$$\frac{|fl(x) - x|}{|x|} \leq \frac{1}{2}\varepsilon_{mach}$$

$$|fl(x) - x| \leq \frac{1}{2}\varepsilon_{mach}|x|$$

$$-\frac{1}{2}\varepsilon_{mach}|x| \leq fl(x) - x \leq \frac{1}{2}\varepsilon_{mach}|x|$$

$$x - \frac{1}{2}\varepsilon_{mach}|x| \leq fl(x) \leq x + \frac{1}{2}\varepsilon_{mach}|x|$$

Therefore $fl(x) = x + \varepsilon x$ where $|\varepsilon| \leq \frac{1}{2}\varepsilon_{mach}$.

$$\mathrm{fl}(x) = x(1+\epsilon) \quad \text{for some } |\epsilon| \leq \tfrac{1}{2}\epsilon_{\mathrm{mach}}. \qquad (1.1.6)$$

What about base 10? In base 10 we write

$$\pm\left(b_0 + \sum_{i=1}^{d} b_i 10^{-i}\right) \times 10^n \quad \text{where} \quad b_i \in \{0, 1, \ldots, 9\} \text{ and } b_0 \neq 0.$$

Number of accurate decimal digits in an approximation $\tilde{x}$ of x

$$-\log_{10}\left|\frac{\tilde{x} - x}{x}\right|$$

Example: $\pi \approx \frac{22}{7}$

```
julia> p=22/7
3.142857142857143

julia> float(π)
3.141592653589793

julia> -log10(abs(p-π)/π)
3.3952347251747166
```

3 digits agree

truncate $3 = \lfloor 3.3952\ldots \rfloor$ ← round down.

now close the other digits were

```
julia> p=22/7
3.142857142857143
1 23456789012 3456

julia> float(π)
3.141592653589793

julia> -log10(abs(p-π)/π)
3.3952347251747166
```

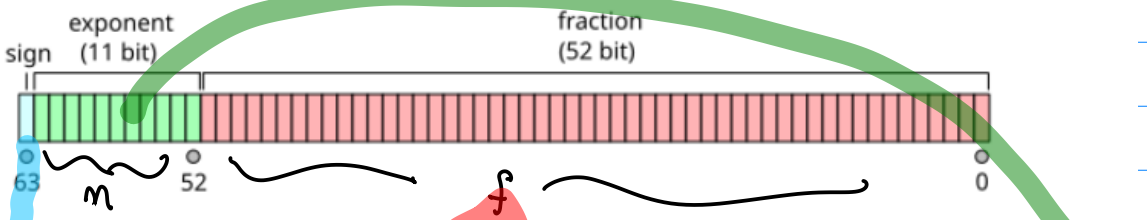
precision 16 digits

accuracy only about 3.

total 64 bits (8 bytes)

- Sign bit: 1 bit
- Exponent: 11 bits
- Significand precision: 53 bits (52 explicitly stored)

d = 52



Idea is 0.0 should be all bits equal zero---

$$\pm\left(1+\sum_{i=1}^{d} b_i 2^{-i}\right) \times 2^{n}$$

first n is a integer $n = e - 1023$ where

e = the integer represented by

let's look at the bits in some examples

```
julia> bitstring(1.0)
"0011111111110000000000000000000000000000000000000000000000000000"
```

f=0

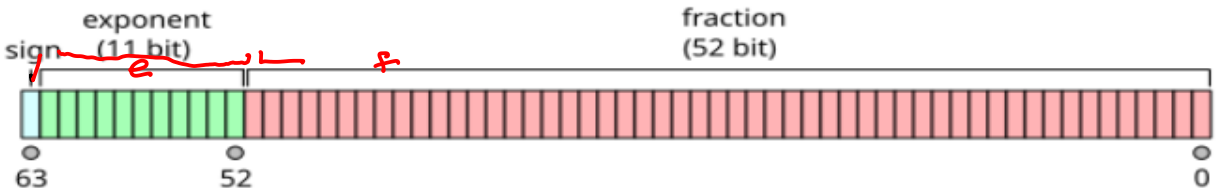
$n = e - 1023 = 1023 - 1023 = 0$

e = 01111111111 = 1023

512 256 128 64 32 16 8 4 2 1

```
julia> 1+2+4+8+16+32+64+128+256+512
1023
```

$(1+0)\times 2^0 = 1$



$f = 0$ $e = 0$

$n = 0 - 1023 = -1023$

$(1+0)\times 2^{-1023}$ special case representation that means exactly zero...

as a result the smallest exponent used in double precision arithmetic E_{52} is -1022

```
julia> bitstring(-0.0)
"1000000000000000000000000000000000000000000000000000000000000000"
```

that's strange...

```
julia> 0.0==-0.0
true
```

even stranger ...

```
julia> 1+2+4+8+16+32+64+128+256+512+1024-1023
1024
```

largest exponent that can be stored except that $n = 1024$ is another special case

```
julia> bitstring(Inf)
"0111111111110000000000000000000000000000000000000000000000000000"

julia> bitstring(-Inf)
"1111111111110000000000000000000000000000000000000000000000000000"
```

for double precision $\varepsilon_{mach} = \frac{1}{2^{52}}$

for $x \in [2^{-1022}, 2^{1023}]$

$$\left| \frac{fl(x) - x}{x} \right| \leq \frac{1}{2}\varepsilon_{mach} = \frac{1}{2^{53}}$$

```
julia> emach=1/2.0^52
2.220446049250313e-16

julia> -log10(1/2*emach)
15.954589770191003
```

there are 15 digits of decimal precision in double precision float

```
julia> p=22/7
3.142857142857143

julia> float(π)
3.141592653589793

julia> -log10(abs(p-π)/π)
3.3952347251747166
```

16 digits

16 decimal digits is enough to uniquely determine a point in $\mathbb{F}_{52}$.