1.6 Floating-point numbers

Floating-point numbers and normalized scientific notation

An ***integer*** is a whole number, like 42, 0, or -95. A ***floating-point number*** is a real number, like 98.6, 0.0001, or -666.667. The point" refers to the decimal point being able to appear anywhere ("float") in the number.

- Integers are typically used for values that can be counted, like 42 cars, 0 pizzas, or -95 days.
- Floating-point numbers are typically used for values that are measured, like 98.6 degrees, 0.00001 meters, or -666.66

To improve readability and consistency, floating-point numbers are commonly written using ***normalized scientific notation*** 10^1, 1.0×10^{-4}, or -6.66667×10^2, where the number is written as a digit (+/- 1 to 9), decimal point, fractional part, times 10 term "normalized" is in contrast to non-normalized where more than one digit, or a 0, may precede the decimal point, such $^{0.1}$ or 0.1×10^{-3}.

The parts of scientific notation are named ***significand*** for the part before × and ***exponent*** for the power of 10: significand × exponent is 0, the power of ten part is sometimes omitted, as in 5.7.

In binary, normalized scientific notation consists of $1.f \times 2^{exponent}$, like 1.010×2^5. f is the fractional part.

PARTICIPATION ACTIVITY 1.6.1: Normalized scientific notation: Decimal.

Indicate which numbers are in decimal normalized scientific notation.

1) 2.05×10^3

- Yes
- No

2) 0.50×10^3

Yes

No

3) 27.8×10^3

Yes

No

4) 3.5

Yes

No

5) -5.77×10^3

Yes

No

6) 2.05×10^{-3}

Yes

No

7) 0.0

Yes

No

PARTICIPATION ACTIVITY 1.6.2: Normalized scientific notation: Binary.

Indicate which numbers are in binary normalized scientific notation.

1) 0.0

Yes

No

2) 1.01×2^3

Yes

No

3) 0.10×2^3

Yes

No

4) 11.10×2^3

Yes

No

5) -1.01×2^3

Yes

No

6) 1.01×2^{-3}

Yes

No

CHALLENGE ACTIVITY 1.6.1: Normalized scientific notation.

Start

1

2

Write the number in normalized scientific notation.

Use ^ for exponents. Ex: 10^4 for 10^4.

177.757 = Ex: 1.23x10^3

1 2 3 4

Check Next

3

4

Fractions in binary

Fractions in binary are similar to fractions in decimal. In decimal, each digit after the decimal point has weight $1/10^1$, $1/10^2$
binary, each digit after the binary point has weight $1/2^1$, $1/2^2$, $1/2^3$, $1/2^4$, etc. (so 1/2, 1/4, 1/8, 1/16, etc.). Ex: 1.1101 is 1 + 1
Note that for binary numbers, the "dot" is called a ***binary point*** (versus decimal point for decimal numbers) The general terr

Figure 1.6.1: Fractional digit weights for decimal and binary.

Decimal	__	__	__	.	__	__	__	__
	10^2	10^1	10^0		10^{-1}	10^{-2}	10^{-3}	10^{-4}
	100	10	1		1/10	1/100	1/1000	1/10000

Binary	__	__	__	.	__	__	__	__
	2^2	2^1	2^0		2^{-1}	2^{-2}	2^{-3}	2^{-4}
	4	2	1		1/2	1/4	1/8	1/16
					0.5	0.25	0.125	0.0625

Ex: 1.01 = 1 + 1/4 = 1.25

Converting decimal to binary or vice-versa when fractions are involved uses the same process as without fractions. Howev
manually converting a decimal with a fraction into binary, converting the whole and fraction parts separately, then concater
easier. Ex: For 12.25, 12 is 1100_2, and 0.25 is 0.01_2, yielding 1100.01.

If a decimal fraction cannot be exactly represented as a binary fraction using limited bits, one gets as close as possible, ove
To represent 0.8 *with only four* binary fraction bits:

- 0.1100 is 1/2 + 1/4 = 0.75, which is under by 0.05.
- 0.1110 is 0.875, which is over by 0.075.
- 0.1101 is 0.8125, also over but only by 0.0125, which is closest and so is the best representation.

Obviously, more bits means binary fraction values can be closer.

PARTICIPATION ACTIVITY 1.6.3: Fractions in binary.

1) 1.1 binary = _____ decimal. Type as: #.#

Check Show answer

2) 0.001 binary = _____ decimal. Type as: 0.###

Check Show answer

3) 10.101 binary = _____ decimal. Type as: #.###

Check Show answer

4) 0.75 decimal = _____ binary. Type as: 0.##

Check Show answer

5) 0.625 decimal = _____ binary. Type as: 0.###

Check Show answer

6) 16.75 decimal = _____ binary. Type as: #####.##

Check Show answer

7) 0.6 decimal = _____ binary, using two fraction bits. Type as: 0.##

Check Show answer

Binary floating-point representation

In a computer, a binary floating-point number must be represented using a fixed number of bits. Normalized scientific nota commonly using 32 or 64 bits. A common 32-bit floating-point binary representation has these items:

- Sign: 1 bit. 0 means positive, 1 negative.
- Exponent: 8 bits. Instead of two's complement, the exponent is biased, meaning a fixed value is subtracted, in this cas exponent of 00000000 means $2^{0 - 127} = 2^{-127}$, and of 1111111 means $2^{255 - 127} = 2^{128}$.
- Fraction: 23 bits. Because the significand's first digit is always 1, the 1 implicitly precedes the fraction and thus isn't e represented. The significand in scientific notation is commonly called the *mantissa*.

