Numbers

Douglas Crockford

Fingers

Tools

Accounting

Writing

Writing

Fertile Crescent

China

America

Egypt

1

10

100

1,000

10,000

100,000

1,000,000

Greece

Α	1	Ι	10	Ρ	100
Β	2	Κ	20	Σ	200
Γ	3	Λ	30	Τ	300
Δ	4	Μ	40	Υ	400
Ε	5	Ν	50	Φ	500
Ϛ	6	Ξ	60	Χ	600
Ζ	7	Ο	70	Ψ	700
Η	8	Π	80	Ω	800
Θ	9	Ϟ	90	Ϡ	900

Rome

I	1	X	10	C	100
II	2	XX	20	CC	200
III	3	XXX	30	CCC	300
IV	4	XL	40	CD	400
V	5	L	50	D	500
VI	6	LX	60	DC	600
VII	7	LXX	70	DCC	700
VIII	8	LXXX	80	DCCC	800
IX	9	XC	90	CM	900

Rome

I	1	X	10	C	100
II	2	XX	20	CC	200
III	3	XXX	30	CCC	300
IV	4	XL	40	CD	400
V	5	L	50	D	500
VI	6	LX	60	DC	600
VII	7	LXX	70	DCC	700
VIII	8	LXXX	80	DCCC	800
IX	9	XC	90	CM	900

China

一	1	十	10
二	2	百	100
三	3	千	1,000
四	4		
五	5	萬	10,000
六	6	億	100,000,000
七	7	兆	1,000,000,000,000
八	8		
九	9		

India

0	Zero
1	Positional
2	
3	
4	
5	
6	
7	
8	
9	

DCIX

六百九

609

Whole Numbers

· · · 4 3 2 1 0

609

6 * 10↑2 + 0 * 10↑1 + 9 * 10↑0

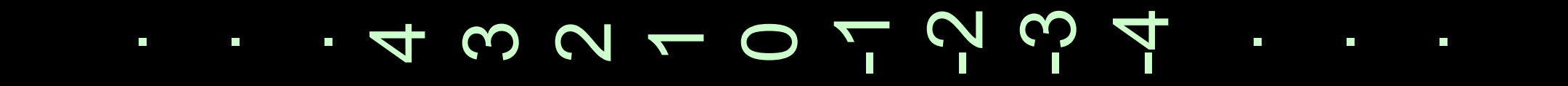
Integers

· · · 4 3 2 1 0

-609

**(0 - 1) *
(6 * 10↑2 + 0 * 10↑1 + 9 * 10↑0)**

Real



-60902

**(0 - 1) *
(6 * 10↑2 + 0 * 10↑1 + 9 * 10↑0
+ 0 * 10↑-1 + 2 * 10↑-2)**

Decimal Point

60$\overline{9}$02

60$\dot{9}$02

60$\underset{\cdot}{9}$02

60$\underset{,}{9}$02

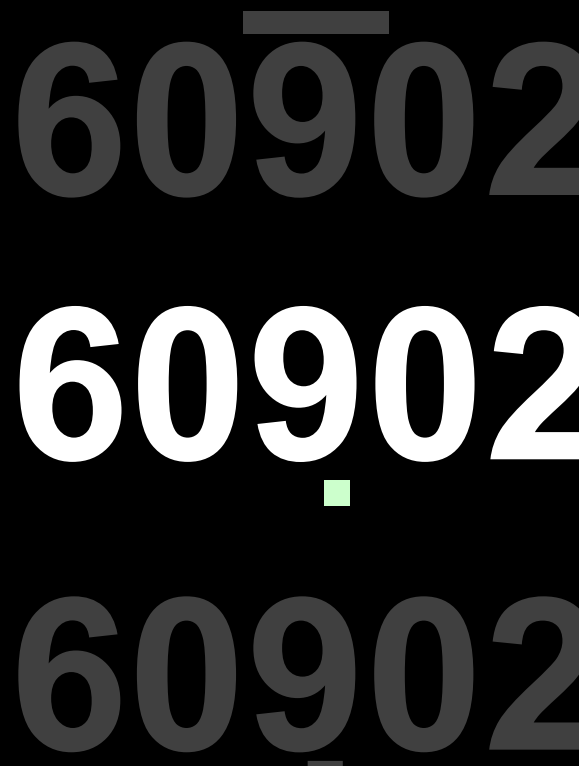
Decimal Point

2,048

60̣902

60̣902

Decimal Point



Base

10

Base

10

20

Base

10

20

12

Base

10

20

12

60

Binary

· · · 4 3 2 1 0

1100

1 * 2↑3 + 1 * 2↑2 +
0 * 2↑1 + 0 * 2↑0

Sign bit

- Signed magnitude
  - Two zeros: **0** and **-0**
- One's complement
  - **-n = not n**
  - End around carry
  - Two zeros: **0** and **-0**
- Two's complement
  - **-n = (not n) + 1**
  - The only zero is **0**
  - Extra negative

indexOf

"abc".indexOf("z")

