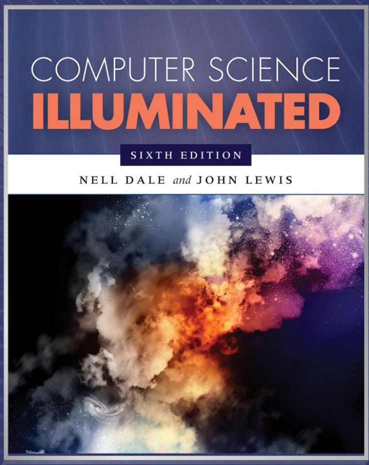
Data Representation

Part 2



Chapter Goals

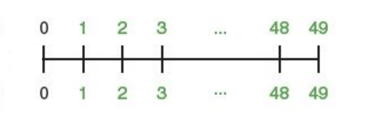
- Distinguish between analog and digital information
- Explain the binary formats for negative and floating-point values
- Describe the characteristics of the ASCII and Unicode character sets

Number Representation

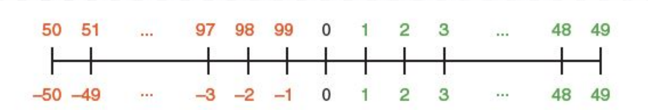
- Remember last lecture?
- Represent a number with a fixed **word length**
- This is the limitation of mapping the continuous to the discrete space.
- Need fancy ways to represent increasingly nuanced number types.

Number Representation

- Natural numbers are straightforward
  - The word = the (approximated) value



- Signed numbers are trickier (but only slightly):
  - First half of map represents natural numbers.
  - Second half represents the negative form.



Representing Real Numbers

Real numbers are numbers with a whole part and a fractional part (either of which may be zero)

104.32
0.999999
357.0
3.14159

In decimal, positions to the right of the decimal point are the tenths, hundredths, thousandths, etc.:

10^{-1}, 10^{-2} , 10^{-3} …

Representing Real Numbers

Same rules apply in binary as in decimal

Radix point is general term for “decimal point”

Positions to the right of the radix point in binary:

2^{-1} (halves position),
2^{-2} (quarters position),
2^{-3} (eighths position)
…

Representing Real Numbers

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

$$\text{sign} \times \text{mantissa} \times \text{base}^{\text{exp}}$$

This representation is called **floating point** because the **radix point** "floats"

In analogy to the fixed number of bits that computers use to represent integers, we'll treat the mantissa as having a fixed number of digits

Representing Real Numbers

Why? $\text{sign} \times \text{mantissa} \times \text{base}^{\text{exp}}$

Instead of representing real number as:

3.1415

We can use natural numbers, and shift a radix:

= +1 * 31415 * 10^{-4}
= 31415 * 10^{-4}
= 3.1415

Representing Real Numbers

$$\text{sign} \times \text{mantissa} \times \text{base}^{\text{exp}}$$

- Sign: Indicator of positive or negative state
- Mantissa: Significant digits of the number being represented
- Base: Number of unique digits in the representation system
- Exponent: Scales our number by shifting position of our radix

Representing Real Numbers

$$\text{sign} \times \text{mantissa} \times \text{base}^{\text{exp}}$$

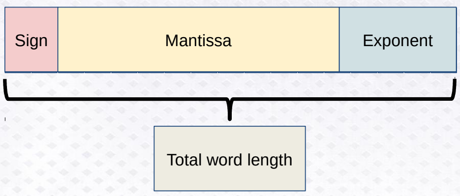
Say we have the decimal number: 123.456

What is this in above representation scheme?

Sign = 0 (positive)
Mantissa = 123456
Base = 10 (decimal)
Exponent = -3 (move radix left 3 positions)

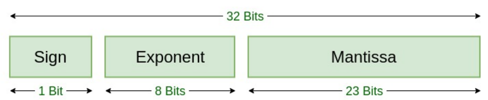
Representing Real Numbers

- Representation as a word:

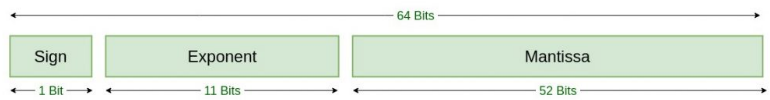


Representing Real Numbers

- IEEE 754 Standard:



Single Precision
IEEE 754 Floating-Point Standard



Double Precision
IEEE 754 Floating-Point Standard

https://www.geeksforgeeks.org/ieee-standard-754-floating-point-numbers/

Real Numbers in Base X

Remember our method for calculating an integer number with positional notation:

```
While (value is not zero)
    Divide value by the new base
    Store remainder
    Replace value with the quotient
Read remainders in reverse order
```

A similar method holds for the fraction component in a real number...