PARTICIPATION ACTIVITY 1.6.4: A 32-bit floating-point representation.

Start 2x speed

	Sign	Exponent	Significand
8	0	1 0 0 0 0 0 1 0	1. 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
-20	1	1 0 0 0 0 0 1 1	1. 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
	31	30 29 28 27 26 25 24 23	1. 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
	0 +, 1 -	$2^{(x-127)}$	

$+1.0 \times 2^{(130 - 127)}$

$+1.0 \times 2^{3} = 8$

$-1.25 \times 2^{(131 - 127)}$

$-1.25 \times 2^{4} = -20$

The above is known as the ***IEEE single-precision binary floating-point*** format, which uses 32 bits: 1 bit for sign, 8 bits for e by 127), and 23 bits for significand (leading 1 before binary point is implicit). ***IEEE double-precision binary floating-point*** fo bits: 1 bit for sign, 11 bits for exponent, and 52 bits for significant (leading 1 before binary point is implicit).

Note: In C, C++, and Java, a variable like "float x" uses 32-bits (single precision), while "double x" uses 64 bits (double precis Programmers usually use double to obtain more significand precision, unless memory is tightly constrained.

PARTICIPATION ACTIVITY 1.6.5: Representation of single-precision floating-point values.

Enter a decimal value:

Convert

Sign	Exponent									Significand														
0	0	0	0	0	0	0	0	0	1.	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
31	30	29	28	27	26	25	24	23		22	21	20	19	18	17	16	15	14	13	12	11	10	9	8

PARTICIPATION ACTIVITY 1.6.6: Binary floating-point to decimal.

Given the following binary floating-point representation, determine the following.

0 10000001 00100000000000000000000

1) Sign bit (type 0 or 1)

Check Show answer

2) Exponent bits

Check Show answer

3) Exponent in decimal

Check Show answer

4) Significand's fraction bits (consider copy-pasting)

Check Show answer

5) Decimal scientific notation

$\times\ 2^2$

Check Show answer

6) Decimal number

Check Show answer

PARTICIPATION ACTIVITY 1.6.7: Negative exponents.

A 32-bit binary floating-point number has an exponent of 01111101 (125). Assume the sign bit is 0 and the significand is 1.000...

1) The number is $+1 \times 2^?$

- 125
- 127
- 2
- -2

2) The number in base ten is _____ .

0.25

- 1.000...
- -2

3) In the 32-bit representation, the sign bit applies to _____ .

- the significand
- the exponent
- both the significand and the exponent

Decimal to binary floating-point

Decimal is converted to 32-bit floating-point by first converting the decimal number to binary, then normalizing and adjustir and finally filling in the appropriate fields.

PARTICIPATION ACTIVITY 1.6.8: Converting decimal with fraction to binary floating-point.

Start 2x speed

decimal	binary	normalized binary	
12	1 1 0 0	1 1 0 0 $\times 2^{3}$	(3 + 127 = 130)
2.75	1 0 . 1 1	1 0 1 1 $\times 2^{1}$	(1 + 127 = 128)

	Sign	Exponent	Significand
12	0	1 0 0 0 0 0 1 0	1. 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
2.75	0	1 0 0 0 0 0 0 0	1. 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
	31	30 29 28 27 26 25 24 23	22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

PARTICIPATION ACTIVITY 1.6.9: Binary floating-point.

Consider converting a binary number to the above 32-bit binary floating-point format.

1) For 1.001_2, the 23 fraction bits are _____ .

- 1.001000...
- 001000...

2) For 11.101_2, the 23 fraction bits are _____ .

- 11101000...
- 1101000...
- 101000...

3) For 1111.01_2, normalizing yields 1.11101. Thus, the *unbiased* exponent should be _____ .

- 0
- 2
- 3

4) For 10010.01, normalizing yields 1.001001. Thus, the *biased* exponent should be _____ .

- 4
- 127

○ 131

PARTICIPATION ACTIVITY 1.6.10: Decimal to binary floating-point.

Consider converting the decimal number -16.25 to the above 32-bit binary floating-point format. Note that -16.25 is -10000.01 in binary.

10000011 4 1.000001000... 1 000001000... 131

Sign bit

Unbiased exponent in base ten

Biased exponent in base ten

Exponent bits

Fraction bits

Significand

Reset

CHALLENGE ACTIVITY 1.6.2: Binary floating point.

1

Start

Enter the decimal equivalent of the exponent of the following floating point binary number.

1 10000001 1.00100000000000000000000

Decimal exponent: Ex: 10

1	2	3	4	5	6

Check Next

2 3 4 5 6

Exploring further:

- IEEE single-precision binary floating-point format
- IEEE double-precision binary floating-point format

Provide feedback on this section