-1

null

int

int32 + int32

int

	Java	Correct
int32 + int32	**int32**	**int33**

int

	Java	Correct
int32 + int32	**int32**	**int33**
int32 * int32		

int

	Java	Correct
int32 + int32	**int32**	**int33**
int32 * int32	**int32**	**int63**

int

Overflow

	Java	Correct
int32 + int32	**int32**	**int33**
int32 * int32	**int32**	**int63**

Overflow: What should happen?

- Store a **null** value.
- Store the largest possible (saturation).
- Fault.
- The machine should halt.

To maximize creation of error:

- Discard the most significant bits without notification.

Scaled Arithmetic

scaled value ← integer value * scale factor

Values can be added only if they have the same **scale factor**.

product ←
(multiplicand * scaled value) /
scale factor

quotient ←
(dividend * scale factor) /
scaled value

In some applications, it may be difficult to find an optimal **scale factor**.

Floating Point

Approximately real.

More convenient and accurate
than scaled arithmetic.

Decimal Floating Point

significand * 10^$\log_{10}$(scale factor)

Binary Floating Point

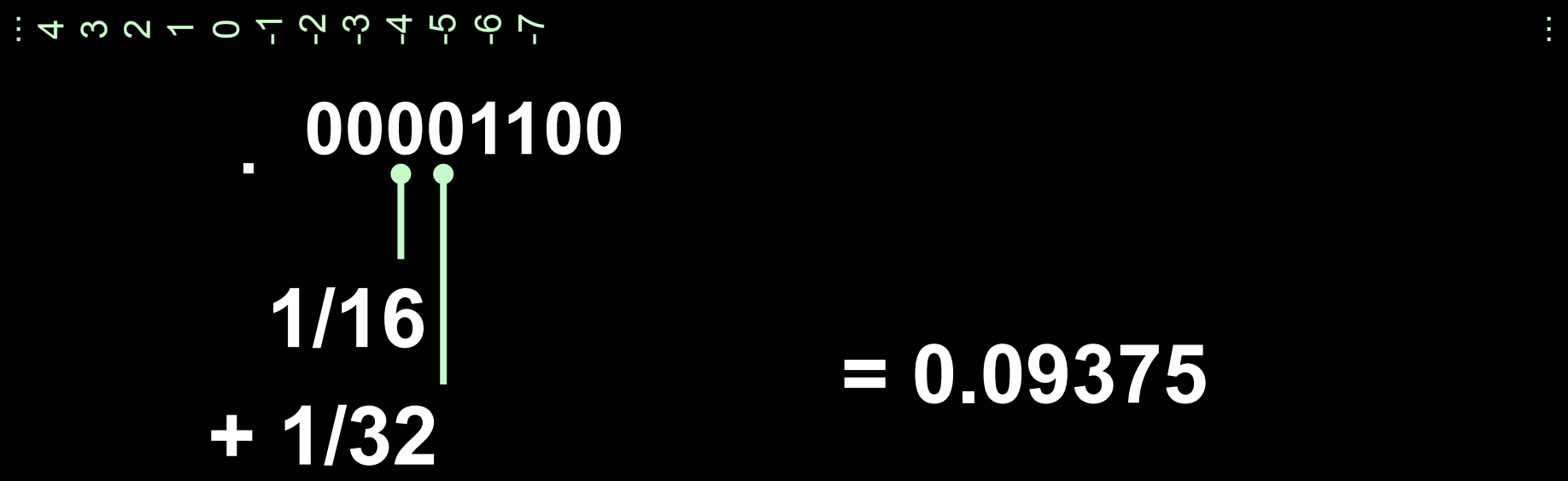
(sign * significand) * 2^exponent

0.1 + 0.2

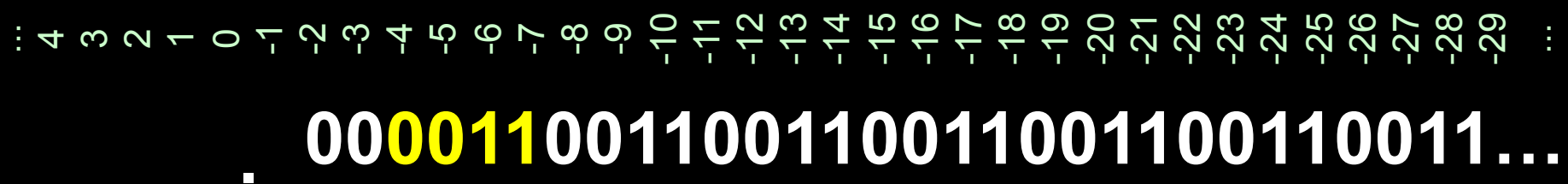
=

0.30000000000000004

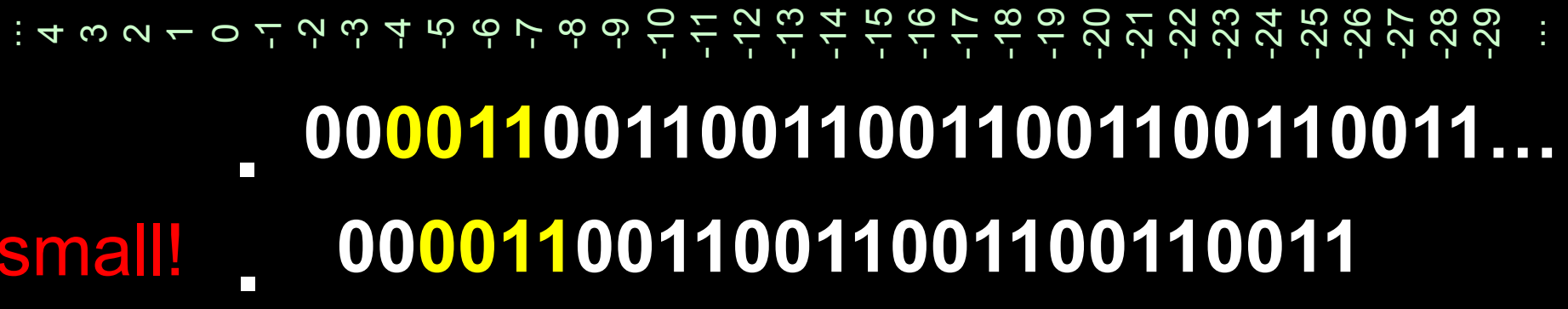
0.1



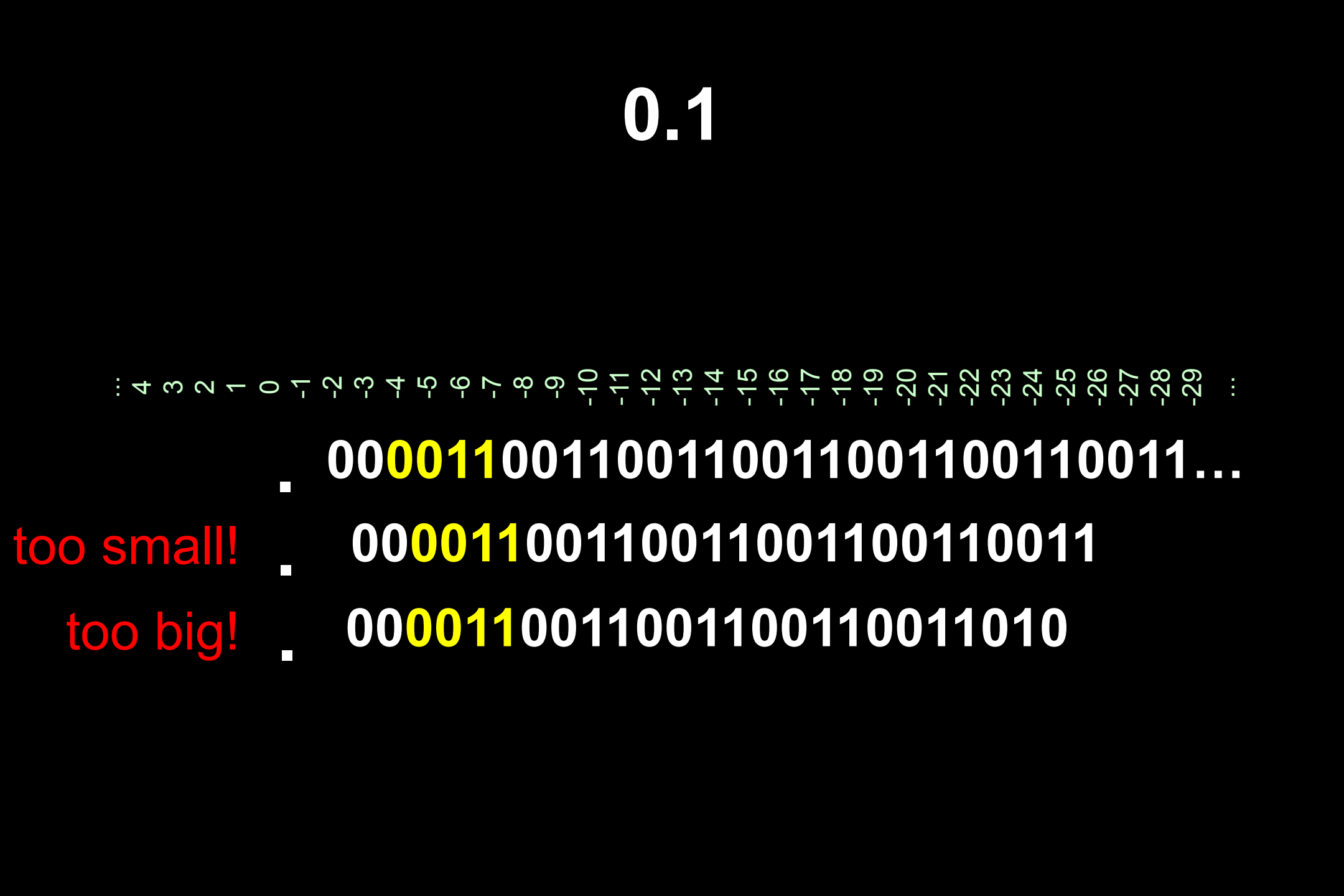
0.1


... 4 3 2 1 0 -1 -2 -3 -4 -5 -6 -7 -8 -9 -10 -11 -12 -13 -14 -15 -16 -17 -18 -19 -20 -21 -22 -23 -24 -25 -26 -27 -28 -29 ...
. 00011001100110011001100110011...

0.1


... 4 3 2 1 0 -1 -2 -3 -4 -5 -6 -7 -8 -9 -10 -11 -12 -13 -14 -15 -16 -17 -18 -19 -20 -21 -22 -23 -24 -25 -26 -27 -28 -29 ...
. 00011001100110011001100110011...
too small!
. 00011001100110011001100110011

0.1


... 4 3 2 1 0 -1 -2 -3 -4 -5 -6 -7 -8 -9 -10 -11 -12 -13 -14 -15 -16 -17 -18 -19 -20 -21 -22 -23 -24 -25 -26 -27 -28 -29 ...
. 00011001100110011001100110011...
too small! . 0001100110011001100110011
too big! . 0001100110011001100110010

The Associative Law holds only if all of the inputs, outputs, and intermediate results can all be represented exactly.

(a + b) + c

a + (b + c)

Business could not use binary floating point.

Scienti c	Business
Fortran	Cobol
Binary Floating Point	BCD

Binary Coded Decimal

0000	**0**	**0101**	**5**
0001	**1**	**0110**	**6**
0010	**2**	**0111**	**7**
0011	**3**	**1000**	**8**
0100	**4**	**1001**	**9**

Binary Floating Point Text Conversion

- Needs to be
  - Correct
  - Optimal
  - Unsurprising
- Computationally expensive

Most modern programming languages are a confusion of faulty number types.

byte char short int long float double

The Ariane 5 Failure

bh = (short)dbHorizontalBias;

JavaScript has only one number type.

This avoids a large class of errors.

JavaScript has only one number type.

This avoids a large class of errors.

Unfortunately, it is the wrong type.

DEC64

- A modern decimal floating point type.
- I recommend that DEC64 is the only number type in well designed application programming languages.
- If there is only one number type, you cannot make an error by choosing the wrong type.
- In a hardware implementation, DEC64 integers can be added in a single cycle, eliminating the performance justification for **int**.

