1. For any string $w \in \{0,1\}^*$, let $slash(w)$ be the string in $\{0,1,/\}^*$ obtained from w by inserting a new symbol / between any two consecutive appearances of the same symbol. For example:

$$slash(\varepsilon) = \varepsilon$$
$$slash(0000) = 0/0/0/0$$
$$slash(10101) = 10101$$
$$slash(0010101110) = 0/010101/1/10$$

(a) Let L be an arbitrary regular language over the alphabet $\{0,1\}$. Prove that the language $\textsc{Slash}(L) = \{slash(w) \mid w \in L\}$ is also regular.

Solution: Given a DFA $M = (Q, s, A, \delta)$ for L, we construct a ***DFA*** $M' = (Q', s', A', \delta')$ for $slash(L)$ as follows. The second component of each state indicates the last one or two symbols read, or ε if no symbols have been read.

$$Q' = \big(Q \times \{\varepsilon, 0, 0/, 1, 1/\}\big) \cup \{\text{junk}\}$$
$$s' = (s, \varepsilon)$$
$$A' = A \times \{\varepsilon, 0, 1\}$$

$$\delta'((q,\varepsilon),0) = (\delta(q,0),0)$$
$$\delta'((q,\varepsilon),1) = (\delta(q,1),1)$$

$$\delta'((q,0),1) = (\delta(q,1),1)$$
$$\delta'((q,0),/) = (q,0/)$$

$$\delta'((q,0/),0) = (\delta(q,0),0)$$

$$\delta'((q,1),0) = (\delta(q,0),0)$$
$$\delta'((q,1),/) = (q,1/)$$

$$\delta'((q,1/),1) = (\delta(q,1),1)$$

All unspecified transitions go to the junk state. M' rejects if the input string starts with /, ends with /, or contains any of the substrings 00, 11, //, 0/1, or 1/0. Otherwise, M' passes all 0s and 1s in the input string to M. ■

Rubric: 5 points, standard language transformation rubric (scaled). This is not the only correct solution.

(b) Let L be an arbitrary regular language over the alphabet $\{0, 1, /\}$. Prove that the language $\textsc{Unslash}(L) = \{w \mid \mathit{slash}(w) \in L\}$ is also regular.

Solution: Let $M + (Q, s, A, \delta)$ be any DFA for L. We construct a ***DFA*** $M' = (Q', s', A', \delta')$ for $\textsc{Unslash}(L)$ as follows. The second component of every state of M' is the last symbol read (or ε if no symbols have been read).

$$Q' = Q \times \{\varepsilon, 0, 1\}$$
$$s' = (s, \varepsilon)$$
$$A' = A \times \{\varepsilon, 0, 1\}$$

$$\delta'((q, \varepsilon), 0) = (\delta(q, 0), 0)$$
$$\delta'((q, \varepsilon), 1) = (\delta(q, 1), 1)$$

$$\delta'((q, 0), 0) = (\delta(\delta(q, /), 0), 0)$$
$$\delta'((q, 0), 1) = (\delta(q, 1), 1)$$

$$\delta'((q, 1), 0) = (\delta(q, 0), 0)$$
$$\delta'((q, 1), 1) = (\delta(\delta(q, /), 1), 1)$$

We can describe the transition function δ' more concisely as follows:

$$\delta'((q, a), b) = \begin{cases} (\delta(\delta(q, /), b), b) & \text{if } a = b \\ (\delta(q, b), b) & \text{otherwise} \end{cases}$$

Whenever M' reads the same symbol twice in a row, it sends a / to M between those two identical symbols; otherwise, M' passes its input directly to M. ■

Rubric: 5 points, standard language transformation rubric (scaled). This is not the only correct solution.

2. Describe context-free grammars for the following languages, and clearly explain how they work and the set of strings generated by each nonterminal. Grammars with unclear or missing explanations may receive little or no credit. On the other hand, we do *not* want formal proofs of correctness.

(a) $\left\{ 0^a 1^b 0^c \mid \text{if } a = 1 \text{ then } b = c \right\}$

Solution:

$$\begin{array}{ll} S \to BC \mid 0E \mid ABC & \{0^a 1^b 0^c \mid \text{if } a = 1 \text{ then } b = c\} \\ A \to 00C & \{0^a \mid a \geq 2\} = 000^* \\ B \to \varepsilon \mid 1B & \{1^b \mid b \geq 0\} = 1^* \\ C \to \varepsilon \mid 0C & \{0^c \mid c \geq 0\} = 0^* \\ E \to \varepsilon \mid 1E0 & \{1^b 0^c \mid b = c\} \end{array}$$

The three production rules from S consider strings $0^a 1^b 0^c$ where $a = 0$, $a = 1$, and $a \geq 2$, respectively. ■

Rubric: 4 points = 2 for grammar + 2 for descriptions. This is not the only correct solution.

