3-1

0 10000001 1100 0— (7)

0 10000010 0— (8)

0 10000010 1110— (15)

130-127=3 $(1+1/2+1/4+1/8) \times 2^3$

<+, 2, 1.75> (7)

<+, 2, 1> (4)

<+, 2, 2.75> = 11

↳ <+, 3, $\frac{2.75}{2}$ = 1.375> = 11

<+, 3, 1>

<+, 3, 1

<+, X, Y>

+ <+, X, Z>

= <+, X, □>

0 1000 0110 0000 1011 0—

2^7 1.04296875

x+y = z

23-bit 23-bit 24-bit

H	e	1	bitstring x
H	e	1	y
H	e+1	1	z

<+, 2, 1.75>

7 0 1000 0001 1 1100— make 8 smaller

8 0 1000 0010 1 0—

H H

1 0— => 1 0 0—

<+, 2, 2.00>

make 7 bigger

23-bits

7 0 1000 0010 0 H 1 1100— 0

<+, 3, 0.875>

0 $2^{-23} \times 2^e$

$\frac{2^2 \times 1.75}{2^3} = 2^3 \times X$

Adding loses information about smaller number

<+, e_x, m_x>

+<+, e_y, m_y> (assume $e_y > e_x$)

<+, e_y, m_z> $m_z = m_y + (m_x >> (e_y - e_x))$

e field is 8-bits $\in [-127, +127]$

i-2/ $e = 100$ (x)
$+ e = 25$ (y)
$= x$

lose: 75-bits
my 24-bits

```
for (float f=10mil; f>0; f -= 1.0) {
  ...
}
```

Algorithm for Addition: — Sort numbers x and y => small and big
- Turn mantissa(s) into 24-bit numbers (orig 23-bit)
- Right-shift m_{small} by $(e_{big} - e_{small})$
- Add mantissas => 25 bits $(24 + 24 = 25)$

25 24 mantissa
- 0 0 ——> not possible
- 0 1 ——> easy (common case of large + small)
- 1 0 ——> increment e, right-shift m by 1 (carry)
- 1 1 ——> "

lost $2^{-23} \times 2^e$

Floats ≠ Reals

e+1, 2, 1.175> 7.000 —

2^{32}

```
x = a + b + c;
y = b + c + d;
```

=>

```
t = b + c;
x = a + t;
y = t + d;
```

b and c are constant
for (changes a)
for (changes d)
=>

Reals are associative
$(x + y) + z = x + (y + z)$

$x(y + z) = xy + xz$
Floats don't

Floats are NOT

"fast-math"
-ffast-math

$a \geq b \quad a + x \geq b + x$ (monotonicity) [≠ a+1 > a]

Round-to-even => make the number even (stats wanted)
Round-to-zero => closer to zero
Round-down => truncate
Round-up => ceiling