RegEx

Format:

1. $a \in \Sigma$, "a" is a RegEx over Σ, where L(a) = {a}
2. $b \in \Sigma$, "b" is a RegEx over Σ, where L(b) = {b}
3. $c \in \Sigma$, "c" is a RegEx over Σ, where L(c) = {c}
4. $\lambda \in \Sigma$, "λ" is a RegEx over Σ, where L(λ) = {λ}
5. "Σ" is a RegEx, L(Σ) = Σ = {a, b, c} (of length 1, so $\Sigma \circ \Sigma$ has length 2)
6. "Σ" is a RegEx (line 8), so "$(\Sigma)^\star$" is a a RegEx over Σ, L($(\Sigma)^\star$) = $\Sigma^\star$, set of all strings
7. "b" and "a" are RegEx (lines 2, 1), so "(b ∪ a)" is a RegEx over Σ, L(b ∪ a) = L(b) ∪ L(a)
8. "(b ∪ a)" is a RegEx (line 4), so "((b ∪ a))*" is a RegEx over Σ, L(((b ∪ a))*) = {b,c}*, set of all strings that do not include an "a"
9. "a" and "c" are RegEx (lines 1, 3), so "(a ∘ c)" is a RegEx over Σ, L(a ∘ c) = L(a) ∘ L(b), set of size one
10. "((b ∪ a))*" and "(a ∘ c)" are RegEx (lines 5, 4), so "(((b ∪ a))*∘(a ∘ c))" is a RegEx over Σ, L(((b ∪ a))*) ∘ L(a ∘ c), set of strings where any of "b" or "a" come before the last "a", followed by a "c"
11. "c" and "λ" are RegEx (line 3, 4), so "$(c \cup \lambda)$" is a RegEx over Σ, L($c \cup \lambda$) = L(c) ∪ L(λ), set of strings that include at most 1 (or no) "c's"

DFA

To recognize this language, the DFA needs to remember cases to hold $\omega \in \Sigma^\star$:
List cases (e.g. $\{S_0 = \omega \in \Sigma\star|\omega$ does not include an "a's"}; corresponds to state q_0)

Sanity Check 1: If it has a finite number of subsets

List subsets (e.g. S_0, S_{od}, S_{ev})

Sanity Check 2: Every string belongs to exactly one subset

e.g. Every string must include some fixed number of "a's". Zero ($\omega \in S_0$), odd ($\omega \in S_{od}$), or at least two and even ($\omega \in S_{ev}$). Cannot be in more than one of these sets simultaneously: $S_0 \cap S_{od} = S_0 \cap S_{ev} = S_{od} \cap S_{ev} = \emptyset$

Sanity Check 3: If states are described as above...

e.g. Then, $S_0 \cap L = S_{od} \cap L = \emptyset$, while $S_{ev} \subseteq L$. From this, it follows that q_{ev} is an accepting state, hence, $q_{ev} \in F$

Sanity Check 4: Transitions must be well-defined

e.g. Transitions out of each are as follows:

- Number of "b's" and "c's" don't change the number of "a's" we need:
  - $\{\omega \cdot b|\omega \in S_0\} \subseteq S_0$, $\{\omega \cdot c|\omega \in S_0\} \subseteq S_0$
  - (Same idea for sets S_{od}, S_{ev})

  Thus, retained in the same state.
- Number of "a's" change when you add an "a":
  - Zero to one: $\{\omega \cdot a|\omega \in S_0\} \subseteq S_{od}$, well-defined transition from $q_0 \to q_{od}$ for "a"
  - One to at least two: $\{\omega \cdot a|\omega \in S_{od}\} \subseteq S_{ev}$, well-defined transition from $q_{od} \to q_{ev}$ for "a"
  - Even number to odd that is at least two: $\{\omega \cdot a|\omega \in S_{ev}\} \subseteq S_{od}$, well-defined transition from $q_{ev} \to q_{od}$ for "a"

All sanity checks passed, we know that S_0 is the same as $\{\omega \in \Sigma^\star|\delta^\star(q_0, \omega) = q_0\}$ (Same for others), where $S_0, \ldots$ are the sets of strings described at the beginning of this answer. Since $F = \{q_{ev}\}, L = S_{ev}$ is the language of this DFA - as desired.

DFA NFA Equivalence

Format:

1. 1st state introduced: initally, $\hat{Q} = \emptyset$, $\hat{R} = \{\hat{q_0}\}$, so $\hat{q_0}$ is $Cl\lambda(q_0) = \{q_0\}$.
   Our current DFA: (Drawing of DFA w/ start state)
   We pick $\hat{q_0}$ since it's the only state in $\hat{R}$ so far:
   (a) Transitions out for $\omega = a$ for $\hat{q_0}$
   $\bigcup_{r\in S_0}\bigcup_{s\in\delta(r,a)} Cl\lambda(s) = \bigcup_{r\in\{q_0\}}\bigcup_{s\in\delta(r,a)} Cl\lambda(s)$
   $= \bigcup_{s\in\delta(q_0,a)} Cl\lambda(s)$
   $= \bigcup_{s\in\{q_0\}} Cl\lambda(s)$
   $= Cl\lambda(q_0)$
   $= \{q_0\} = S_0$
   Same as S_0, corresponding to state $\hat{q_0}$. Thus, $\hat{\delta}(\hat{q_0}, a) = \hat{q_0}$
   (b) Transitions out for $\omega = b$ for $\hat{q_0}$ (Same outcome as $\omega = a$)
   (c) Transitions out for $\omega = c$ for $\hat{q_0}$
   $\bigcup_{r\in S_0}\bigcup_{s\in\delta(r,c)} Cl\lambda(s) = \bigcup_{r\in\{q_0\}}\bigcup_{s\in\delta(r,c)} Cl\lambda(s)$
   $= \bigcup_{s\in\delta(q_0,c)} Cl\lambda(s)$
   $= \bigcup_{s\in\{q_0,q_1\}} Cl\lambda(s)$
   $= Cl\lambda(q_0) \cup Cl\lambda(q_1)$
   $= \{q_0\} \cup \{q_1\} = \{q_0, q_1\}$
   This set does not correspond to any states in $\hat{Q} \cup \hat{R}$, so it will be a new set $S_1 = \{q_0, q_1\}$ corresponding to the state $\hat{q_1}$.

   Now, $\hat{Q} = \{\hat{q_0}\}, \hat{R} = \{\hat{q_1}\}$. Our DFA: (draw DFA w/ state $\hat{q_0} \rightarrow \hat{q_1}$)
2. Since only $\hat{R} = \{\hat{q_1}\}$, set state to $\hat{q_1}$
   (a) Transitions out for $\omega = a$ for $\hat{q_1} \rightarrow$ same process, set $S_2 = \{q_0, q_2\}$
   (b) Transitions out for $\omega = b$ for $\hat{q_1} \rightarrow$ corresponds to state $\hat{q_0}$ (b transition back to $\hat{q_0}$)
   (c) Transitions out for $\omega = c$ for $\hat{q_1} \rightarrow$ corresponds to state $\hat{q_1}$ (c transition remain in $\hat{q_1}$)

   Now, $\hat{Q} = \{\hat{q_0}, \hat{q_1}\}, \hat{R} = \{\hat{q_2}\}$. Our DFA: (draw DFA w/ state $\hat{q_2}$)
3. Repeat until no more new states found

- Since $\hat{R} = \emptyset$, we have to find set $\hat{F}$ of accepting states:
  - $S_0 = \{q_0\}$ corresponds to $\hat{q_0}$, and $S_0 \cap F = \{q_0\} \cap \{q_2\} = \emptyset$
  - $S_1 = \{q_0, q_1\}$ corresponds to $\hat{q_1}$, and $S_0 \cap F = \{q_0, q_1\} \cap \{q_2\} = \emptyset$
  - $S_2 = \{q_0, q_2\}$ corresponds to $\hat{q_2}$, and $S_0 \cap F = \{q_0, q_2\} \cap \{q_2\} = \{\hat{q_2}\} \neq \emptyset, \hat{q_2} \in \hat{F}$
  - $S_3 = \{q_0, q_1, q_2\}$ corresponds to $\hat{q_3}$, and $S_0 \cap F = \{q_0, q_1, q_2\} \cap \{q_2\} = \{\hat{q_2}\} \neq \emptyset, \hat{q_3} \in \hat{F}$

  Thus, $\hat{F} = \{\hat{q_2}, \hat{q_3}\}$. Final DFA:

Non-regular languages

Claim:

Proof by Contradiction.

- Suppose that L is a regular language. It follows by the pumping lemma that there exists a positive int p, that for every string $s \in \Sigma^\star$ such that $s \in L$ and $|s| \geq p$, there exists string $x, y, z \in \Sigma^\star$ such that $s = xyz$ and the following properties are satisfied:
  1. $xy^iz \in L$ for every int i such that $i \geq 0$. 2. $y > 0$ (so $y \neq \lambda$). 3. $|xy| \leq p$
- Let $s = a^k$, where $k = 2^p$. Then $s \in L$, since $|s| = k = 2^p = 2^l$ for int $l = p$, and $|s| = 2^p \geq p$. It follows that there exists strings $x, y, z \in \Sigma^\star$ such that $s = xyz$ and properties 1, 2, 3 are all satisfied.
  - Property 3: $|xy| = h$ for some int h such that $0 \leq h \leq p$. Since $s = xyz = a^{(2^p)}$, xy is the prefix, a^h, of len h - and $z = a^{(2^p-h)}$ is the suffix, $a^{(2^p-h)}$, of s with len $2^p - h$
  - Property 2: $|y| > 0$, so that $|y| = l$ for some int l such that $1 \leq l|y| \leq |xy| = h$. Furthermore, y is the suffix, a^l, of xy with len l - and $x = a^{h-l}$ is the prefix, a^{h-l}, of xy with len $|xy| - |y| = h - l$

  In summary, $x = a^{a-l}, y = a^l, z = a^{(2^p-h)}$
  - Property 1: Let $i = 1$. Then, it follows by above that string $xy^iz = xy^2z$ is also in the language l.