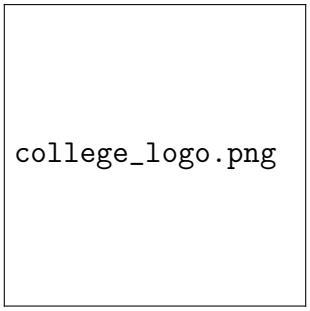

RUSTAMJI INSTITUTE OF TECHNOLOGY, TEKANPUR

Department of Computer Science Engineering

A Practical File Report On

Theory of Computation (CS-501)

Submitted By:

Name: _____

Enrollment No: _____

Submitted To:
Department of CSE
Yograj Sharma

_____ _____

(Signature of Student) **(Signature of Teacher)**

Contents

1 DFA for strings starting with "ab"

Problem Statement

Construct a DFA over $\Sigma = \{a, b\}$ that accepts all strings starting with the substring "ab".

Explanation

The DFA must see an 'a' first, moving from the start state q_0 to q_1. From q_1, it must see a 'b' to move to the accepting state q_2. Once in q_2, any subsequent sequence of 'a's and 'b's is accepted. Any other sequence of inputs at the start, such as a 'b' in state q_0 or an 'a' in state q_1, leads to a non-accepting trap state q_3.

Formal Definition (5-Tuple)

The DFA is defined as $M = (Q, \Sigma, \delta, q_0, F)$, where:

- $Q = \{q_0, q_1, q_2, q_3\}$ is the set of states.
- $\Sigma = \{a, b\}$ is the input alphabet.
- q_0 is the initial state.
- $F = \{q_2\}$ is the set of final states.
- δ is the transition function, defined by the state transition table below.

State Transition Table

State	Input a	Input b
$\rightarrow q_0$	q_1	q_3
q_1	q_3	q_2
$*q_2$	q_2	q_2
q_3	q_3	q_3

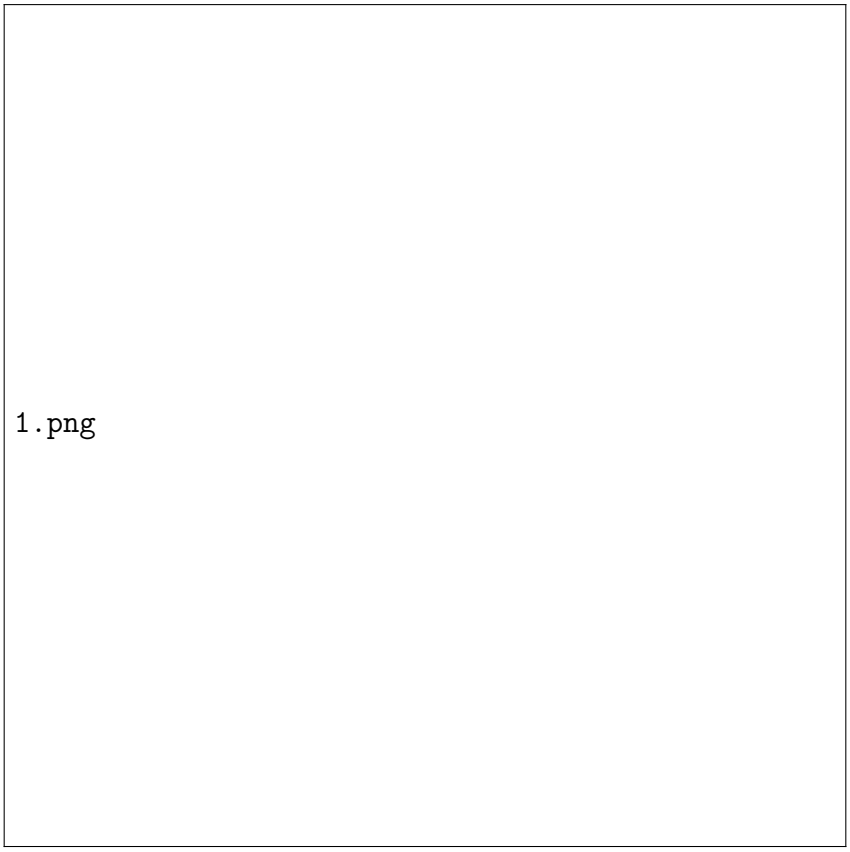

Figure 1: DFA for strings starting with "ab".

2 DFA for strings with length divisible by 3

Problem Statement

Construct a DFA over $\Sigma = \{a, b\}$ that accepts strings whose length is a multiple of 3.

Explanation

This DFA acts as a counter for the length of the string modulo 3. State q_0 represents 'length mod $3 = 0$', state q_1 represents 'length mod $3 = 1$', and state q_2 represents 'length mod $3 = 2$'. Any input symbol increments the length by one, causing a transition to the next state in the cycle. The initial state q_0 is also the final state, as a length of 0 is a multiple of 3.

Formal Definition (5-Tuple)

The DFA is defined as $M = (Q, \Sigma, \delta, q_0, F)$, where:

- $Q = \{q_0, q_1, q_2\}$ is the set of states.
- $\Sigma = \{a, b\}$ is the input alphabet.
- q_0 is the initial state.
- $F = \{q_0\}$ is the set of final states.

- δ is the transition function, defined by the state transition table below.

