Individual

(1) (a) Read the first k-1 symbols of the input, ignoring their values, and accept if the k-th symbol is a, reject otherwise.

(b)

\[ \rightarrow O \xrightarrow{a,b} O \xrightarrow{a,b} \ldots \xrightarrow{a,b} O \xrightarrow{a,b} \]

\[ k-1 \text{ edges}, \quad \text{or} \]

\[ 1k \text{ states in total} \]

Explanation: The path of length k-1 reads and ignores the first k-1 symbols, the transitions, landing us in the state \[ \rightarrow O \xrightarrow{a,b} O \xrightarrow{a,b} \]

\[ b \xrightarrow{O \xrightarrow{a,b}} \]

\[ 0\rightarrow O \xrightarrow{a,b} \]

\[ b \xrightarrow{O \xrightarrow{a,b}} \]

\[ 0\rightarrow O \xrightarrow{a,b} \]

\[ b \xrightarrow{O \xrightarrow{a,b}} \]
from here an "a" takes us to $\mathcal{O}^a_{a,b}$, which accepts regardless of what follows, and a "b" takes us to $\mathcal{O}^b_{a,b}$, which (similarly) rejects regardless of what follows.

Note: The above descriptions may be a bit long-winded, but you must have some descriptions in parts (a) and (b) that go beyond mere descriptions of the DFA itself. Same for parts (c) and (d) below.
(c) We allow ourselves to pass over any number of symbols in the input. At some we (non-deterministically) allow a symbol of "a" to take us to a part of the algorithm that ignores the next k-1 symbols and accepts (but doesn't allow for any more symbols of input).

(d) We start with \( \square \overset{a,b}{\rightarrow} 0 \overset{a}{\rightarrow} \) the rest which allows us to read any number of symbols before the "a" transition.
From \( a \rightarrow \begin{array}{c}
\text{the} \\
\text{rest}
\end{array} \) we read any \( k-1 \) symbols and accept (rejecting anything longer). Hence
\[
\begin{array}{c}
\rightarrow \\
\text{the} \\
\text{rest}
\end{array} \rightarrow \begin{array}{c}
\rightarrow \\
\text{the} \\
\text{rest}
\end{array}...
\]
\[
\rightarrow \begin{array}{c}
\rightarrow \\
\text{the} \\
\text{rest}
\end{array} \rightarrow \begin{array}{c}
\rightarrow \\
\text{the} \\
\text{rest}
\end{array}...
\rightarrow \begin{array}{c}
\rightarrow \\
\text{the} \\
\text{rest}
\end{array} \rightarrow \begin{array}{c}
\rightarrow \\
\text{the} \\
\text{rest}
\end{array}
\]

path of length \( k-1 \).

So the entire NFA is
\[
\begin{array}{c}
\rightarrow \\
\text{the} \\
\text{rest}
\end{array} \rightarrow \begin{array}{c}
\rightarrow \\
\text{the} \\
\text{rest}
\end{array}...
\]
\[
\rightarrow \begin{array}{c}
\rightarrow \\
\text{the} \\
\text{rest}
\end{array} \rightarrow \begin{array}{c}
\rightarrow \\
\text{the} \\
\text{rest}
\end{array}...
\rightarrow \begin{array}{c}
\rightarrow \\
\text{the} \\
\text{rest}
\end{array} \rightarrow \begin{array}{c}
\rightarrow \\
\text{the} \\
\text{rest}
\end{array}
\]

path of length \( k-1 \).
Note: In part (d), we used a slightly different style of description, which

- first describes the components,
- then gives the NFA,

rather than vice versa.

Either style is fine.
\[(2) \quad \text{AccFut}_{c_1}(\varepsilon) = C_1 = \Sigma^*a\]

\[
\text{AccFut}_{c_1}(a) = \varepsilon \cup \Sigma^*a
\]

Note: If you did the group homework first, you are more likely to write

\[
\text{AccFut}_{c_1}(a) = \varepsilon \cup \Sigma^*a
\]

\[
\text{AccFut}_{c_1}(b) = \Sigma^*a
\]

and, sure, \(\text{AccFut}_{c_1}(b) = \text{AccFut}_{c_1}(\varepsilon)\)

Note, here we use regular expressions like \(\Sigma^*a\) and
Since they are more convenient (perhaps...).