DEC64

- Numbers work the way humans think numbers work.
- Elimination numeric confusion reduces errors.
- Conversion of DEC64 numbers to text and back is simple, efficient, correct, and unsurprising.
- DEC64 can exactly represent decimal fractions. 16 digits.

1.0E-127 3.6028797018963967E+143

DEC64

Number ←

Coefficient * 10↑Exponent

dec64.com

https://github.com/douglascrockford/DEC64/

0 / 0

Unde ned.

Catch re!

null

0

1

2

$$0 * n$$

$$\frac{0 * n}{0}$$

$$\frac{0 * n}{}$$

0

NaN

0 * n

Code generators

Macro processors

Partial evaluators

Non-conditional idioms

0 / n

0 * n

n * 0

0 modulo n

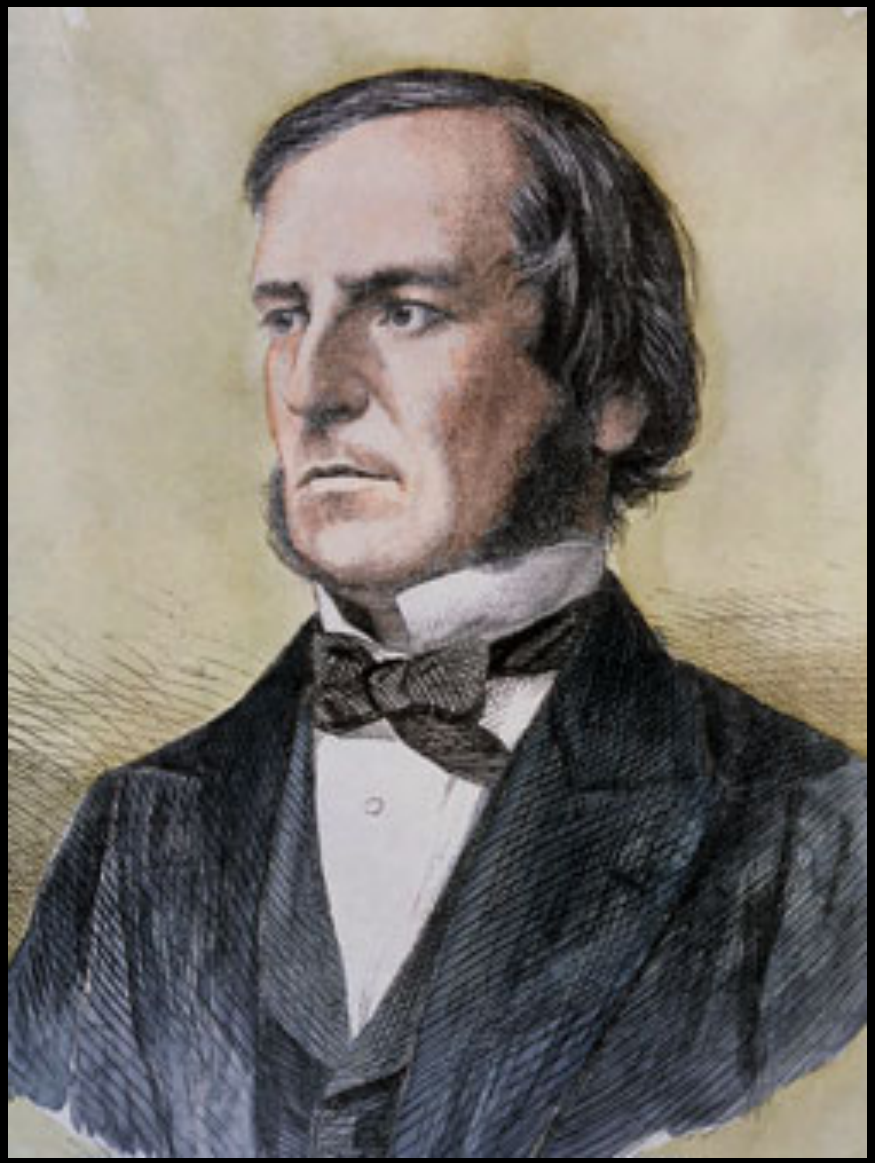
0

Thanks

Leonardo Fibonacci of Pisa

Gottried Wilhelm Leibniz

George Boole



Claude Shannon

Tehuti