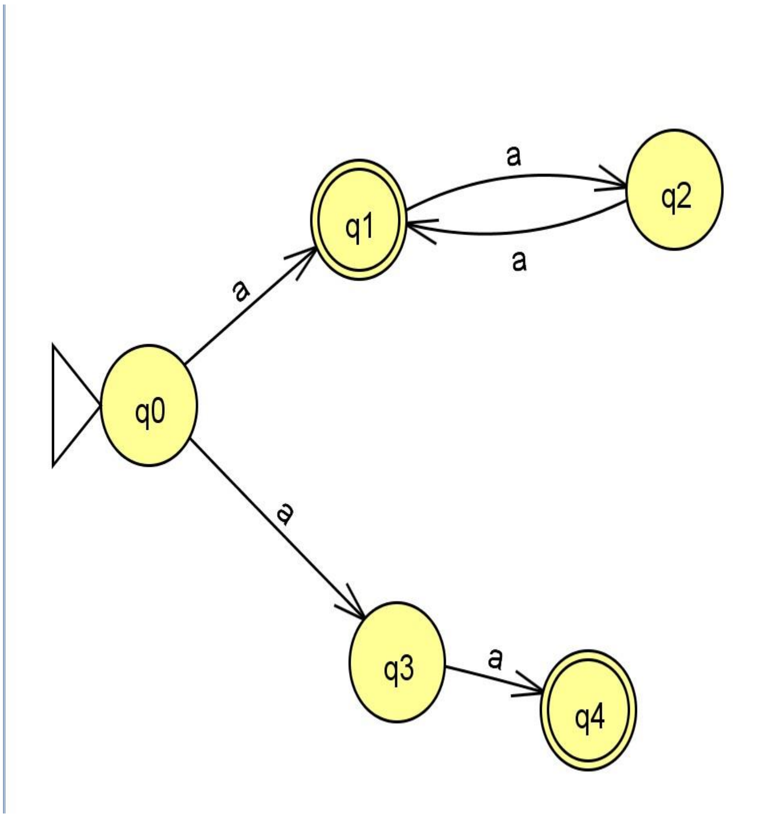
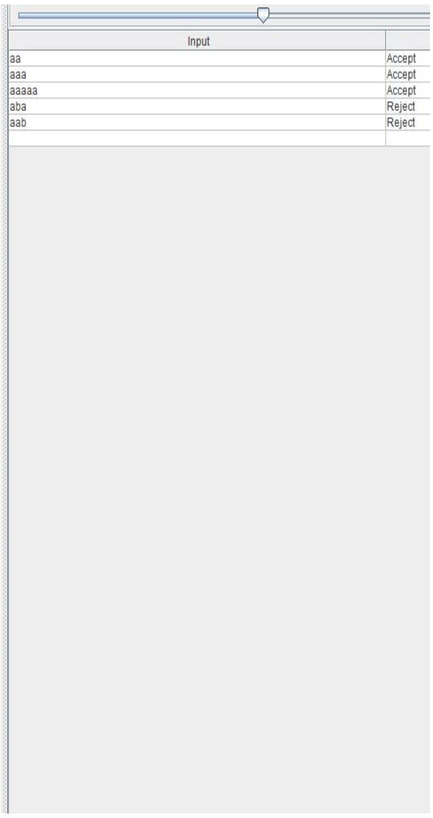
1: Create an nfa for Σ = {a} that accepts the set of all strings that consist of an odd number of 'a's ("a", "aaa", etc.) or of exactly 2 'a's ("aa").