In an exam you are free to do the same.

In any case since one set contains $\emptyset$ and the other doesn't, these two sets are different.

Hence any DFA recognizing $C_1$ must have at least two states.

(3) Solution to be released later (but before the midterm).
For any $k \in \mathbb{N}$, $a^k \in \text{AccFut}_L(a^k b)$ since $a^k b a^k$ is a palindrome, and no shorter string lies in $\text{AccFut}_L(a^k b)$: indeed, if $\gamma_1 \ldots \gamma_n \in \text{AccFut}_L(a^k b)$ with $n < k$, and $\gamma_1 \ldots \gamma_n$ is a palindrome, then $\gamma_n, \gamma_{n-1}, \ldots, \gamma_{n-k+1} = a$ and $\gamma_{n-k} = b$ then $\gamma_{n-k} = b \neq a = \gamma_1 = \ldots = \gamma_k$ so $n-k > k$ and $n \geq 2k$ so $n \geq 2k+1$
Hence \( |\sigma_{k+2} \ldots \sigma_n| = n-(k+2)+1 \)
\[= n - k - 1 \]
satisfies
\[n - k - 1 \geq (2k+1) - k - 1 \geq k.\]

Since the shortest string in \( \text{AccFut}_L(a^kb) \)
is of length \( k \), so for \( k=1,2,3, \ldots \)
all these sets are distinct.

Hence \( L \) is non-regular.
2 (a) Since $aaa \in C_3$, but $bba \notin C_2$, we have

$\exists \in \text{AccFut}_{C_3}(aaa)$ and $\exists \notin \text{AccFut}_{C_3}(aaa)$

(b) Similarly, for any $\sigma_1, \ldots, \sigma_4$ we have

$\exists \in \text{AccFut}_{C_3}(a \sigma_1 \sigma_2)$, $\exists \notin \text{AccFut}_{C_3}(b \sigma_3 \sigma_4)$.

(c) $\text{AccFut}_{C_3}(a) = \Sigma^2 \cup C_3$, since $at \in C_3$ implies $|t| \geq 2$, so $t$ can be any length 2 string, or any element of $C_3$ (if $|t| \geq 3$, then the “a” at the beginning of “at” no effect on whether or not at $\in C_3$)
We also have
\[ \text{AccFut}_c(bba) = \Sigma^2 v C_3, \]
since the starting "bb" shows that \text{AccFut}_c(bba) contains no word of length \( \leq 2 \). The "a" in "bba" shows that all strings of length 2 lie in \text{AccFut}_c(bba). Of course, for any string \( s \) we have
\[ C_3 \subseteq \text{AccFut}_c(s). \]

(d) \[ \text{AccFut}_c(\epsilon) = C_3, \text{ and} \]
\[ \text{AccFut}_c(bbb) = C_3, \text{ since the starting} \]
"bbb" does not allow any strings of length \( \leq 3 \) to lie in \text{AccFut}_c(bbb). \]
(e)(i) Let \( s, s' \in \Sigma^3 \). Parts (a, b) show that \( \text{AccFut}_C^1(s) \neq \text{AccFut}_C^1(s') \)
provided that the first letter of \( s \) and \( s' \) are different. Similarly, for any \( \sigma_1, \sigma_2, \sigma_4 \in \Sigma \) we have
\[
\text{b} \in \text{AccFut}_C^1(\sigma_1 a \sigma_2), \quad \text{b} \notin \text{AccFut}_C^1(\sigma_3 b \sigma_4).
\]
Also
\[
\text{a} \in \text{AccFut}_C^1(\sigma_1 a \sigma_2), \quad \text{a} \notin \text{AccFut}_C^1(\sigma_3 b \sigma_4).
\]

Hence \( \text{AccFut}_C^1(s) \neq \text{AccFut}_C^1(s') \)
provided that the second letter of \( s \) and \( s' \) are different.
Similarly \( \Sigma^2 \subset \text{AccFut}_C^1(\sigma_1, \sigma_2 a) \).
and $\Sigma^2$ has no string in $\text{AccFut}_{c_3} \{ \sigma_3 \sigma_4 \sigma_6 \}$.