(b) The set of all palindromes in Σ^* whose lengths are divisible by 5.

Solution: Each nonterminal P_i generates the set of all palindromes w such that $|w| \bmod 5 = i$. The start symbol for this grammar is P_0.

$$\begin{array}{ll} P_0 \to 0P_30 \mid 1P_31 \mid \varepsilon & \text{palindromes with length mod } 5 = 0 \\ P_1 \to 0P_40 \mid 1P_41 \mid 0 \mid 1 & \text{palindromes with length mod } 5 = 1 \\ P_2 \to 0P_00 \mid 1P_01 & \text{palindromes with length mod } 5 = 2 \\ P_3 \to 0P_10 \mid 1P_11 & \text{palindromes with length mod } 5 = 3 \\ P_4 \to 0P_20 \mid 1P_21 & \text{palindromes with length mod } 5 = 4 \end{array}$$

■

Solution: The following grammar has a total of 75 production rules.

$$\begin{array}{lll} S \to E \mid O & & \text{palindromes div by 5} \\ E \to \varepsilon & & \textit{even} \text{ palindromes div by 5} \\ \quad \mid wEw^R & \text{for every string } w \in \Sigma^5 & \\ O \to wOw^R & \text{for every string } w \in \Sigma^5 & \textit{odd} \text{ palindromes div by 5} \\ \quad \mid ab0ba & \text{for all symbols } a, b \in \Sigma & \\ \quad \mid ab1ba & \text{for all symbols } a, b \in \Sigma & \end{array}$$

■

Rubric: 3 points = 1½ for grammar + 1½ for descriptions. These are not the only correct solutions.

(c) Even-length binary strings whose first half contains an odd number of 1s. More formally:

$$\left\{ w \in \Sigma^* \;\middle|\; \begin{array}{c} w = xy \text{ for some strings } x \text{ and } y \text{ such that} \\ |x| = |y| \text{ and } \#(1, x) \text{ is odd} \end{array} \right\}$$

Solution:

$$\begin{aligned} S &\to 0S0 \mid 0S1 \mid 1E0 \mid 1E1 && \text{odd 1s in first half} \\ E &\to 0E0 \mid 0E1 \mid 1S0 \mid 1S1 \mid \varepsilon && \text{even 1s in first half} \end{aligned}$$

The second nonterminal E generates the set of even-length binary strings whose first half contains an ***Even*** number of 1s. ■

Rubric: 3 points = 1½ for grammar + 1½ for descriptions. This is not the only correct solution.

(d) ***Practice only. Do not submit solutions.***

Strings in which the substrings `00` and `11` appear the same number of times. For example, `1100011` $\in L$ because both substrings appear twice, but `01000011` $\notin L$.

Solution (counting): Let $\#(\mathtt{11}, w)$ denote the number of times `11` appears as a substring of w, and let $\#(\mathtt{1}^+, w)$ denote the number of runs of `1`s in w. For example:

$$\#(\mathtt{11}, \mathtt{11110011101}) = 5 \qquad \#(\mathtt{1}^*, \mathtt{11110011101}) = 3$$

Each `1` in a binary string is the beginning of a `11` substring, except for the last `1` in every run; it follows that

$$\#(\mathtt{11}, w) = \#(\mathtt{1}, w) - \#(\mathtt{1}^+, w)$$

for every string w. Symmetric arguments imply $\#(\mathtt{00}, w) = \#(\mathtt{0}, w) - \#(\mathtt{0}^*, w)$ for every string w. Thus

$$\#(\mathtt{00}, w) = \#(\mathtt{11}, w)$$
$$\iff$$
$$\#(\mathtt{0}, w) - \#(\mathtt{0}^+, w) = \#(\mathtt{1}, w) - \#(\mathtt{1}^+, w)$$
$$\iff$$
$$\#(\mathtt{0}, w) - \#(\mathtt{1}, w) = \#(\mathtt{0}^+, w) - \#(\mathtt{1}^+, w)$$

But because runs of `0`s and `1`s in any binary string w alternate, the number of `0`-runs and the number of `1`-runs always differ by at most 1. More specifically:

- If w starts and ends with `0`, then $\#(\mathtt{1}^+, w) = \#(\mathtt{0}^+, w) - 1$.
- If the first and last symbols in w are different, then $\#(\mathtt{1}^+, w) = \#(\mathtt{0}^+, w)$.
- If w starts and ends with `1`, then $\#(\mathtt{1}^+, w) = \#(\mathtt{0}^+, w) + 1$.

