} + b_{-4} * 2^{-3} + b_{-5} * 2^{-4}$

Notice that the result is still less than 1, so b_{-1} must be 0, since we assume that all digits are positive
Multiply by 2 again,

$1.50 = b_{-2} * 2^{0} + b_{-3} * 2^{-1} + b_{-4} * 2^{-2} + b_{-5} * 2^{-3}$

Now we know that b_{-2} must be 1, since the rest of the expression is a fraction:

$0.50 = b_{-3} * 2^{-1} + b_{-4} * 2^{-2} + b_{-5} * 2^{-3}$

Repeat the process:

$1.00 = b_{-3} * 2^{0} + b_{-4} * 2^{-1} + b_{-5} * 2^{-2}$

$0.00 = b_{-4} * 2^{-1} + b_{-5} * 2^{-2}$

What are all the bits to the right?

0

Converting number bases: repeating fractions

Given: decimal fraction representation
Find: binary representation

value: 0.27

		b_{-1}	b_{-2}	b_{-3}	b_{-4}	b_{-5}
weight		0.5	0.25	0.125	0.0625	0.03125
digit	0.	0	0	1	1	0

Sometimes, a fraction may not be exactly represented:

1/3 = 0.3333 . . .

$0.2_{ten} = b_{-1} * 2^{-1} + b_{-2} * 2^{-2} + b_{-3} * 2^{-3} + b_{-4} * 2^{-4} + b_{-5} * 2^{-5}$

If we multiply by 2,

$0.4 = b_{-1} * 2^{0} + b_{-2} * 2^{-1} + b_{-3} * 2^{-2} + b_{-4} * 2^{-3} + b_{-5} * 2^{-4}$

Multiply by 2 again,

$0.8 = b_{-2} * 2^{0} + b_{-3} * 2^{-1} + b_{-4} * 2^{-2} + b_{-5} * 2^{-3}$

$1.6 = b_{-3} * 2^{0} + b_{-4} * 2^{-1} + b_{-5} * 2^{-2}$

$0.6 = b_{-4} * 2^{-1} + b_{-5} * 2^{-2}$

$1.2 = b_{-4} * 2^{0} + b_{-5} * 2^{-1}$

$0.2 = b_{-5} * 2^{-1}$

At this point, we are back to the original value 0.2, so the digits must repeat.

$0.27_{ten} = 0.00110011\ldots_{two}$

Converting number bases: any float

Given: decimal float representation
Find: binary representation

value: 13.375

	b_3	b_2	b_1	b_0		b_{-1}	b_{-2}	b_{-3}
weight	8	4	2	1		0.5	0.25	0.125
digit	1	1	0	1	.	0	1	1
				13	.	375		

Integer part and fractional part are independent.

1. Convert integer
2. Convert fraction
3. Add together