State Transition Table

State	Input a	Input b
$\rightarrow *q_0$	q_1	q_1
q_1	q_2	q_2
q_2	q_0	q_0

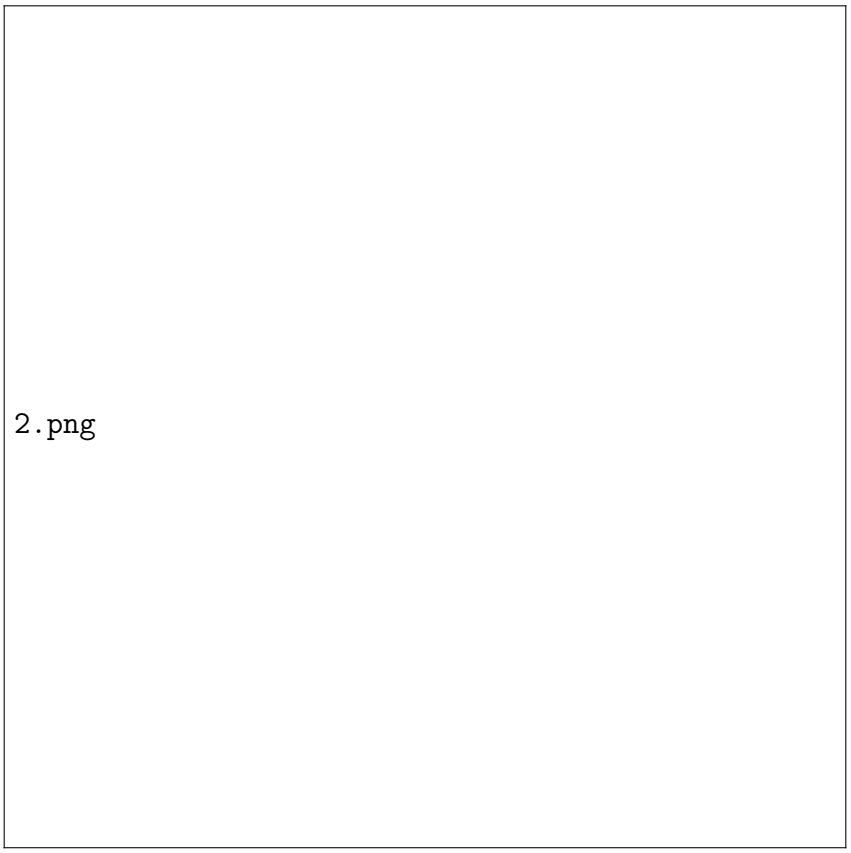

Figure 2: DFA for strings with length divisible by 3.

3 DFA for strings containing an even number of 0s

Problem Statement

Construct a DFA that accepts strings over $\Sigma = \{0, 1\}$ having an even number of 0s.

Explanation

This DFA tracks the parity of the number of 0s. State q_0 represents an even count of 0s, and state q_1 represents an odd count. The machine starts in q_0 (zero is even), which is also the accepting state. Reading a '0' toggles the state (from q_0 to q_1 or vice versa). Reading a '1' does not change the parity of 0s, so the state remains unchanged.

Formal Definition (5-Tuple)

The DFA is defined as $M = (Q, \Sigma, \delta, q_0, F)$, where:

- $Q = \{q_0, q_1\}$ is the set of states.
- $\Sigma = \{0, 1\}$ is the input alphabet.
- q_0 is the initial state.
- $F = \{q_0\}$ is the set of final states.
- δ is the transition function, defined by the state transition table below.

State Transition Table

State	Input 0	Input 1
$\rightarrow *q_0$	q_1	q_0
q_1	q_0	q_1

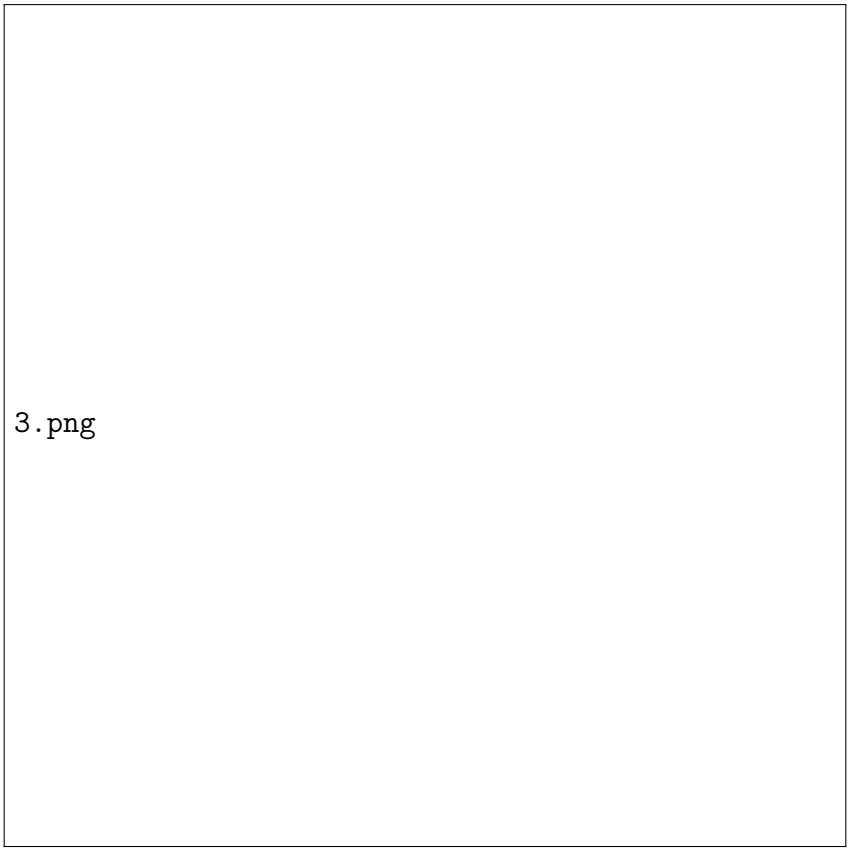

Figure 3: DFA for strings with an even number of 0s.

