BEFORE WE START…

PYTHON/JUPYTER SETUP

ASSIGNMENT 1

QUIZ 1

GEOTAB

PSCB57 - PROF. HANNO REIN

FLOATING POINT NUMBERS

WHY SHOULD YOU CARE?

1. Fundamental to all calculations on a computer
2. It works differently than pen and paper
3. Important to know the limitations
4. Calculations can be slightly or completely wrong

BINARY SYSTEM	DECIMAL SYSTEM
0	0
1	1
10	2
100	4
100100	36
-1	-1
-101	-5
0.01	0.25
0.0101010...	0.3333333...
0.000110011...	0.1

BITS AND BYTES

Smallest amount of information a computer can store is 1 bit, a 0 or 1.

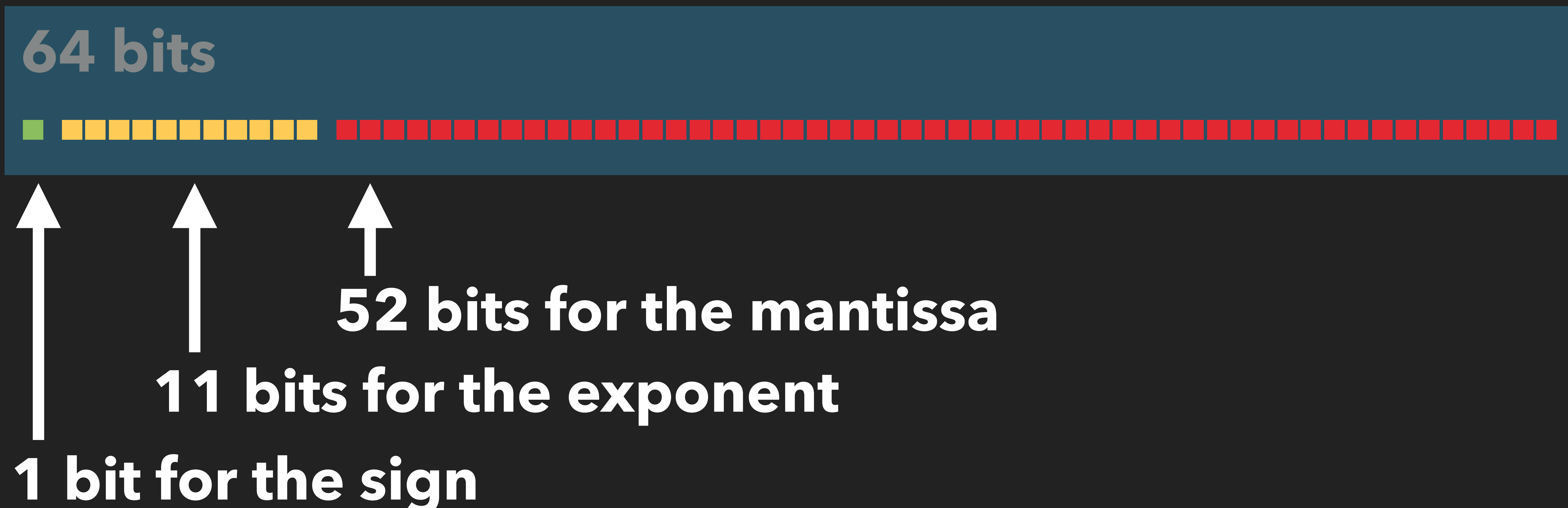
8 bits are one byte

2 bytes are one word

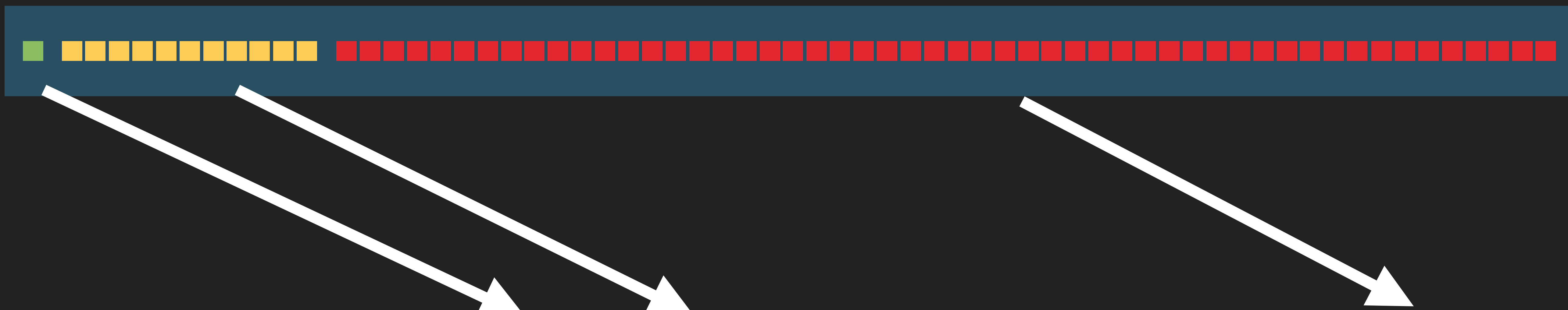
SCIENTIFIC NOTATION

`5.1e+309` = $5.1 \cdot 10^{309}$

IEEE754 STANDARD FOR FLOATING POINT NUMBERS

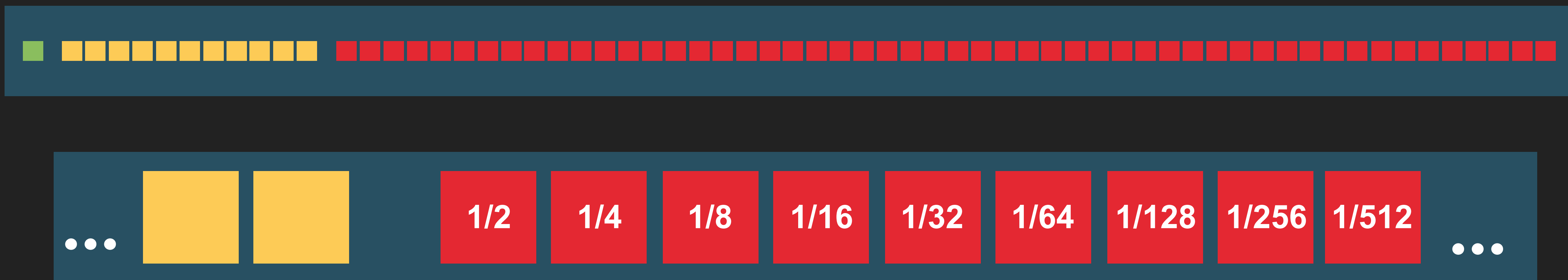


WE CAN WRITE ANY REAL NUMBER WITH A SIGN, EXPONENT+MANTISSA



$$x = (-1)^s \cdot 2^{e-1023} \cdot (1+m)$$

WE CAN WRITE ANY REAL NUMBER WITH A SIGN, EXPONENT+MANTISSA



$$2^{e-1023}\cdot(1+m)$$

EXAMPLES

$$x = (-1)^s \cdot 2^{e-1023} \cdot (1+m)$$

$$s = 0$$
$$e = 1023 \quad \longrightarrow \quad x = 1.5$$
$$m = 0.5$$

EXAMPLES

$$x = (-1)^s \cdot 2^{e-1023} \cdot (1+m)$$

$$s = 0$$
$$e = 1020$$
$$m = 0.5$$

$$\longrightarrow$$

$$x = 0.1875$$

EXAMPLES

$$x = (-1)^s \cdot 2^{e-1023} \cdot (1+m)$$

$$s = 0$$
$$e = 1024$$
$$m = 0$$

$\longrightarrow$

$$x = 2$$

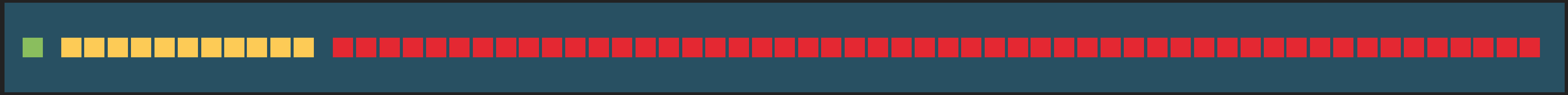
