Alphabets & Languages

An alphabet is any finite set of symbols,

like $\{a, b, c, \ldots, z\}$ or $\{0, 1\}$,

ASCII basic computer alphabet

Unicode, etc.

usually we use Σ for our alphabet.

A string is any finite sequence of alphabet characters.

so for $\Sigma = \{0, 1\}$,

strings look like 0011010,

111111,

00, etc.

a string has a length $|0011101| = 6$

$|x|$ means length of x.

The empty string is written ε,

so $|\varepsilon| = 0$

(ε is not a character in the alphabet)

generally, we use a, b, c as letters,

x, y, z are strings.

There is a concatenation operation

if $x = abaab$

$y = baab$,

in python: $x+y$

then $xy = abaab\ baab$

$yx = baab\ abaab$

We also can do exponents:

if $x = ababb$

$x^2 = xx = ababb\ ababb$

$x^3 = xxx$, etc.

$x^0 = \varepsilon$

(obeys $x^m x^n = x^{m+n}$)

no such thing as x^{-1}

A language is any set of strings from a particular alphabet

for $\Sigma = \{0, 1\}$, then

$\{0, 1, 00, 11\}$ is a language on $\{0,1\}$

$\{0^n \mid n \in \mathbb{N}\}$ $\quad (\mathbb{N} = \{0, 1, 2, \ldots\})$

↳ is a language

↳ $\{\varepsilon, 0, 00, 000, 0000, \ldots\}$

or $\{ab(a)^n ba \mid n \in \mathbb{N}\}$ is a language

$= \{abba, ababa, abaaba, \ldots\}$

$\emptyset$ is a language. The empty language.

Σ has a largest possible language,

we write it like Σ^*

Σ^* = set of all possible strings on Σ.

Σ^* is the Kleene closure of Σ.

A language is a specific set of strings.

the english language would be the set of all "legal" sentences in english

$\Sigma = \{a, \ldots, z, \text{ }, .\}$ (space) if E = the english language,

"i have a dog." $\in E$

"i have no dog." $\in E$

"i." $\notin E$

"i am am baby" $\notin E$

Python consists of all legal (syntactically valid) strings which make a python program.

Any string accepted by the interpreter counts as a string in the language.

First Type of Machine: DFA

A DFA takes a string as input, either "accepts" or "rejects".

Deterministic (no randomness — the same input always produces the same computation) **Finite** (has a finite number of "states") **Automaton** (computing machine)

The DFA takes an input string,
reads it one letter at a time L to R

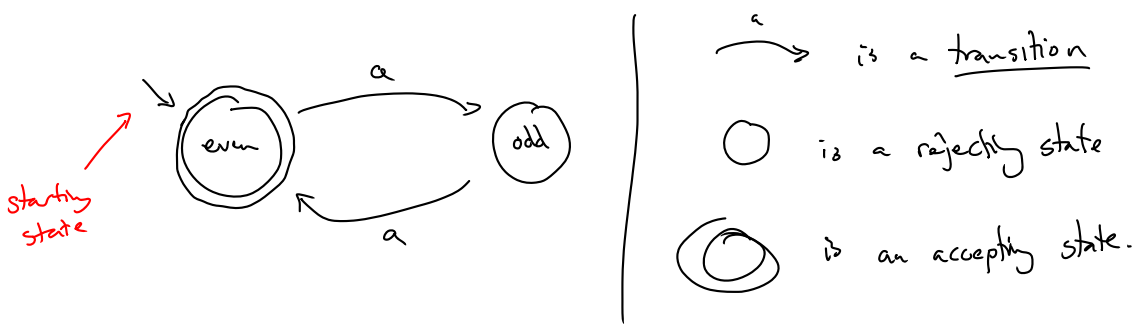
With each letter, it can change its state,
when it gets to the end, it accepts or rejects
based on the final state.

Example For $\Sigma = \{a\}$, let's make a DFA
to accept any even-length string,
reject any odd-length.

Idea: We have 2 states: even & odd,
each time I read a, we
switch from one to the other.

Start in even state.
at the end, we accept in the even state
reject in the odd state.

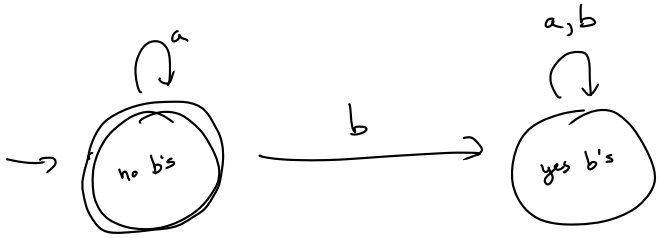
This is expressed like so:



is aaaa accepted? is accepted!

aaa is rejected

Ex/ For $\Sigma = \{a, b\}$, make a DFA which accepts any string with no b.



all states must have outgoing arrows for each letter