4 DFA for strings ending with "ba"

Problem Statement

Construct a DFA over $\Sigma = \{a, b\}$ that accepts strings ending with the substring "ba".

Explanation

The DFA needs to remember the last one or two characters to see if they form "ba". State q_0 is the initial state. If a 'b' is seen, it moves to q_1 (potential start of "ba"). From q_1, if an 'a' is seen, it moves to the final state q_2. If a 'b' is seen in q_1, it stays in q_1 (as the string now ends in 'b'). From q_2, a 'b' sends it back to q_1, while an 'a' sends it back to q_0 (ending in 'a').

Formal Definition (5-Tuple)

The DFA is defined as $M = (Q, \Sigma, \delta, q_0, F)$, where:

- $Q = \{q_0, q_1, q_2\}$ is the set of states.
- $\Sigma = \{a, b\}$ is the input alphabet.
- q_0 is the initial state.
- $F = \{q_2\}$ is the set of final states.

- δ is the transition function, defined by the state transition table below.

State Transition Table

State	Input a	Input b
$\rightarrow q_0$	q_0	q_1
q_1	q_2	q_1
$*q_2$	q_0	q_1

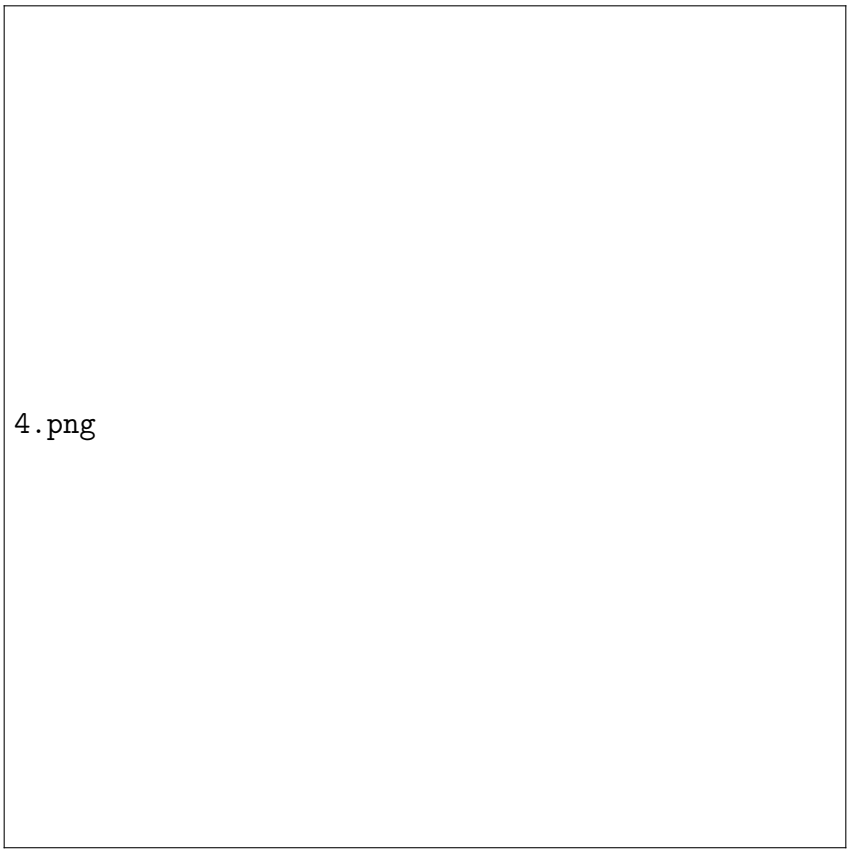

Figure 4: DFA for strings ending with "ba".

5 DFA for strings with at most two 'a's

Problem Statement

Construct a DFA over $\Sigma = \{a, b\}$ that accepts strings containing at most two 'a's.

Explanation

This DFA counts the occurrences of the symbol 'a' up to a limit. State q_0 (zero 'a's), q_1 (one 'a'), and q_2 (two 'a's) are all accepting states. Reading a 'b' does not change the count, so the DFA stays in its current state. Reading an 'a' transitions the DFA to the next state. If a third 'a' is read while in state q_2, it moves to a non-accepting trap state q_3, from which it can never leave.

Formal Definition (5-Tuple)

The DFA is defined as $M = (Q, \Sigma, \delta, q_0, F)$, where:

- $Q = \{q_0, q_1, q_2, q_3\}$ is the set of states.
- $\Sigma = \{a, b\}$ is the input alphabet.
- q_0 is the initial state.
- $F = \{q_0, q_1, q_2\}$ is the set of final states.
- δ is the transition function, defined by the table below.