Real Numbers in Base X

To calculate the value of the fractional part, we now want to scale up (due to the negative exponent)

```
While (value is not zero OR precision is reached)
    Multiply value by the new base
    Store the whole part
    Replace value with the fractional part of the result
Read whole parts in order
```


Real Numbers in Binary

Consider converting 3.625 in base 10 to binary:

- For the whole part (3), use the **division** and **remainder** method:

  3 / 2 = 1, remainder: 1
  1 / 2 = 0, remainder: 1

  Read in reverse: 11

Real Numbers in Binary

Consider converting 3.625 in base 10 to binary:

- For the fractional part (0.625), use the **multiplication** and **whole** method:

0.625 * 2 = 1.25, whole part: 1
0.25 * 2 = 0.5, whole part: 0
0.5 * 2 = 1.0, whole part: 1

Read in order: 101

Real Numbers in Binary

Consider converting 3.625 in base 10 to binary:

- Whole part (3) = 11
- Fractional part (0.625) = 101

- Therefore, 3.625 (base 10) in binary:

11.101

Representing Real Numbers

TABLE 3.1 Values in decimal notation and floating-point notation (five digits)

Real Value	Floating-Point Value
12001.00	$12001 * 10^{0}$
–120.01	$-12001 * 10^{-2}$
0.12000	$12000 * 10^{-5}$
–123.10	$-12310 * 10^{-2}$
155555000.00	$15555 * 10^{4}$

Fundamentally, the floating-point used by computers is very similar,
but uses complicated

tricks to represent more
numbers and improve
efficiency

Representing Real Numbers

Scientific notation

A form of floating-point representation in which the decimal point is kept to the right of the leftmost digit

12001.32708 is 1.200132708E+4 in scientific notation (E+4 is how computers display $x10^4$)

What is 123.332 in scientific notation?

What is 0.0034 in scientific notation?

Representing Text

What must be provided to represent text?

The number of characters to represent is finite (phew!), so list them all and assign each a binary string

Character set

A list of characters and the codes used to represent each one

Computer manufacturers agreed to standardize

The ASCII Character Set

ASCII stands for American Standard Code for Information Interchange

ASCII originally used seven bits to represent each character, allowing for 128 unique characters

Later extended ASCII evolved so that all eight bits were used

How many characters could be represented?

ASCII Character Set Mapping

Left Digit(s) \ Right Digit	ASCII 0	1	2	3	4	5	6	7	8	9
0	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT
1	LF	VT	FF	CR	SO	SI	DLE	DC1	DC2	DC3
2	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS
3	RS	US	☐	!	"	#	$	%	&	'
4	(	)	*	+	,	–	.	/	0	1
5	2	3	4	5	6	7	8	9	:	;
6	<	=	>	?	@	A	B	C	D	E
7	F	G	H	I	J	K	L	M	N	O
8	P	Q	R	S	T	U	V	W	X	Y
9	Z	[	\	]	^	_	`	a	b	c
10	d	e	f	g	h	i	j	k	l	m
11	n	o	p	q	r	s	t	u	v	w
12	x	y	z	{	|	}	~	DEL		

FIGURE 3.5 The ASCII character set

The ASCII Character Set

The first 32 characters in the ASCII character chart do not have a simple character representation to print to the screen

What do you think they are used for?

The Unicode Character Set

Extended ASCII is not enough for international use

One Unicode mapping uses 16 bits per character

How many characters can this mapping represent?

The first 256 characters correspond exactly to the extended ASCII character set

The Unicode Character Set

Code (Hex)	Character	Source
0041	A	English (Latin)
042F	Я	Russian (Cyrillic)
0E09	ฉ	Thai
13EA	Ꮺ	Cherokee
211E	℞	Letterlike symbols
21CC	⇌	Arrows
282F	⠯	Braille
345F	㑟	Chinese/Japanese/Korean (common)

FIGURE 3.6 A few characters in the Unicode character set

The Unicode Character Set

Text Representation

- All these Unicode characters require 16 bits to represent
- Consider
  - average PhD thesis of 100,000 words
  - average word length of 5 characters
  - 8,000,000 bits without whitespace!
- Actually, that's only 1 megabyte 😄