EXAMPLES

$$x = (-1)^s \cdot 2^{e-1023} \cdot (1+m)$$

$$s = 0$$
$$e = 2040 \quad \longrightarrow \quad x \approx 1.4044 \cdot 10^{306}$$
$$m = 0$$

EXAMPLES

$$x = (-1)^s \cdot 2^{e-1023} \cdot (1+m)$$



$$s = 0$$
$$e = 2$$
$$m = 0$$

$$\longrightarrow \quad x = 4.45 \cdot 10^{-308}$$

IMPORTANT SCALES

$$x = (-1)^s \cdot 2^{e-1023} \cdot (1+m)$$

EXPONENT

$$2^{11} = 2048$$

$$2^{1024} \approx 10^{308}$$

Range: $10^{-308} - 10^{308}$

IMPORTANT SCALES

$$x = (-1)^s \cdot 2^{e-1023} \cdot (1+m)$$

MANTISSA / FRACTION

$$2^{-52} \approx 10^{-16}$$

Precision: 1 part in 10^{16}

PLAY WITH FLOATING POINT NUMBERS YOURSELF

1) Jupyter notebook in PSCB57 repo
2) www.h-schmidt.net/FloatConverter/IEEE754.html

EXAMPLES!

PSCB57 - PROF. HANNO REIN

ALGORITHMIC COMPLEXITY

FIBONACCI NUMBER EXAMPLE

```
def g2(x):
    if x==0:
        return 0
    if x==1:
        return 1
    return g2(x-1)+g2(x-2)
```

$$O(2^N)$$

ALGORITHMIC COMPLEXITY

$O(1)$	Constant
$O(\log(N))$	Logarithmic
$O(N)$	Linear
$O(N \log(N))$	Log Linear
$O(N^2)$	Quadratic
$O(N^3)$	Cubic
$O(2^N)$	Exponential

MATH

FIBONACCI NUMBER EXAMPLE

```
def fib(n):
    if n==0:
        return 0
    if n==1:
        return 1
    if n%2==0:
        fn = fib(n/2)
        fn1 = fib(n/2-1)
        return (2*fn1+fn)*fn
    if n%2==1:
        fn = fib((n+1)/2)
        fn1 = fib((n+1)/2-1)
        return fn*fn+fn1*fn1
```

$$O(\log(N))$$