State Transition Table

State	Input a	Input b
$\rightarrow *q_0$	q_1	q_0
$*q_1$	q_2	q_1
$*q_2$	q_3	q_2
q_3	q_3	q_3

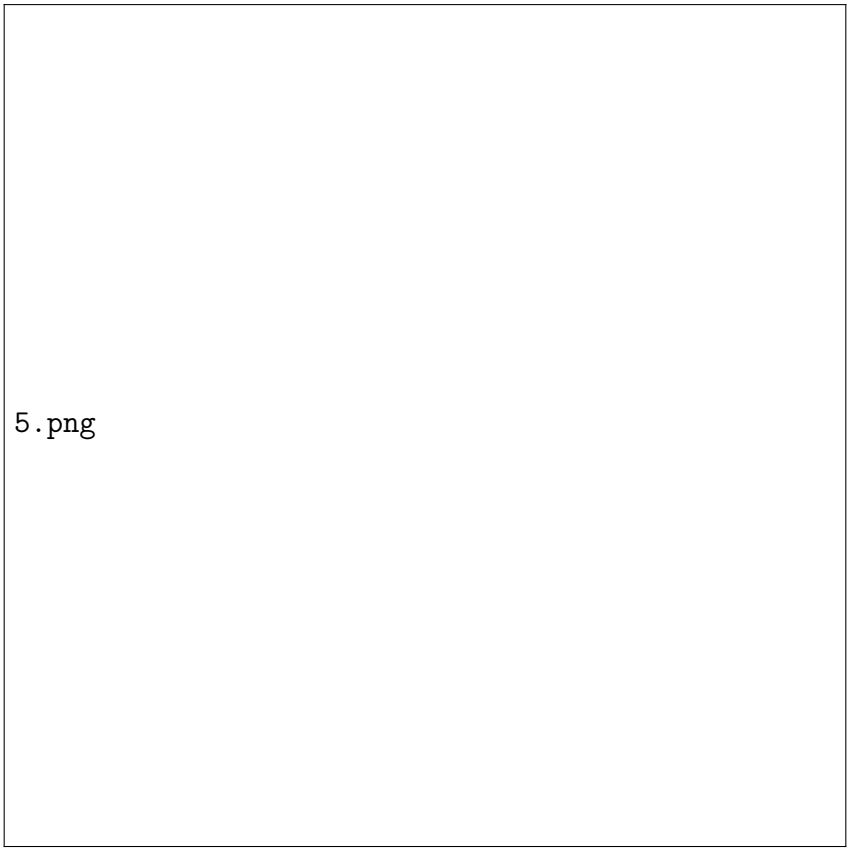

Figure 5: DFA for strings with at most two 'a's.

6 DFA for the language of all strings except "a" and "b"

Problem Statement

Construct a DFA over $\Sigma = \{a, b\}$ that accepts all strings except the single-character strings "a" and "b".

Explanation

The empty string (length 0) and strings of length 2 or more should be accepted. Strings of length 1 should be rejected. The start state q_0 must be accepting. The first input ('a' or 'b') takes the DFA to a non-accepting state q_1. Any second input takes it to a final, accepting state q_2, where it remains for all subsequent inputs.

Formal Definition (5-Tuple)

The DFA is defined as $M = (Q, \Sigma, \delta, q_0, F)$, where:

- $Q = \{q_0, q_1, q_2\}$ is the set of states.
- $\Sigma = \{a, b\}$ is the input alphabet.
- q_0 is the initial state.

- $F = \{q_0, q_2\}$ is the set of final states.
- δ is the transition function, defined below.

State Transition Table

State	Input a	Input b
$\rightarrow *q_0$	q_1	q_1
q_1	q_2	q_2
$*q_2$	q_2	q_2

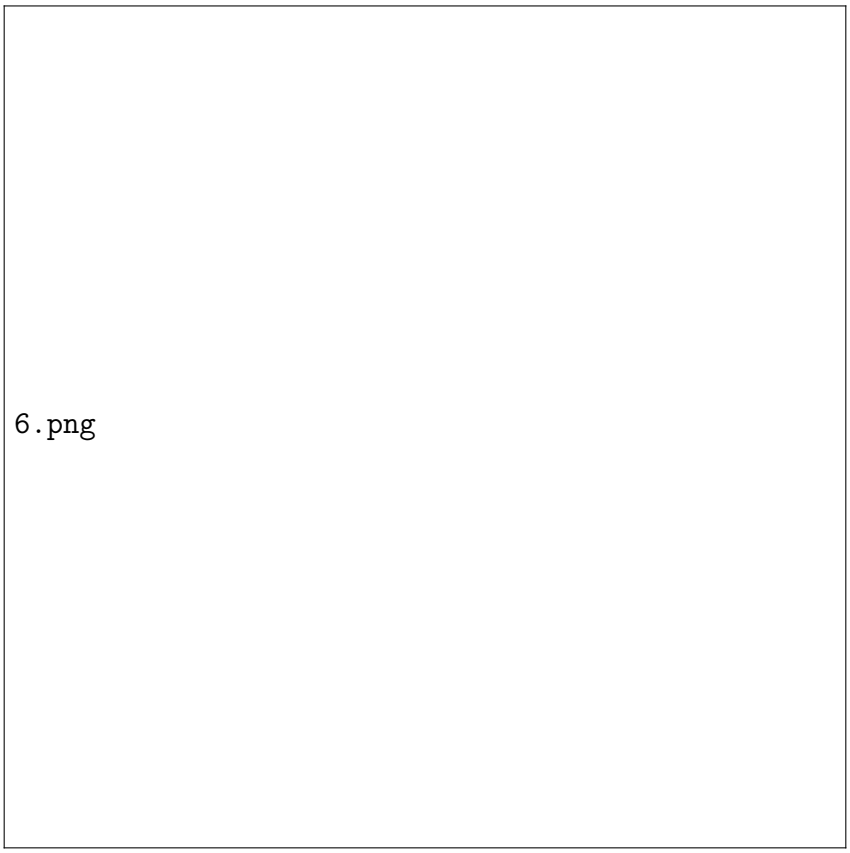

Figure 6: DFA accepting all strings except "a" and "b".

