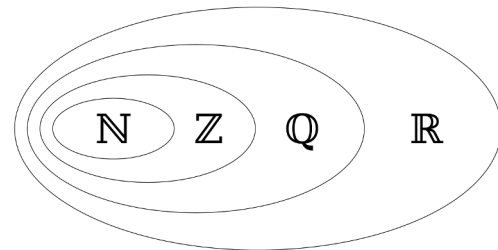


Numbering Systems Review



- Natural numbers: Things that we count and the base system
 - base 60: from the Sumerians: used for time, angles, geographical coordinates
 - base 20: e.g., the Mayan system
 - base 12: Jan, Feb, ... Dec
 - base 10: Jan, Feb, ... June, Sept, Oct, Nov, Dec!
 - base 7: Sun, Mon, Tues, ... Sat, Sun
 - base 2, 8 (2^3), 16 (2^4)
 - unary: 110, 11110, 11111111111110, 111011011110=324
- Integers: Positive, Zero, and Negative
- Rational: e.g., $\frac{1}{3}$, 2.3 but not... π
- Real Numbers: e.g., π
- Complex Numbers: e.g., $3 + 2i$ where $i = \sqrt{-1}$
- The concept of both $-\infty$ and ∞ , $x/0 == \text{NaN}$

0	1	2	3	4
	•	••	•••	••••
5	6	7	8	9
10	11	12	13	14
15	16	17	18	19

Different Representations for "42.25"

- Decimal:

- 42.25
- 10#42.25 (Bash shell)
- 4.225×10^1
- 4.225E01 (calculators)

- Hexadecimal:

- 0x2A.4
- 16#2A.4
- 2.A4 $\times 16^1$

- Octal:

- 052.2 (Java, C, etc, but not Javascript)
- 0o52.2 (Javascript)
- 8#52.2
- 5.22 $\times 8^1$

- Binary:

- 0b101010.01
- 2#101010.01
- 1.0101001×2^5

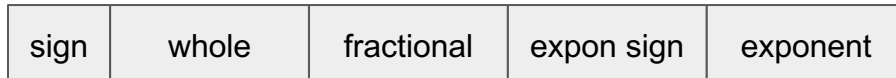
- In COMP122:

- Use context
- Allow spacing for clarity
- Allow for separators
- Allow for signs
- E.g., A UTF-8 3, byte sequence

| 1110 - 1010 | 10 – 10 0101 | 10 – 11 0111 |

Computer Limitations and Representation

- Recall the Universal Computer
 - There is a limited tape size to perform calculation
- Recall the von Neumann and Harvard architecture
 - There is a predefined width to registers and memory
- Abstract representations with limited sizes for:
 - Natural Numbers & Zero: unsigned char, unsigned int
 - Integers: short int, int, long int, long long int
 - Rational/Real:
 - Fix Point ---
 - Floating Point float (single), double
- An encoding of each will include one or more of the following:



+ 4.225 x10⁺²
+1.010101 x2⁺¹⁰¹

Expanded Notation

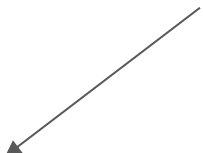
- Recall from grade school

1234 =	1×10^3	$1 * 1000 = 1000$
	$+ 2 \times 10^2$	$2 * 100 = 200$
	$+ 3 \times 10^1$	$3 * 10 = 30$
	$+ 4 \times 10^0$	$4 * 1 = 4$
		1234

thousands	hundreds	tens	ones
1	2	3	4

Expanded Notation for other Bases

Radix Point



BASE	Columns					
Base 16	4096 's	256 's	16 's	1 's	1/16	1/256
Base 10	1000 's	100 's	10 's	1 's	1/10	1/100
Base 8	512 's	64 's	8 's	1 's	1/8	1/64
Base 2	8 's	4 's	2 's	1 's	1/2	1/4

Expanded Notation

- Recall from grade school

154 =	1×10^2	100
	$+ 5 \times 10^1$	50
	$+ 4 \times 10^0$	4
		154

Expanded Notation

- Base 16
16# 9A

Base 10 Value		
0x9A =	9 x 16 ¹	9 x 16 = 144
	+ A x 16 ⁰	10 x 1 = 10
		154

- Base 8
0o232
8# 232

Base 10 Value		
0232 =	2 x 8 ²	2 * 64 = 128
	+ 3 x 8 ¹	3 * 8 = 24
	+ 2 x 8 ⁰	2 * 1 = 2
		154