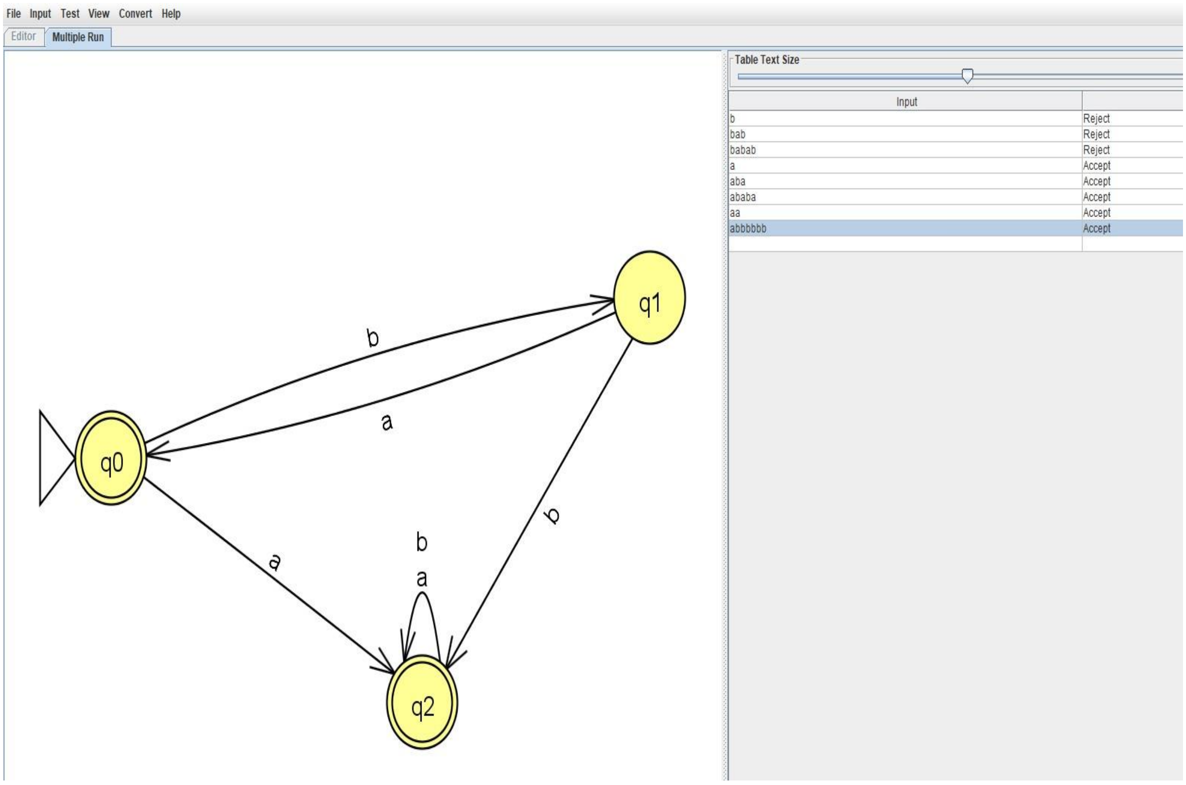




Input	
aa	Accept
aaa	Accept
aaaaa	Accept
aba	Reject
aab	Reject

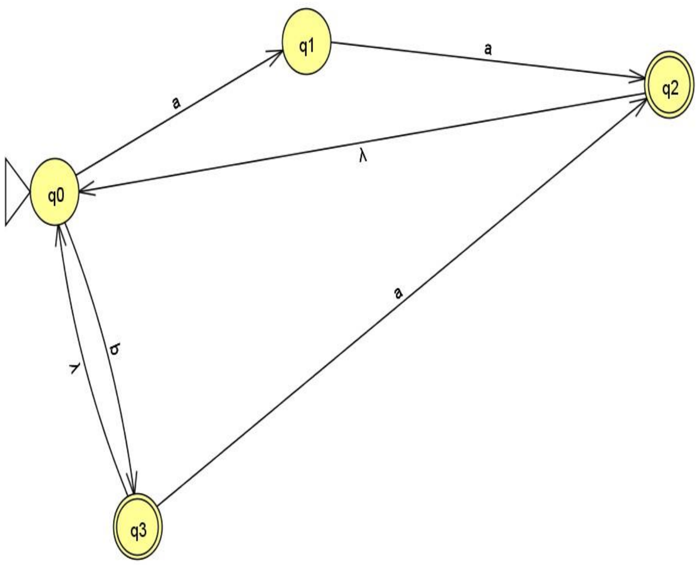
2. Create an nfa for Σ = {a,b} that accepts the **COMPLEMENT** of the language defined by the following nfa:

   states: {q0,q1}
   input alphabet: {a,b}
   initial state: q0
   final states: {q1}
   transitions:
   δ(q0,b) = {q1}
   δ(q0,λ) = {q1}
   δ(q1,a) = {q0}



3. Create an nfa that accepts **L***, where L is the language defined by the following nfa:

   states: {q0,q1,q2}
   input alphabet: {a,b}
   initial state: q0
   final states: {q2}
   transitions:
   $\delta(q0,a) = \{q1\}$
   $\delta(q1,a) = \{q2\}$
   $\delta(q0,b) = \{q1,q2\}$



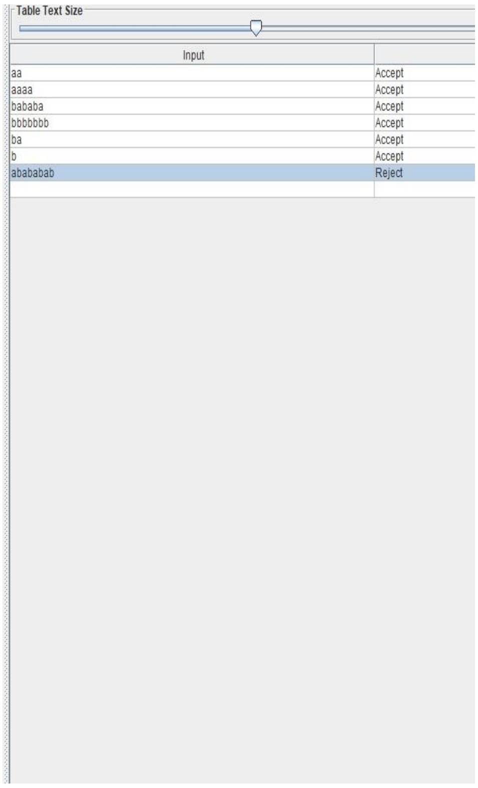
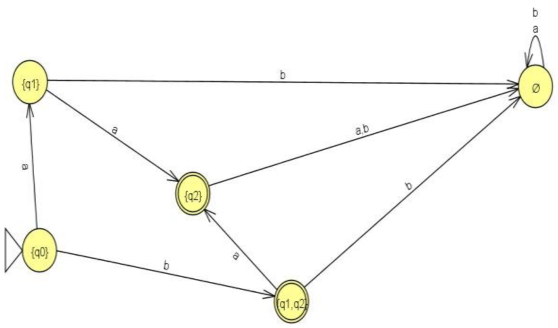


