

CS1100: Practice sheet 2

Decimal to Binary & Floating-Point Representation

Prashanth L.A.

2026-08-13

Exercise 1

Convert $(13.625)_{10}$ into binary

Solution 1

1. Converting the integer part:

$$(13)_{10} = (1101)_2$$

2. Converting the fractional part $(0.625)_{10}$:

- ▶ $0.625 \times 2 = 1.250 \rightarrow$ Integer part: 1
- ▶ $0.250 \times 2 = 0.500 \rightarrow$ Integer part: 0
- ▶ $0.500 \times 2 = 1.000 \rightarrow$ Integer part: 1

Reading the integer parts top-to-bottom gives
 $(0.625)_{10} = (0.101)_2$.

3. Combining:

$$(13.625)_{10} = (1101.101)_2$$

Exercise 2

1. Convert $(0.6)_{10}$ to binary fraction (up to 6 places).
2. Why can't computers store $(0.6)_{10}$ or $(0.9)_{10}$ with exact precision in binary?

Solution 2

1. Fractional Conversion:

- ▶ $0.6 \times 2 = 1.2 \rightarrow$ Integer part: 1
- ▶ $0.2 \times 2 = 0.4 \rightarrow$ Integer part: 0
- ▶ $0.4 \times 2 = 0.8 \rightarrow$ Integer part: 0
- ▶ $0.8 \times 2 = 1.6 \rightarrow$ Integer part: 1
- ▶ $0.6 \times 2 = 1.2 \rightarrow$ Integer part: 1 (repeats)

$$(0.6)_{10} = (0.100110\dots)_2 = (0.\overline{1001})_2$$

2. Reason:

Base-2 fractions terminate only if the denominator prime factors are powers of 2. $0.6 = \frac{3}{5}$ and $0.9 = \frac{9}{10}$ contain factor 5, causing infinite repeating bits that must be truncated.

Exercise 3

1. Calculate $(0.150 \times 0.150)_{10}$ using fixed-point with 3 decimal places.
2. Demonstrate how floating-point scientific notation avoids precision loss.

Solution 3

1. Fixed-Point (3 decimal places):

- ▶ Exact product: $0.150 \times 0.150 = 0.0225$
- ▶ Truncated: 0.022 (digit 5 is lost)

2. Floating-Point Notation:

$$(1.50 \times 10^{-1}) \times (1.50 \times 10^{-1}) = 2.25 \times 10^{-2}$$

Exponent adjusts dynamically, preserving all digits (2.25).

IEEE 754 Format

32-Bit Layout

- ▶ **Sign (s):** 1 bit (0 for positive, 1 for negative)
- ▶ **Exponent (E):** 8 bits (stores biased exponent)
- ▶ **Mantissa (m):** 23 bits (fractional component)
- ▶ **Value Formula:**

$$v = (-1)^s \times (1.m)_2 \times 2^{E-127}$$

Why biased exponent?

- ▶ Converts signed exponents (-126 to $+127$) into unsigned non-negative integers (1 to 254).
- ▶ Enables fast left-to-right floating-point comparisons using simple unsigned integer hardware.
- ▶ Exponent 11111111 represents NaN (Not a number).

Exercise 4

Convert $(-25.75)_{10}$ to normalized binary scientific notation $(-1.m \times 2^e)$ and identify 32-bit IEEE 754 components.

Solution 4

1. Raw Binary:

$$25.75 = 11001.11_2 \implies (-25.75)_{10} = (-11001.11)_2$$

2. Normalized Form:

$$(-11001.11)_2 = -1.100111_2 \times 2^4$$

3. IEEE 754 Components:

- ▶ **Sign Bit:** 1
- ▶ **Exponent:** $4 + 127 = 131 = 10000011_2$
- ▶ **Mantissa:** 10011100000000000000000 (23 bits)

Exercise 5

Compare maximum values stored in 16 bits:

1. Fixed-Point (8 int, 8 frac bits)
2. Floating-Point (1 sign, 5 exponent with bias 15, 10 mantissa bits)

Solution 5

1. Fixed-Point:

$$(11111111.11111111)_2$$

$$\text{Max} = (2^8 - 1) + (1 - 2^{-8}) = 255.99609375 \approx 256$$

$$\text{Integer part: } (2^8 - 1) = 255$$

Fractional part $(0.11111111)_2$:

$$\begin{aligned}(0.11111111)_2 &= 2^{-1} + 2^{-2} + 2^{-3} + 2^{-4} + 2^{-5} + 2^{-6} + 2^{-7} + 2^{-8} \\ &= 0.99609375\end{aligned}$$

Geometric sum: In n fractional bits are set to 1, the value equals $1 - 2^{-n}$.

$$(0.11111111)_2 = 1 - 2^{-8} = 1 - \frac{1}{256} = \frac{255}{256} = 0.99609375$$

Solution 5

2. Floating-Point (16-bit):

- ▶ Max Exp: $e = 30 - 15 = 15$ **Code 31 is reserved for Infinity and NaN.**
- ▶ Max Mantissa: $1.1111111111_2 = 2 - 2^{-10}$

$$\text{Max} = (2 - 2^{-10}) \times 2^{15} = 2^{16} - 2^5 = 65536 - 32 = 65,504$$