Def: An alphabet Σ is a finite set of symbols

$\Sigma = \{0,1\}$, $\Sigma = \{a,b,c\} = \{b,c,a\}$

Def: A string over an alphabet Σ is a finite sequence of characters from Σ

Let $\Sigma = \{0,1\}$, 0 00 101 1100

Ex: 101010... ⇐ not a string

Def: the length of a string s is the number of characters in s

$|0| = 1$, $|101| = 3$

$|1100| = 4$

The empty string ε (→ epsilon) is the (only) string of length 0

⇒ ε is the same thing conceptually as ""

The concatenation of two strings s and t is $s \circ t$, it denotes the string st

$110 \circ 00 = 11000$

$\varepsilon \circ 110 = 110$

$\underbrace{11}_{s} \circ \underbrace{000}_{t} = 11000$

$t \circ s$

00011

Given an alphabet Σ:

$\Sigma^k = \{\text{all strings of length } k \text{ over } \Sigma\}$

$\{0,1\}^2 = \{01, 10, 00, 11\}$

$\{0,1\}^1 = \{0, 1\}$

$\{0,1\}^0 = \{\varepsilon\}$ the string of length 0

$\Sigma^* = \{\text{all strings over } \Sigma\} = \bigcup_{k=0}^{\infty} \Sigma^k$

$\{0,1\}^* = \{\varepsilon, 0, 1, 01, 10, 00, 11, 000, 111, \ldots\}$

is an infinite set

Def: a language over an alphabet

Σ is a set of strings $L \subseteq \Sigma^*$

Over $\Sigma = \{0,1\}$

$L = \{\varepsilon\}$, $L = \{\} = \emptyset \Leftarrow$ the empty set

finite

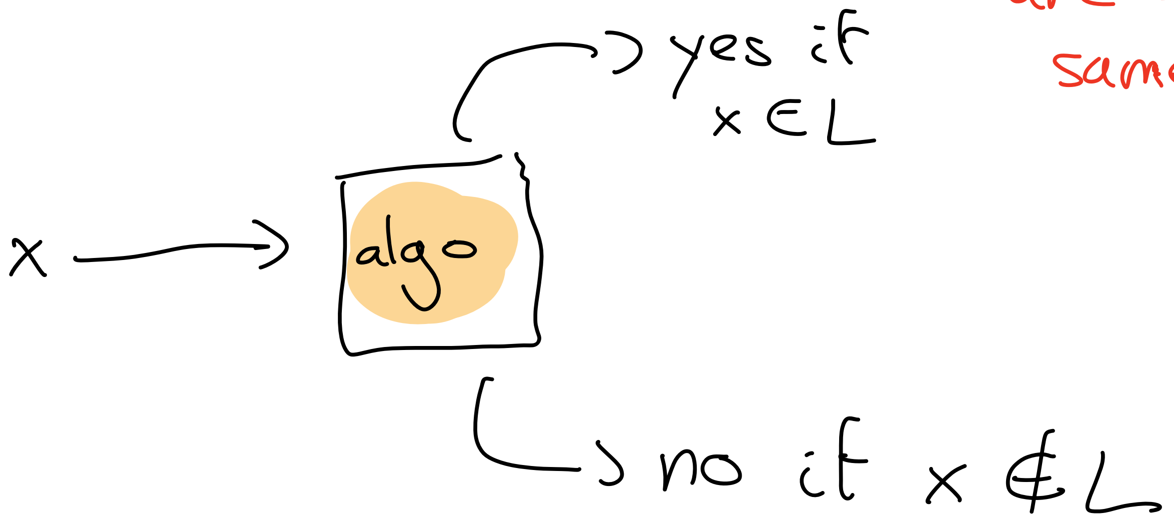
$L = \{x \mid x$ is a string over $\{0,1\}$, x has an even number of 0's$\}$

such that

$= \{00, 100, 0000, 010, \dots\}$

infinite

Fix some language L, we're given as input some string x (x and L are over the same alphabet)



for different languages L how hard is

it to build such an algorithm?

The algorithm we build will have to depend on the model of computation

Ex: $\Sigma = \{0,1\}$

(1) $L = \Sigma^* = \{$all strings over $\{0,1\}\}$

algorithm: on input x:

output yes

(2) $L = \{\} = \underline{\emptyset}$

notation for empty set

algorithm: on input x:

output no

(3) $L = \{x \mid x$ has an even number of 0's$\}$

algorithm: on input x:

- count the # of 0s in x
- if this number is 0 mod 2
  ↳ output yes
- else: output no

algorithm: on input x:

- compute the length of x => L

- take the sum
  of all digits in x
  => S

- if L - S is even
  => output yes
- else output no

algorithm: on input x

only store b
- have one bit b
  starting as 0
- read x from left
  to right
- whenever you see a
  0 set $b \leftarrow b+1 \bmod 2$

- output yes if $b=0$ at the end
- else output no

$L = \{00, \varepsilon, 11, 01, 10\}$

On input x:

If $|x| = 0 \Rightarrow$ output yes

If $|x| = 1 \Rightarrow$ output no

If $|x| = 2 \Rightarrow$ output yes

Else output no