Table Text Size

Input	
aa	Accept
aaaa	Accept
bababa	Accept
bbbbbbb	Accept
ba	Accept
b	Accept
abababab	Reject

4. Use the construction of Theorem 2.2 to **convert** the following nfa **into an equivalent dfa**:

states: {q0,q1,q2}
input alphabet: {a,b}
initial state: q0
final states: {q2}
transitions:
$\delta(q1,b) = \{q1\}$
$\delta(q1,\lambda) = \{q2\}$
$\delta(q0,a) = \{q1,q2\}$



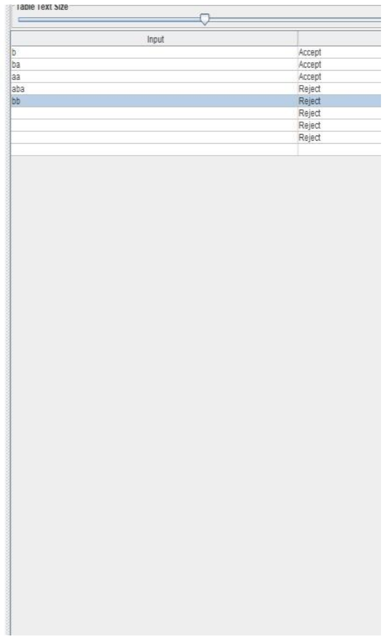


Table Text Size

Input	
b	Accept
ba	Accept
aa	Accept
aba	Reject
bb	Reject
	Reject
	Reject
	Reject

5. Prove that all finite languages are regular.

Latex → Print
Random ~~oracle model~~
Formal proof for Hash Diffie Hellman

5. Prove that all finite languages are regular.

→ If A is a finite language, then it contains finite no. of strings $a_0, a_1, \ldots, a_n$.
The language $\{a_i\}$ consists of a single string a_i is regular.
The union of a finite number of regular languages is regular as well.
i.e. $A = \{a_0\} \cup \{a_1\} \cup \ldots \cup \{a_n\}$ is regular. — ①

Also, if L is an empty set, then it is defined by the regular expression ϕ and so is regular.
If it is any finite language composed of strings $s_1, s_2, \ldots, s_n$ for some +ve integer n, then it is defined by regular expression

$$s_1 + s_2 + \ldots + s_n$$

i.e. This is regular as well — ②

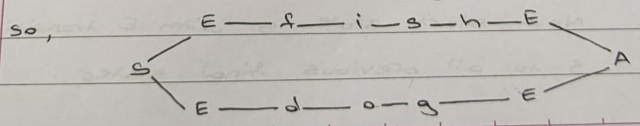
Moreover, if L consists of $a_1, a_2, \ldots, a_n$
consider a NFA to accept L:

Starting state S
Accepting state A.

In between there are many paths.

The machine can only get from the beginning to the end if it sees string a.

So,



If a language L contains strings $v_1, \dots, v_n$ then one possible expression is -

$$v_1 \cup v_2 \cup \dots \cup v_n \ .$$

Since there is a regular expression for the language, we prove that the language is regular.

Prove that if L is regular, so is L^R.

Let L be a regular

6. Prove that if L is regular, so is L^R.

language, we prove that the langu-
age is regular.

6. Prove that if L is regular, so is L^R.

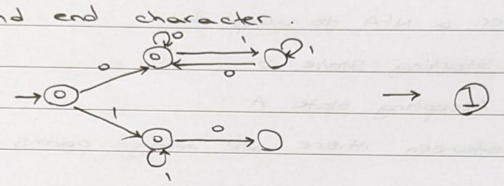
→ Let L be a regular language and

$L^R = \{ w^R = w \in L \}$.

If w is $w_1, w_2, \dots, w_n$,

then w_R is $w_n, w_{n-1}, \dots, w_1$.

Consider a DFA D for L with same start and end character.

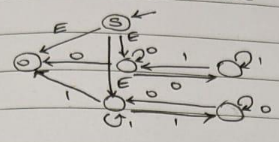


Consider a NFA N as follows -

Transitions are reversed from D.

New start state S with E transition from S to all previous final states.

If a string ends up in a final state before we want to start at that point, then the old start of D is the only final state in NFA.



→ ②

∴ From ① and ②;

When L is regular, then L^R is regular.

For ②, we cannot explain further wrt reversing / flip since it would lead to multiple start states.