7 NFA for strings with 'a' as the third symbol

Problem Statement

Construct an NFA over $\Sigma = \{a, b\}$ that accepts all strings where the third symbol is 'a'.

Explanation

This NFA must read two arbitrary symbols, then an 'a', and then any sequence of symbols. It moves from q_0 to q_1 on any input, then from q_1 to q_2 on any input. The required 'a' transition takes it from q_2 to the final state q_3. State q_3 has a self-loop on both 'a' and 'b' to accept any characters that follow.

Formal Definition (5-Tuple)

The NFA is defined as $M = (Q, \Sigma, \delta, q_0, F)$, where:

- $Q = \{q_0, q_1, q_2, q_3\}$ is the set of states.
- $\Sigma = \{a, b\}$ is the input alphabet.
- q_0 is the initial state.
- $F = \{q_3\}$ is the set of final states.
- δ is the transition function, defined by the table below.

State Transition Table

State	Input a	Input b
$\rightarrow q_0$	$\{q_1\}$	$\{q_1\}$
q_1	$\{q_2\}$	$\{q_2\}$
q_2	$\{q_3\}$	$\emptyset$
$*q_3$	$\{q_3\}$	$\{q_3\}$

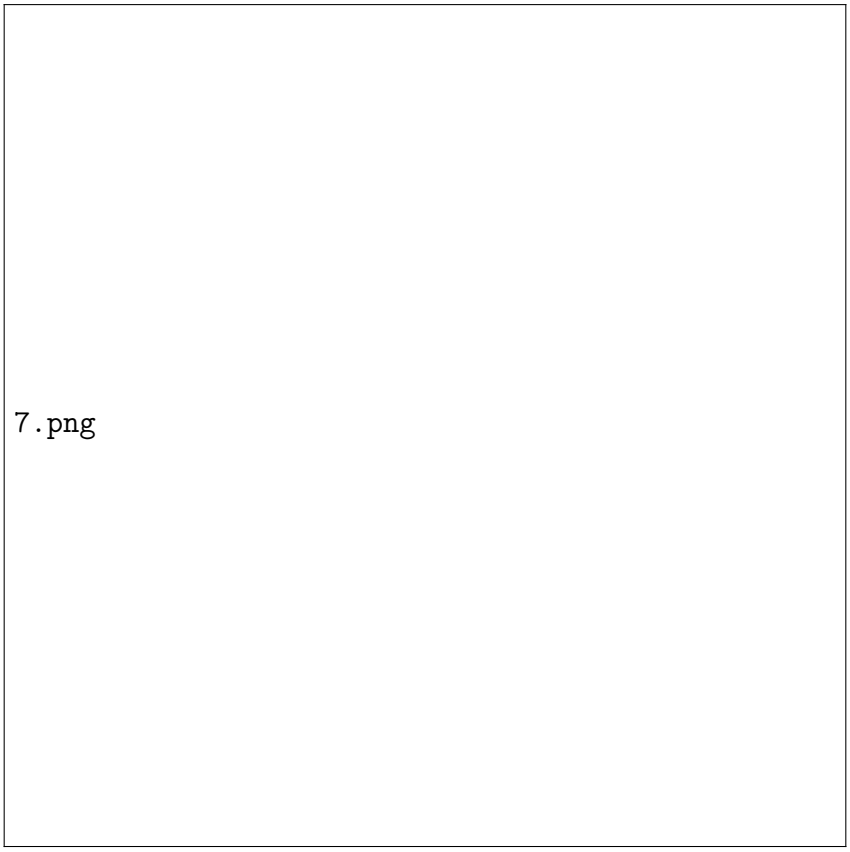

Figure 7: NFA for strings with 'a' as the third symbol.

8 NFA for the regular expression $(ab)^*$

Problem Statement

Construct an NFA over $\Sigma = \{a, b\}$ for the language defined by the regular expression $(ab)^*$.

Explanation

This NFA accepts zero or more repetitions of the substring "ab". The initial state q_0 must be accepting to handle the empty string (zero repetitions). From q_0, an 'a' transitions to a non-accepting state q_1. From q_1, a 'b' transitions back to the accepting state q_0, completing one "ab" cycle. Any other input leads to a dead configuration.

Formal Definition (5-Tuple)

The NFA is defined as $M = (Q, \Sigma, \delta, q_0, F)$, where:

- $Q = \{q_0, q_1\}$ is the set of states.
- $\Sigma = \{a, b\}$ is the input alphabet.
- q_0 is the initial state.
- $F = \{q_0\}$ is the set of final states.

- δ is the transition function, defined below.

