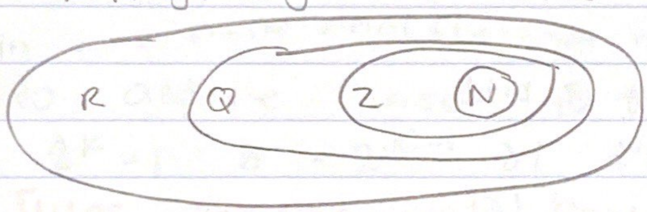
on the nature of numbers

- representation of numbers
- we use the hindu-arabic numeral system
- base 10 because we have 10 fingers
- babylonans use base 10
- natural numbers (N)
  ↳ positive int
- integers (Z)
  ↳ all #s
- rational nums (Q)
  ↳ ratio $\frac{a}{b}$, $a, b \in Z$
- irrational numbers (R)
- imaginary numbers/complex (C)

all difef sets



Computers only work on a finite set of nums
digit for a computer = a bit
↳ a bit can be 0 or 1.
- signed : include negatve
- unsigned : 0 - long number
- evry num can be written as a polynomial

unsigned char - 8 bits
unsigned int -> short / long / long long

can be signed

- char
- unsigned char

int
unsigned int
short int
unsigned

Name	size
char	usually 8 bits
short	usually 16
int	at least 16
long	at least 32
long long	at least 64

include <stdint.h>

signed	unsigned	size
int8_t	uint8_t	8 bit
int16_t	uint16_t	16 bits
int32_t	uint32_t	32 bits
int64_t	uint64_t	64 bits

For loops that won't overflow = use int

- outside of range with overflow
- remember 32 bits in a word!

binary arithmetic

- Just like normal arithmetic

$0+0=0$

$1+0=1$

$1+1=10$, 0 carry the 1

$0 \times 0=0$

$1 \times 0=0$

$1 \times 1=1$

$101+11=1000$

$101 \times 101=11001$

in a finite field (fix set numbers)

do additive inerse to be its inverse

$2^{k}-1$ & -2^{k-1}

Twos Complement Arithmetic

0000	0000	, flip the numbers, then add one to the result
0001	1111	
0010	1110	
0011	1101	
0100	1100	
0101	1011	
0110	1010	
0111	1001	

Real numbers

- $\mathbb{R}$
- continous
- uncountably infinite
- same number of infinite numbers
- you cannot write down real numbers, only rational, golden ratio, pi, or e.

floating point numbers are not real number
↳ proper subet of real, ration, ~~set~~ int sets
- They're an approximation.
- a double is at least as precious as a float

float	single percicion	IEEE 754 single	32
double	double precision	IEEE 754 double	64
long double	extended precision	IEEE 754 long double	124

$\frac{2}{5} = 0.4 = 0 \times 10^{-1}$

$\frac{1}{3} = 0.333...$

→ review binary fraction

Single precision

128 power and -128 power

fraction 23 bits

- first one = sign

$$1 + \sum_{i=1}^{23} b_{23-i} \, 2^{-i}$$

the square root of 2 is not rational

intel extended precision

1st bit = sing

15 bit = exponent

1 bit = integer part

64-bit = mantisa

80 or 112 bit

Big endian, little endian

least significant bit is

little endian = low address byte

big endian = high address byte

big endian little endian

1-8 7 8 5 6 3 4 1 2

Random numbers

computers cannot create random numbers

arithmetic operators

x mult

/ diviso

% modulo - intergers only ~~unsit~~ remainder

\+ addition & - subtraction

- follows pemdas
- mixed types
- the lower type get promoted to a higher type

x / %	—> research these
+ -	left
<< >>	left
< <= > >=	left
== !=	left
&	left
^	left
|	left
&&	left
||	left
? :	right
= += -= *= /= %=	right