10	A
11	B
12	C
13	D
14	E
15	F

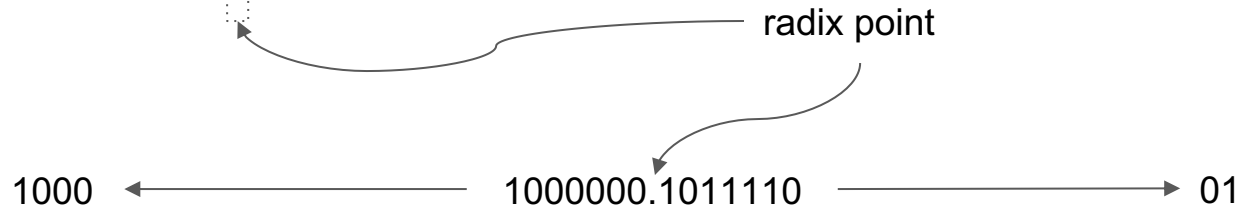
Expanded Notation

- Base 2: 2_{10} 1001 1010

1001 1010 =	1×2^7	128
	$+ 0 \times 2^6$	0
	$+ 0 \times 2^5$	0
	$+ 1 \times 2^4$	16
	$+ 1 \times 2^3$	8
	$+ 0 \times 2^2$	0
	$+ 1 \times 2^1$	2
	$+ 0 \times 2^0$	0
		154

Real Numbers

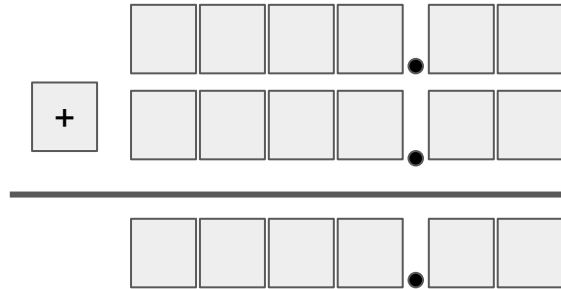
- 45.34
- - 45.34
- 1011110.0100
- - 1011110.0100



- But what about my limitations on the computer!

Fixed Point Numbers

- There is a need to represent rational numbers
- Consider operations on US currency. (dollars and cents)
 - Place the two digits after the decimal (radix) point
 - Provide a register of size 6



Note:

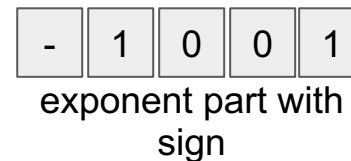
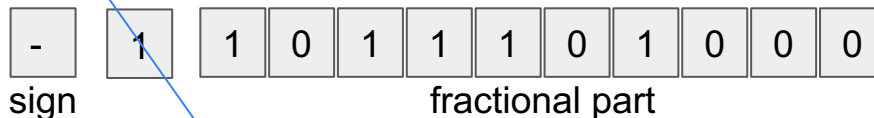
We can't represent standard price of a gallon of gas! \$4.96 9/10

sign	whole	fractional	expon sign	exponent
------	-------	------------	------------	----------

Scientific Notation

$$\begin{array}{r}
 14.3 \times 10^7 \\
 - \quad \frac{9.2 \times 10^7}{5.1 \times 10^7}
 \end{array}$$

- All numbers represented as: $m \times 10^N$
- Simplifies operations on large and small numbers.
 - Distance between sun and earth: $92,000,000 = 9.2 \times 10^7$
 - Distance between sun and mars: $143,000,000 = 1.43 \times 10^8$
- Floating point representation
 - a representation of scientific notation
 - introduces the notion of infinity, and NaN ($0 / 0 = ?$, $0 \times \text{infinity} = ?$)
- Representation of: $-1.1011101 \times 2^{-1001}$
 - Assume a size of 16
 - Note the whole part is always "1"!



Floating Point Encoding

Original number: 2# - 0.000100101

Recall Scientific Notation: $-1.00101 \times 2^{-100} // 4$

always 1: so we don't store it

- Components to Encode
 - sign: negative
 - coefficient: "1.00101" and the mantissa: "00101"
 - exponent: - 100
 - Issue: negative exponents
 - Solution: store values with a bias
- Bias:
 - Shift all numbers along the number line by N
 - Typically it is half the range:
 - 3 bits -> 011 == 3
 - 5 bits -> 0 1111 == 15
 - 8 bits -> 0111 1111 == 127
 - 11 bits -> 011 1111 1111 == 1023

Symbol	Encoding
+	0
-	1

Number		Encoding (Bias: 4)
-4		000
-3		001
-2		010
-1		011
0	000	100
1	001	101
2	010	110
3	011	111