State Transition Table

State	Input a	Input b
$\rightarrow *q_0$	$\{q_1\}$	$\emptyset$
q_1	$\emptyset$	$\{q_0\}$

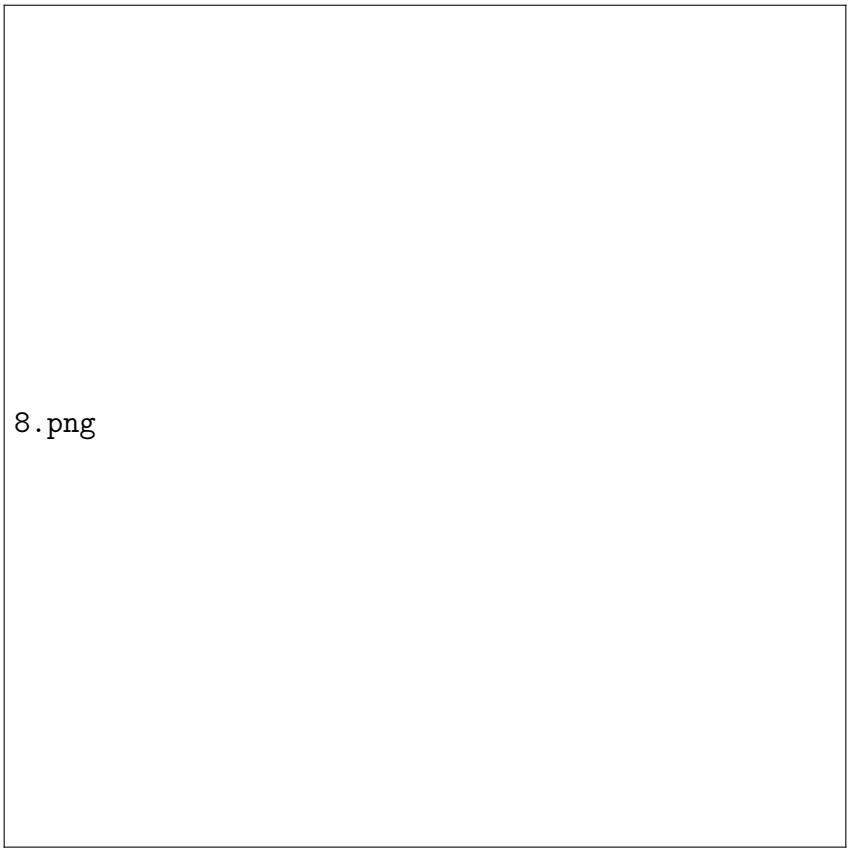

Figure 8: NFA for the regular expression $(ab)^*$.

9 NFA with ϵ-moves for $a^* + b^*$

Problem Statement

Construct an NFA with ϵ-moves for the regular expression $a^* + b^*$.

Explanation

This NFA accepts strings consisting of either only 'a's or only 'b's. The initial state q_0 uses ϵ-transitions to non-deterministically jump to one of two paths. The path starting at q_1 accepts zero or more 'a's. The path starting at q_2 accepts zero or more 'b's. Both q_1 and q_2 are accepting states.

Formal Definition (5-Tuple)

The NFA is defined as $M = (Q, \Sigma, \delta, q_0, F)$, where:

- $Q = \{q_0, q_1, q_2\}$ is the set of states.
- $\Sigma = \{a, b\}$ is the input alphabet.
- q_0 is the initial state.
- $F = \{q_1, q_2\}$ is the set of final states.
- δ is the transition function, defined by the table below.

State Transition Table

State	**Input a**	**Input b**	ϵ
$\rightarrow q_0$	$\emptyset$	$\emptyset$	$\{q_1, q_2\}$
$*q_1$	$\{q_1\}$	$\emptyset$	$\emptyset$
$*q_2$	$\emptyset$	$\{q_2\}$	$\emptyset$

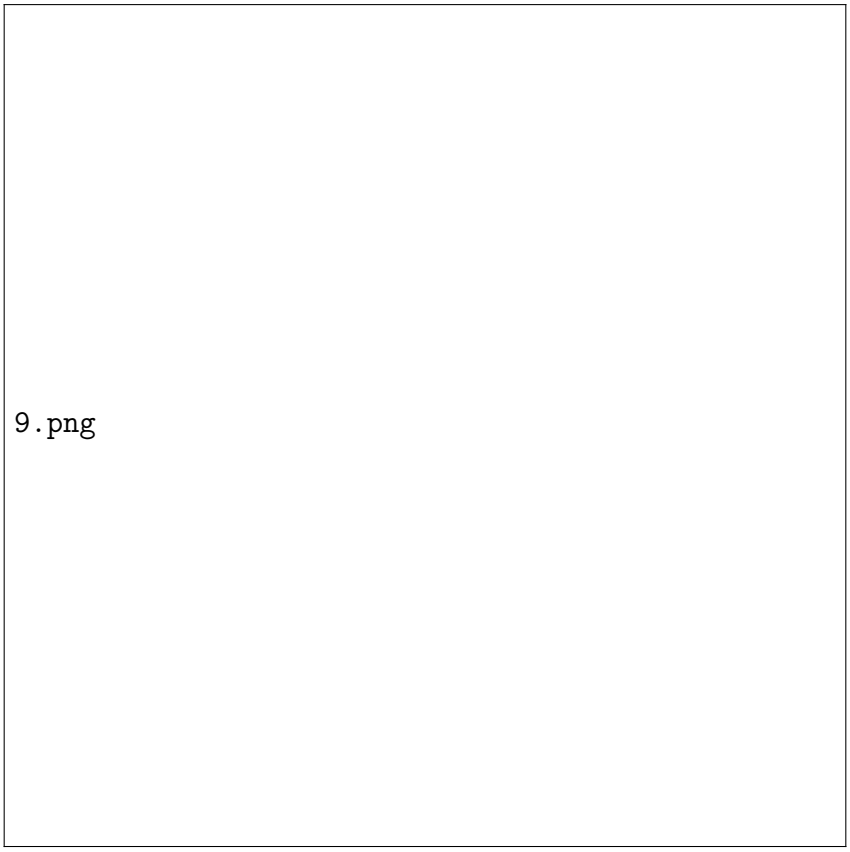

Figure 9: NFA for regular expression $a^* + b^*$.