Let $\Delta(w) = \#(\mathtt{1}, w) - \#(\mathtt{0}, w)$. The preceding case analysis implies that our target language L contains a binary string w if and only if w satisfies one of the following conditions:

- $w = \varepsilon$
- w starts with `0` and ends with `1`, and $\Delta(w) = 0$
- w starts with `1` and ends with `0`, and $\Delta(w) = 0$
- w starts with `0` and ends with `0`, and $\Delta(w) = -1$. In this case, dropping the final `0` leaves a string with equal `0`s and `1`s. So there are two subcases:
  - $w = \mathtt{0}$
  - w contains at least one `1`. Then w must have a prefix x that starts with `0` and ends with `1`, such that $\Delta(x) = 0$. We can write $w = xy\mathtt{0}$ for some (possibly empty) string y with $\Delta(y) = 0$.
- w starts with `1` and ends with `1`, and $\Delta(w) = 1$. This case is symmetric to the previous case.

Finally, we can write down a grammar that follows the preceding case analysis.

$S \to \varepsilon \mid A \mid B \mid C \mid D$	target language L
$A \to 0E1$	starts with 0, ends with 1, and $\Delta = 0$
$B \to 1E0$	starts with 1, ends with 0, and $\Delta = 0$
$C \to 0 \mid AE0$	starts with 0, ends with 0, and $\Delta = -1$
$D \to 1 \mid BE1$	starts with 1, ends with 1, and $\Delta = +1$
$E \to \varepsilon \mid EE \mid 0E1 \mid 1E0$	$\Delta = 0$ (from the lecture notes)

This grammar can be written more compactly as follows:

$S \to \varepsilon \mid 0 \mid 1 \mid 0E1 \mid 1E0 \mid 0E1E0 \mid 1E0E1$	$\#11 = \#00$
$E \to \varepsilon \mid EE \mid 0E1 \mid 1E0$	$\#1 = \#0$

■

*3. ***Practice only. Do not submit solutions.***

Let L_1 and L_2 be arbitrary regular languages over the alphabet $\Sigma = \{0, 1\}$. Prove that the following languages are also regular.

(a) $\textsc{Faro}(L_1, L_2) := \{faro(x, z) \mid x \in L_1 \text{ and } z \in L_2 \text{ with } |x| = |z|\}$, where

$$faro(x,z) := \begin{cases} z & \text{if } x = \varepsilon \\ a \cdot faro(z, y) & \text{if } x = ay \end{cases}$$

Solution: Let $M_1 = (Q_1, s_1, A_1, \delta_1)$ and $M_2 = (Q_2, s_2, A_2, \delta_2)$ be arbitrary DFAs that accepts the regular languages L_1 and L_2, respectively. We build a **DFA** $M = (Q, s, A, \delta)$ that accepts $\textsc{Faro}(L_1, L_2)$ using the following modified product construction:

$$\begin{aligned} Q &= Q_1 \times Q_2 \times \{1, 2\} \\ s &= (s_1, s_2, 1) \\ A &= A_1 \times A_2 \times \{1\} \end{aligned}$$

$$\begin{aligned} \delta((q_1, q_2, 1), a) &= (\delta(q_1, a),\ q_2,\ 2) \\ \delta((q_1, q_2, 2), a) &= (q_1,\ \delta(q_2, a),\ 1) \end{aligned}$$

M reads the input string and alternately passes its input bits to simulations of M_1 and M_2. State (q_1, q_2, i) means that machine M_1 is in state q_1, machine M_2 is in state q_2, and the next input bit should be passed to M_i. The condition $|w| = |z|$ in the definition of $faro(w, z)$ implies that M can accept only if it has read an even number of bits. ■

(b) $\textsc{Shuffles}(L_1, L_2) := \bigcup_{w \in L_1, y \in L_2} \mathit{shuffles}(w, y)$, where $\mathit{shuffles}(w, y)$ is the set of all strings obtained by shuffling w and y, or equivalently, all strings in which w and y are complementary subsequences. Formally:

$$\mathit{shuffles}(w, y) = \begin{cases} \{y\} & \text{if } w = \varepsilon \\ \{w\} & \text{if } y = \varepsilon \\ \{a\} \bullet \mathit{shuffles}(x, y) \cup \{b\} \bullet \mathit{shuffles}(w, z) & \text{if } w = ax \text{ and } y = bz \end{cases}$$

Solution: Let $M_1 = (Q_1, s_1, A_1, \delta_1)$ and $M_2 = (Q_2, s_2, A_2, \delta_2)$ be arbitrary DFAs that accepts the regular languages L_1 and L_2, respectively. We build a **NFA** $M = (Q, s, A, \delta)$ that accepts $\textsc{Shuffles}(L_1, L_2)$ using the following modified product construction:

$$\begin{aligned} Q &= Q_1 \times Q_2 \\ s &= (s_1, s_2) \\ A &= A_1 \times A_2 \end{aligned}$$

$$\delta((q_1, q_2), a) = \big\{(\delta_1(q_1, a), q_2),\ (q_1, \delta_2(q_2, a))\big\}$$

Each time M reads a bit, it nondeterministically guesses whether to pass that bit to M_1 or M_2. ■