Hence $\text{AccFut}_{c_3}(s) \neq \text{AccFut}_{c_3}(s')$

provided that $s$ and $s'$ differ on their first, second, or third letter (symbol).

Since $s \neq s'$ implies that $s, s'$ differ on (at least) one of their 1st, 2nd, or 3rd letters, we have

$\text{AccFut}_{c_3}(s)$

are different sets for the eight elements $s \in \Sigma^3$.

(e)(ii) If $w$ is of length $n \geq 3$ and $w$ ends in $\sigma_{n-2} \sigma_{n-1} \sigma_n \in \Sigma^3$
then \( \text{AccFut}_{C_3}(w) = \text{AccFut}_{C_3}(\bar{\sigma} \bar{\sigma} \sigma \bar{\sigma}) \),

since the first \( n-3 \) letters of \( w \) irrelevant to whether \( d^t \) ndt

\( wt \in C_3 \)

If \( w = \sigma_1 \sigma_2 \) is of length 2,

then

\[
\text{AccFut}_{C_3}(\sigma_1, \sigma_2) = \text{AccFut}_{C_3}(b \sigma_1, \sigma_2),
\]

and similarly

\[
\text{AccFut}_{C_3}(\sigma_1) = \text{AccFut}_{C_3}(bb \sigma_1),
\]

and

\[
\text{AccFut}_{C_3}(\varepsilon) = \text{AccFut}_{C_3}(b b b).
\]

Hence there exactly 8 sets of

the form \( \text{AccFut}_{C_3}(w) \) as
\( w \) varies over all \( w \in \Sigma^* \).

\[\text{(3)} \exists \text{Similarly to (2) there are 4 possible values of} \]

\[\text{AccFut}_L(w)\]

as \( w \) varies over all \( w \in \Sigma^* \), namely

\[\text{AccFut}_L(aa), \text{AccFut}(ab), \text{AccFut}_L(ba), \text{AccFut}(bb).\]

\[\text{(3a) } \text{AccFut}(bba) = \text{AccFut}_{C_2}(ba), \text{AccFut}_{C_2}(ba), \]

hence

\[\delta(q.a) \text{ is the state of all strings whose accepting future equals} \]

\[\text{AccFut}_{C_2}(ba)\]
Or, in brief:

\[ \text{AccFut}_{C_2} (\cdot) = \text{AccFut}_{C_2} (ba) \]

in the notation in class.

(3b) Similarly

\[ \delta(q, a) = \text{state corresponding to } \text{AccFut}_{C_2} (\Delta_{C_a}) \]

and

\[ \delta(q, b) = \text{AccFut}_{C_2} (\Delta_{C_b}) \]

(3c) \( \text{AccFut}_{C_2} (\varepsilon) = \text{AccFut}_{C_2} (bb) \),

(and the initial state is where \( \varepsilon \) winds up in the DFA). Hence

\[ q_0 = \text{state corresponding to } \text{AccFut}_{C_2} (bb). \]
(3d) A state corresponding to \( \text{AccFut}_{C_2}(s) \) is accepting iff
\[ \exists \in \text{AccFut}_{C_2}(s). \text{ Since} \]
\[ aa, ab \in C_2, \quad ba, bb \notin C_2 \]
we have
\[ \exists \in \text{AccFut}_{C_2}(aa) \quad \exists \notin \text{AccFut}_{C_2}(ba) \]
\[ \exists \in \text{AccFut}_{C_2}(ab) \quad \exists \notin \text{AccFut}_{C_2}(bb) \]
Hence \( E \) consists of the states corresponding to
\[ \text{AccFut}_{C_2}(aa), \text{ AccFut}_{C_2}(ab). \]
(3e)

```
\begin{align*}
&\rightarrow q_{bb} \\
&\rightarrow q_{ab} \\
&\rightarrow q_{aa}
\end{align*}
```

Equivalently:

```
\begin{align*}
&\rightarrow q_{aa} \\
&\rightarrow q_{ab} \\
&\rightarrow q_{bb}
\end{align*}
```
where

\[ q^*_w = \text{state corresponding to } \ \text{AccFut}^+_{C_z}(w). \]