1. (1 pt) (a) When is an NFA accepting no strings? (b) What is the language of that NFA? Use mathematical notations.

2. (3=2+1 pts) construct DFA and/or NFA for the following bitstring languages (Hint: it might be faster to skip drawing machines for \( L_1 \) or \( L_2 \), and focus on the semantics of the languages):
   (a) \( L_1 = \) those with odd numbers of 0s; \( L_2 = \) those with even numbers of 1s.
      \( L_1 \cup L_2 \) with DFA
      Now \( L_1 - L_2 \) with NFA or DFA:
      (b) those that contain 0100 using either DFA or NFA:

3. (2 pts) Minimize the following DFA using the partition algorithm.
   (a) Initial partition is:
   (b) Final partition is: because:
   (c) Minimal DFA:

4. (3 pts) For this NFA, compute \( \epsilon \)-closure for each state, and convert to DFA.
   (a) \( \epsilon \)-closures:
   (b) DFA:
   (c) \( E(q) \) is defined to be the smallest set such that \( q \in E(q) \) and if \( p \in E(q) \), then:
   (d) Convert this DFA to RE:
5. (2 pts) Given two \( \epsilon \)-free NFAs \( N_1 = (Q_1, \Sigma, \delta_1, q_{01}, F_1) \) and \( N_2 = (Q_2, \Sigma, \delta_2, q_{02}, F_2) \), construct a new NFA \( N = (Q_1 \times Q_2, \Sigma, \delta, q_0, F) \) so that \( L(N) = L(N_1) \cap L(N_2) \) \textbf{without} first converting them to DFAs:

\[
\begin{align*}
q_0 &= q_{01} \times q_{02} \\
\delta((p, q), a) &= \delta_1(p, a) \times \delta_2(q, a) \\
F &= F_1 \times F_2
\end{align*}
\]

6. (3.5 pts) Complete the following partial proof of \( (uv)^R = v^R u^R \) for any strings \( u \) and \( v \).

First define reverse: \( \epsilon^R = \epsilon \), ________________

Now proof by induction on \( |v| \).

**Base case:** \( |v| = 0 \) so: ________________

**Inductive case:** Assume the inductive hypothesis: \( \text{____________________} \) holds for all: \( \text{____________________} \)

Now for \( uv \) where \( |v| = n + 1 \), we can rewrite \( (uv)^R = \text{______________} \)

By definition of reverse, \( \text{____________________} \)

By IH, \( \text{____________________} \)

By definition of reverse, \( \text{____________________} \)

7. (5.5 pts) Write REs/NFAs for the following languages:

(a) (2 pt) Bitstrings that start and end with the same bit (at least one bit).

RE:

Now convert this RE to NFA:

(b) (1 pt) Decimal integers, could be negative (-) but no need to write the plus (+) sign for positive numbers or zero. No leading zeros. Here are three examples: 2561 -89 0. But -0 is not allowed.

To simplify your notation, use \( D = \{0..9\} \) for digits, and \( \bar{D} = \{1..9\} \) for non-zero digits.

RE:

(c) (1.5 pts) An ONID password must have at least one uppercase letter, one lowercase letter, and a digit. \( \text{(hard)} \)

Use \( L = \{a..z\} \) for lowercase letters, \( U = \{A..Z\} \) for uppercase letters, \( D = \{0..9\} \) for digits, and \( \Sigma \) for all characters allowed in an ONID password (i.e., \( \Sigma = L \cup U \cup D \cup \{!, @, #, $, %, ^, \ldots\} \)). \textbf{Draw NFA, but no RE.}

(d) (1 pt) Actually besides the above requirements, a password \textit{also} needs be at least 8 characters long. How would you construct the RE for this kind of password? Just describe your idea, but no need to implement it.