Exercise 1

Construct a nondeterministic finite automaton (NFA) to recognize the language

\[ \{ w \in \Sigma^* \mid w \text{ ends in } \text{a, bb, or ccc} \}, \]

where \( \Sigma = \{ \text{a, b, c} \} \)

While it's possible to do this completely deterministically, it's a bit easier if you use the guess-and-check approach from class.

Exercise 2

Construct a nondeterministic finite automaton (NFA) to recognize the language

\[ \{ w \in \Sigma^* \mid \text{some character in } \Sigma \text{ appears at most twice in } w \}, \]

where \( \Sigma = \{ \text{a, b, c} \} \).

While a DFA for this language would require at least 64 states, an NFA needs far fewer. Consider: What would you do if you knew which character was going to appear at most twice? Embrace the nondeterminism!
Exercise 3

Convert the following NFA to an equivalent DFA using the subset construction. You may represent the new DFA as either a state-transition diagram or as a table.

It may be simpler to design a new DFA from scratch, but I want to see that you understand the conversion process that we used to prove that an equivalent DFA exists for every NFA.
Exercise 4

In class, we saw that if you take the DFA for a language $L$ and flip which states are accepting and which are rejecting, you have a DFA for its complement, $\overline{L}$.

a. Prove by contradiction that the same is not true for NFAs. That is, draw a simple NFA for a language $L$ where flipping the accepting and rejecting states does not produce an NFA for $\overline{L}$. Briefly (in 1–2 sentences) justify your answer.

b. Explain why the proof from part (a) doesn’t contradict the fact that the regular languages are closed under complementation.