1. Give English-language descriptions of the languages generated by the following regular expressions.

a) $(a \mid b)^*$ **b)** $a^* \mid b^*$ **c)** $b^*(ab^*ab^*)^*$ **d)** $b^*(abb^*)$

2. Give regular expressions over $\Sigma = \{a, b\}$ that generate the following languages.

a) $L_1 = \{x \mid x \text{ contains 3 consecutive } a\text{'s}\}$
b) $L_2 = \{x \mid x \text{ has even length}\}$
c) $L_3 = \{x \mid n_b(x) = 2 \bmod 3\}$
d) $L_4 = \{x \mid x \text{ contains the substring } aaba\}$
e) $L_5 = \{x \mid n_b(x) < 2\}$
f) $L_6 = \{x \mid x \text{ doesn't end in } aa\}$

Regular Languages

- a language is *regular* if it is generated by a regular expression

- the union of two regular languages is regular ✔
- the concatenation of two regular languages is regular ✔
- the Kleene closure of a regular language is regular ✔

- the intersection of two regular languages is regular ?
- the complement of a regular languages is regular ?

3. Prove that all finite languages are regular.

Recognizing Languages

- regular expressions provide a way to mechanically generate languages

- how to mechanically recognize languages?

- this would be useful for implementing pattern-matching, identifying and compiling legal programs, …

Recognizing Languages

Definition 3.5. Formally, a ***deterministic finite-state automaton*** M is specified by 5 components: $M = (Q, \Sigma, q_0, \delta, F)$ where

- Q is a finite set of states;
- Σ is an alphabet called the *input alphabet*;
- $q_0 \in Q$ is a state which is designated as the *start state*;
- F is a subset of Q; the states in F are states designated as *final* or *accepting* states;
- δ is a transition function that takes <state, input symbol> pairs and maps each one to a state: $\delta : Q \times \Sigma \to Q$. To say $\delta(q, a) = q'$ means that if the machine is in state q and the input symbol a is consumed, then the machine will move into state q'. The function δ must be a total function, meaning that $\delta(q, a)$ must be defined for every state q and every input symbol a.

Specifying DFAs

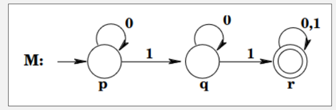
	p	q	r
0	p	q	r
1	q	r	r

- transition table
  - states *p*, *q*, *r*
  - inputs 0, 1

$M = (\{p, q, r\}, \{0, 1\}, p, \delta, \{r\})$, where δ is given by

$$\delta(p,0) = p \qquad \delta(p,1) = q$$
$$\delta(q,0) = q \qquad \delta(q,1) = r$$
$$\delta(r,0) = r \qquad \delta(r,1) = r$$

- formal definition



- transition diagram
  - incoming arrow denotes the start state (*p*)
  - double circles denote accepting states (*r*)

Recognizing Languages

The ***language accepted by*** M, denoted $L(M)$, is the set of all strings $w \in \Sigma^*$ that are accepted by M: $L(M) = \{w \in \Sigma^* \mid \delta^*(q_0, w) \in F\}$.

$\delta^*(q,w)$ = state after consuming *w*, starting from state *q*

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

- the states act as a memory for what has been matched so far in the string
- the transitions capture what can come next
- the accepting states indicate when the match is complete