$(1^* 0 1^* 0 1^* + 1)$ ✓

or $(1^*)(0 1^* 0 1^*)^*$ ✓

d) all strings not beginning with 00

$\cancel{00(0+1)^*}$

cases: begin with 1
or begin with 01

$$1(0+1)^* + 01(0+1)^* + \underbrace{\varepsilon + 0}_{\text{watch out for boundary cases}}$$

e) all strings not containing 00 as a substring

$$(1+01)^* \cdot (\varepsilon + 0)$$

$\{1, 01\}^*$
all strings not containing 00
& not ending in 0

f) all strings with even # of 0's ←
and even # of 1's

idea - divide into blocks of two

$(00, 10, 01, 11)$

00: A, 10: B, 01: B, 11: A

want even # of blocks of type 10 or 01

$\cancel{00}\,10\,\cancel{00}\,01$

$$(A^* B A^* B A^*)^* + A^*$$

e.g. $\underbrace{10}_{B}\underbrace{10}_{B}\underbrace{11}_{A}$

e.g. $\underset{B}{10}\,\underset{B}{10}\,\underset{A}{11}$

$$\Big((00+11)^*\,(10+01)\,(00+11)^*\,(10+01)\,(00+11)^*\Big)^*$$
$$+\ (00+11)^*$$

g) all strings with even # of 0's
and #1's divisible by 13
and **not** containing 10110

hard but possible!

h) $\{\, 0^i 1^i \;:\; i \geq 0 \,\}$

impossible!
but how to prove it?

$\cancel{0^* 1^*}$

Identities:

$$L\big((r_1 + r_2)\, r_3\big) = L\big(r_1 r_3 + r_2 r_3\big)$$
$$L\big((r^*)^*\big) = L\big(r^*\big).$$
$$L\big((rs)^* r\big) = L\big(r(sr)^*\big).$$
$$L\big((r+s)^*\big) = L\big((r^* + s^*)^*\big)$$
$$= L\big((r^* s^*)^*\big)$$

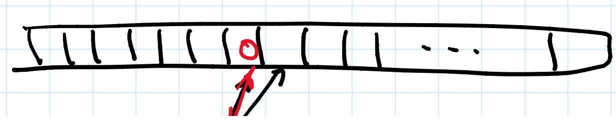
etc.

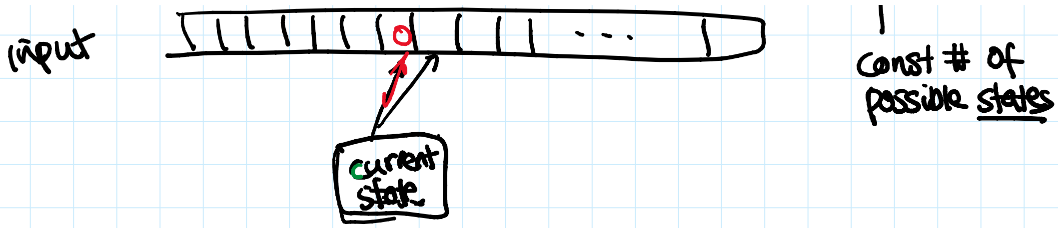
Deterministic Finite Automata (DFA)

intuitively: machine/program that reads input in one pass (from left to right)
& uses const amount of memory

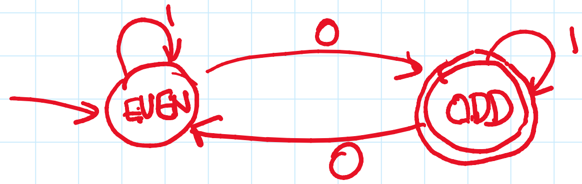
const # of possible states

input

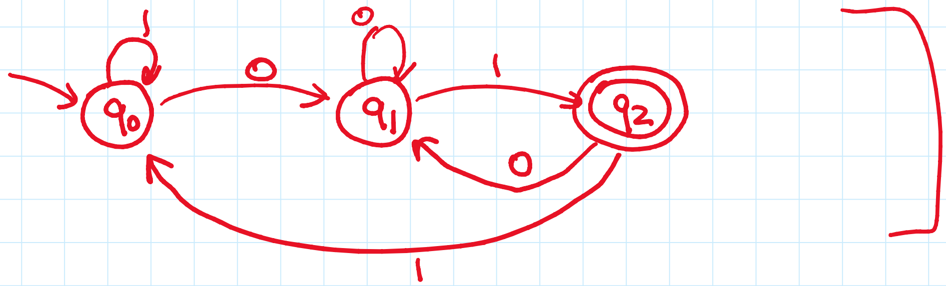




Ex0 all strings over $\{0,1\}$ with odd # of 0's



Ex1 all strings ending with 01



q_1 : just seen 0.

q_2 : just seen 01.

q_0 : none of above..

e.g. input 00101

$$q_0 \xrightarrow{0} q_1 \xrightarrow{0} q_1 \xrightarrow{1} q_2 \xrightarrow{0} q_1 \xrightarrow{1} q_2 .$$

Program:

```
state = q0
while (not end of input) {
    get next input symbol c;
    if (state == q0 && c == 0) state = q1;
    else if (state == q0 && c == 1) state = q0;
    else :
```

$\delta(q_0, 0)$

}
if (state == q_2) output yes; else no;

runs in $O(n)$ time (n = length of input)
& $O(1)$ space.

Formal Def A DFA is specified by 5 things:

$M = (Q, \Sigma, s, \delta, A)$ where

Q is a finite set of states

Σ is finite alphabet

$s \in Q$ is the start state

$A \subseteq Q$ is the set of accepting states

$\delta : Q \times \Sigma \to Q$ is the transition function

($\delta(q,a)$ denote next state if curr state is q & curr input symbol is a).

Ex $Q = \{q_0, q_1, q_2\}$
$\Sigma = \{0,1\}$
$s = q_0$
$A = \{q_2\}$

δ :

q	$\delta(q,0)$	$\delta(q,1)$
q_0	q_1	q_0
q_1	q_1	q_2
q_2	q_1	q_0

Def Given transition fn δ,
define its extended transition fn $\delta^* : Q \times \Sigma^* \to Q$
inductively:

(i) $\delta^*(q, \varepsilon) = q$

(ii) $\delta^*(q, x) = \delta^*(\delta(q,a), y)$ if $x = ay$ with $a \in \Sigma$, $y \in \Sigma^*$

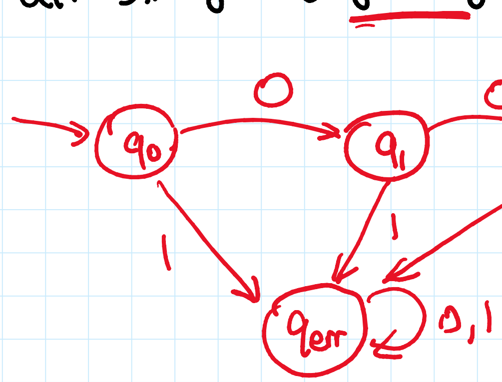
Def M accepts x iff $\delta^*(q_0, x) \in A$.

Define $L(M) = \{x \in \Sigma^* : M \text{ accepts } x\}$.

lang accepted by M

Exs $(\Sigma = \{0,1\})$.

a) all strings beginning with 001



b) all strings containing 001 as substring