10 NFA for strings that start with 'a' or end with 'b'

Problem Statement

Construct an NFA over $\Sigma = \{a, b\}$ that accepts strings that start with 'a' OR end with 'b'.

Explanation

This NFA uses non-determinism from the start. From q_0, it can immediately start checking for the "starts with 'a'" condition by moving to q_1. Or, it can stay in q_0 to check for the "ends with 'b'" condition. The path through q_1 accepts any string once an initial 'a' is read. The path through q_0 accepts a string only if the last symbol read is a 'b', which moves it to the final state q_2.

Formal Definition (5-Tuple)

The NFA is defined as $M = (Q, \Sigma, \delta, q_0, F)$, where:

- $Q = \{q_0, q_1, q_2\}$ is the set of states.
- $\Sigma = \{a, b\}$ is the input alphabet.
- q_0 is the initial state.

- $F = \{q_1, q_2\}$ is the set of final states.
- δ is the transition function, defined by the table below.

State Transition Table

State	Input a	Input b
$\rightarrow q_0$	$\{q_0, q_1\}$	$\{q_0, q_2\}$
$*q_1$	$\{q_1\}$	$\{q_1\}$
$*q_2$	$\emptyset$	$\emptyset$

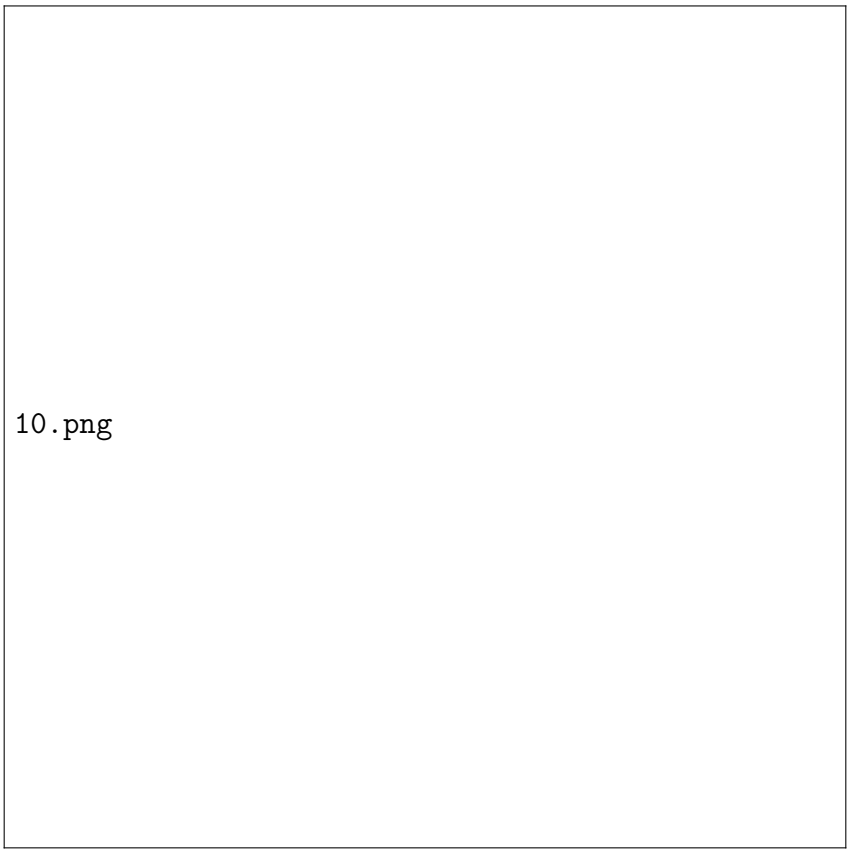

Figure 10: NFA for strings that start with 'a' or end with 'b'.

11 DFA for strings without the substring "bb"

Problem Statement

Construct a DFA over $\Sigma = \{a, b\}$ that accepts all strings that do not contain the substring "bb".

Explanation

The DFA remains in an accepting state as long as "bb" is not found. State q_0 is the initial state. Reading an 'a' keeps it in q_0. Reading a 'b' moves it to q_1. This state remembers that one 'b' has been seen. If an 'a' is read from q_1, it goes back to q_0. If another 'b' is read from q_1, the substring "bb" is formed, and the DFA moves to a permanent non-accepting trap state q_2.

Formal Definition (5-Tuple)

The DFA is defined as $M = (Q, \Sigma, \delta, q_0, F)$, where:

- $Q = \{q_0, q_1, q_2\}$ is the set of states.
- $\Sigma = \{a, b\}$ is the input alphabet.
- q_0 is the initial state.
- $F = \{q_0, q_1\}$ is the set of final states.
- δ is the transition function, defined by the table below.

