HW1 due tomorrow 10am
HW2 available
Q4: subsequence
bmnn subseq of banana

Deterministic Finite Automata (DFA)
machine/program that has
fixed amount of memory
& reads input from left to right

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

Start

Ex1 all strings ending with 01
91: just seen 0
92: just seen 01
90: none of above

Program:

```
State = q0;
while (not end of input) {
    c = next input symbol;
    if (state == q0 && c == 0) state=q1;
    else if (state=q0 && c==1) state=q2;
    ...
}
if (State == q2) output yes; else no;
```

**Formal Def**  A DFA is specified by 5 things $M = (Q, \Sigma, s, A, \delta)$ where

- $Q$ is a finite set of **states**
- $\Sigma$ is finite alphabet
- $s \in Q$ is the **start state**
- $A \subseteq Q$ is the set of **accepting states**
- $\delta: Q \times \Sigma \rightarrow Q$ is the **transition function**

**Ex**

<table>
<thead>
<tr>
<th>$Q$</th>
<th>$Q_0, q_1, q_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Sigma$</td>
<td>${0,1}$</td>
</tr>
<tr>
<td>$s$</td>
<td>$q_0$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$\delta$</th>
<th>$\delta(q_0, 0)$</th>
<th>$\delta(q_0, 1)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q_0$</td>
<td>90</td>
<td>91</td>
</tr>
<tr>
<td>$q_0$</td>
<td>90</td>
<td>91</td>
</tr>
<tr>
<td>$q_1$</td>
<td>91</td>
<td>91</td>
</tr>
<tr>
<td>$q_2$</td>
<td>92</td>
<td>92</td>
</tr>
</tbody>
</table>
\[ s = q_0 \]
\[ A = \{q_2\} \]
\[ q_1 \quad q_1 \quad q_2 \]
\[ q_2 \quad q_1 \quad q_0 \]

**Def** Define the extended transition fn \( \delta^* : Q \times \Sigma^* \rightarrow Q \) inductively:

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

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

**Ex 1**
\( \delta^*(q_0, 01) = \delta^*(\delta^*(q_0, 0), 1) \)
\( = \delta^*(q_1, 1) \)
\( = \delta^*(\delta^*(q_1, 1), \varepsilon) \)
\( = \delta^*(q_2, \varepsilon) = q_2 \)

\( \delta^*(q_0, 01001) = q_2 \), \( 01001 \in L(M) \)

**Def** Define \( L(M) = \{ x \in \Sigma^* \mid \delta^*(s, x) \in A \} \)

(language accepted by \( M \))

**Exs** \( \Sigma = \{0, 1\} \)

a) all strings beginning with 001

1) all strings containing 001 as substring
b) all strings containing 001 as substring

\[ q_0 \rightarrow q_1 \rightarrow q_2 \rightarrow q_3 \]

\[ q_3: \text{found 001} \]
\[ q_2: \text{just seen 00 (but found 001)} \]
\[ q_1: \text{just seen 0 but not 00} \]
\[ q_0: \text{none of above} \]

**note:** generalizes to other pattern strings

\[ \Rightarrow \text{pattern matching algm in } O(n) \text{ time} \]

c) all strings not containing 001

\[ q_0 \rightarrow q_1 \rightarrow q_2 \rightarrow q_3 \]

d) all strings with even \# 0\'s and odd \# 1\'s
Closure Properties

Thm  If L is accepted by DFA M, then L is accepted by some DFA M'.

Pf: idea - complement the accepting states

Given M = (Q, Σ, s, A, δ)
construct M' = (Q, Σ, s, A', δ)
where A' = Q - A

Then x ∈ L(M') ⇔ δ*(s, x) ∈ A'
⇔ δ*(s, x) ≠ A
⇔ x ∈ L(M)
⇔ x ∉ L

Thm  If L1 is accepted by DFA M1, L2 is """" M2,
then L1 ∩ L2 is accepted by some DFA M.

Pf: idea - remember a pair of states

Given M1 = (Q1, Σ, s1, A1, δ1),
M2 = (Q2, Σ, s2, A2, δ2)

Construct M = (Q, Σ, s, A, δ)
where Q = Q1 × Q2
s = (s1, s2)
A = A1 × A2
(\{(a, b) | a ∈ A, b ∈ B\})

(\{(a, b) | a ∈ A, b ∈ B\})
A × B = (A1 × B1)
δ: Q × Σ → Q
\( \delta((q_1,q_2),a) = (\delta_1(q_1,a), \delta_2(q_2,a)) \)

**Lemma:** \( \delta^*((q_1,q_2),x) = (\delta_1^*(q_1,x), \delta_2^*(q_2,x)) \)

**Pf:** by boring induction ... 

Then \( x \in L(M) \iff \delta^*((s_1,s_2),x) \in A \)

\( \iff (\delta_1^*(s_1,x), \delta_2^*(s_2,x)) \in A_1 \times A_2 \)

\( \iff \delta_1^*(s_1,x) \in A_1 \) and \( \delta_2^*(s_2,x) \in A_2 \)

\( \iff x \in L_1 \cap L_2. \quad \square \)

Ex. all strings containing 001 and having odd \# 0's