State Transition Table

State	Input a	Input b
$\rightarrow *q_0$	q_0	q_1
$*q_1$	q_0	q_2
q_2	q_2	q_2

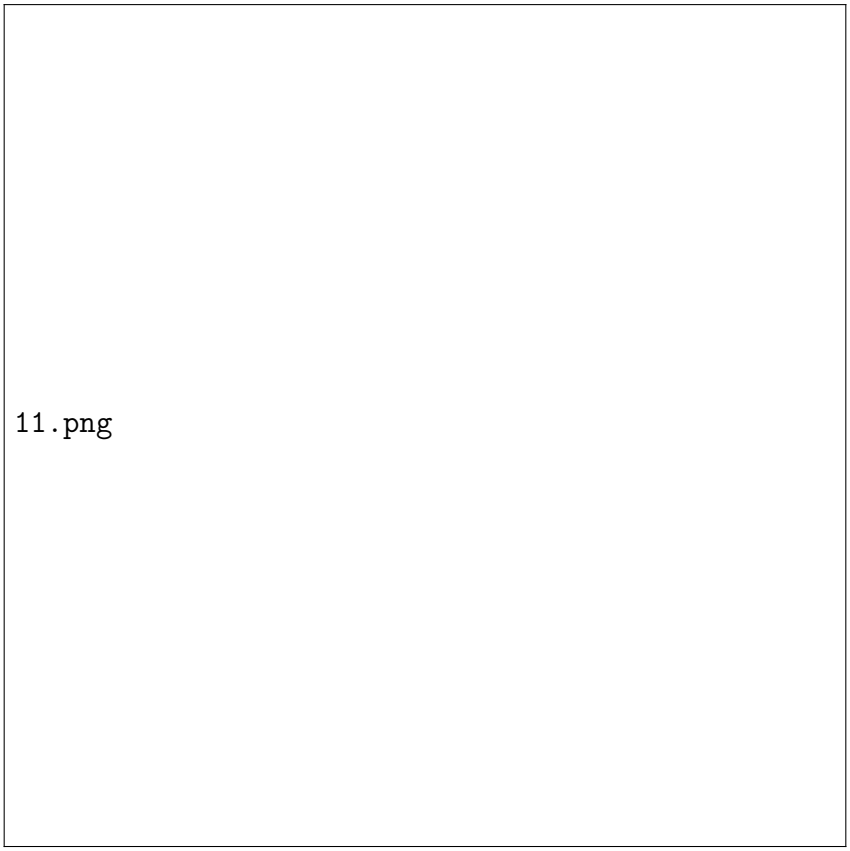

Figure 11: DFA for strings without the substring "bb".

12 DFA for binary numbers divisible by 2

Problem Statement

Construct a DFA that accepts binary strings which, when interpreted as numbers, are divisible by 2 (i.e., are even).

Explanation

A binary number is even if and only if its last digit is 0. The empty string can be considered 0, which is even. Therefore, the start state q_0 is accepting. If the machine reads a '0', it moves to the accepting state q_0. If it reads a '1', it moves to the non-accepting state q_1. This way, the machine's final state is determined solely by the last bit of the string.

Formal Definition (5-Tuple)

The DFA is defined as $M = (Q, \Sigma, \delta, q_0, F)$, where:

- $Q = \{q_0, q_1\}$ is the set of states.
- $\Sigma = \{0, 1\}$ is the input alphabet.
- q_0 is the initial state.

- $F = \{q_0\}$ is the set of final states.
- δ is the transition function, defined by the table below.

State Transition Table

State	Input 0	Input 1
$\rightarrow *q_0$	q_0	q_1
q_1	q_0	q_1

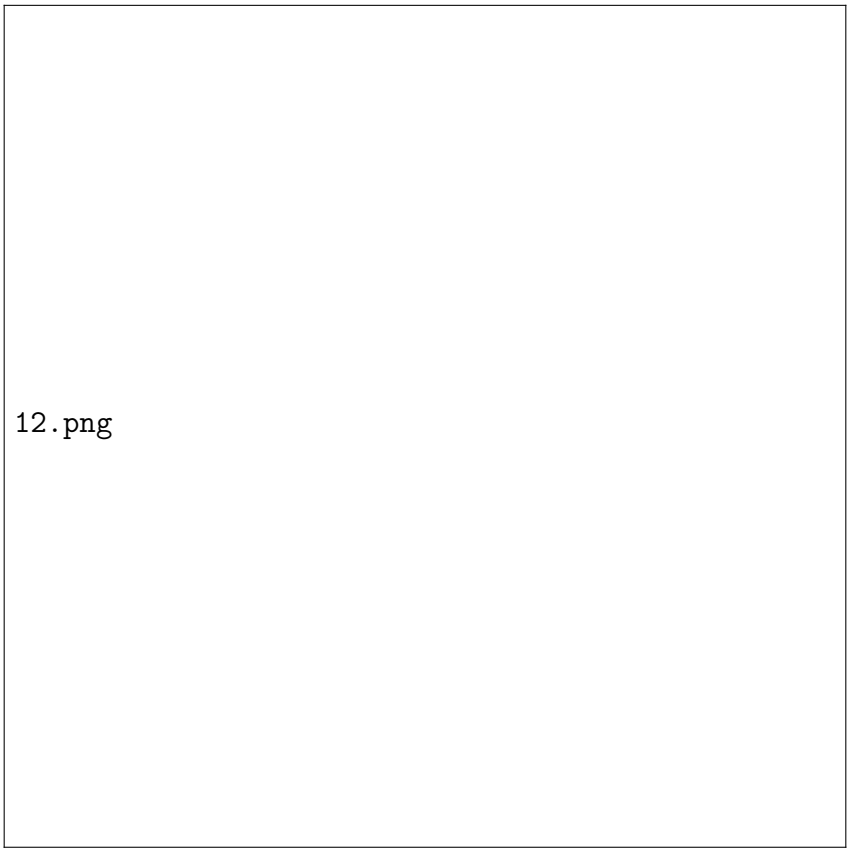

Figure 12: DFA for binary numbers divisible by 2.