1. Describe context-free grammars for the following languages over the alphabet $\Sigma = \{0, 1\}$. For each non-terminal in your grammars, describe in English the language generated by that non-terminal.

(a) $\{0^m 1^n \mid m > n\}$

Solution:

$S \to 0S \mid 0A$ $\quad\{0^m 1^n \mid m > n\}$

$A \to \varepsilon \mid 0A1$ $\quad\{0^m 1^n \mid m = n\}$

■

Solution:

$S \to 0S1 \mid A$ $\quad\{0^m 1^n \mid m > n\}$

$A \to 0 \mid 0A$ $\quad\{0^m \mid m > 0\}$

■

Rubric: 2 points = 1 for grammar + 1 for descriptions. These are not the only correct solutions.

(b) The set of all palindromes in $(0+1)^*$ whose length is divisible by 7.

Solution: The start symbol for this grammar is P_0.

$P_0 \to 0P_50 \mid 1P_51 \mid \varepsilon$ — palindromes with length mod $7 = 0$

$P_1 \to 0P_60 \mid 1P_61 \mid 0 \mid 1$ — palindromes with length mod $7 = 1$

$P_2 \to 0P_00 \mid 1P_01$ — palindromes with length mod $7 = 2$

$P_3 \to 0P_11 \mid 1P_11$ — palindromes with length mod $7 = 3$

$P_4 \to 0P_21 \mid 1P_21$ — palindromes with length mod $7 = 4$

$P_5 \to 0P_31 \mid 1P_31$ — palindromes with length mod $7 = 5$

$P_6 \to 0P_41 \mid 1P_41$ — palindromes with length mod $7 = 6$

■

Solution: The following grammar has a total of 275 production rules.

$S \to E \mid O$ — palindromes div by 7

$E \to \varepsilon$ — *even* palindromes div by 7

$\quad\mid wEw^R$ for every string $w \in \Sigma^7$

$O \to wOw^R$ for every string $w \in \Sigma^7$ — *odd* palindromes div by 7

$\quad\mid w0w^R$ for every string $w \in \Sigma^3$

$\quad\mid w1w^R$ for every string $w \in \Sigma^3$

■

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

(c) $\left\{\mathtt{0}^a\mathtt{1}^b \;\middle|\; a \neq 2b \text{ and } b \neq 2a\right\}$

Solution:

$$
\begin{array}{ll}
S \to A \mid B \mid C & \left\{\mathtt{0}^a\mathtt{1}^b \;\middle|\; a \neq 2b \text{ and } b \neq 2a\right\} \\[1ex]
A \to \mathtt{0}A \mid \mathtt{0}E & \left\{\mathtt{0}^a\mathtt{1}^b \;\middle|\; a > 2b\right\} \\
E \to \mathtt{00}E\mathtt{1} \mid \varepsilon & \left\{\mathtt{0}^a\mathtt{1}^b \;\middle|\; a = 2b\right\} \\[1ex]
B \to B\mathtt{1} \mid F\mathtt{1} & \left\{\mathtt{0}^a\mathtt{1}^b \;\middle|\; b > 2a\right\} \\
F \to \mathtt{0}F\mathtt{11} \mid \varepsilon & \left\{\mathtt{0}^a\mathtt{1}^b \;\middle|\; b = 2a\right\} \\[1ex]
C \to \mathtt{0}C\mathtt{1} \mid \mathtt{00}C\mathtt{1} \mid \mathtt{0}C\mathtt{11} \mid \mathtt{01} & \left\{\mathtt{0}^a\mathtt{1}^b \;\middle|\; a < 2b \text{ and } b < 2a\right\}
\end{array}
$$

■

We can formally justify the last non-terminal as follows. Let $L = \{\mathtt{0}^a\mathtt{1}^b \mid b/2 < a < 2b\}$; we claim that L is the language generated by C.

- First, let w be an arbitrary string generated by C. Assume that every string x that is generated by C and is shorter than w lies in L.
  - Suppose $w = \mathtt{0}x\mathtt{1}$ for some string x generated by C. The induction hypothesis implies $x \in L$ and therefore $x = \mathtt{0}^a\mathtt{1}^b$ for some integers a and b where $b/2 < a < 2b$. Thus, $w = \mathtt{0}^{a+1}\mathtt{1}^{b+1}$. We easily confirm that $(b+1)/2 = b/2 + 1/2 < a+1 < 2b+2 = 2(b+1)$. It follows that $w \in L$.
  - Suppose $w = \mathtt{00}x\mathtt{1}$ for some string x generated by C. The induction hypothesis implies $x \in L$ and therefore $x = \mathtt{0}^a\mathtt{1}^b$ for some integers a and b where $b/2 < a < 2b$. Thus, $w = \mathtt{0}^{a+2}\mathtt{1}^{b+1}$. We easily confirm that $(b+1)/2 = b/2 + 1/2 < a+2 < 2b+2 = 2(b+1)$. It follows that $w \in L$.
  - Suppose $w = \mathtt{0}x\mathtt{11}$ for some string x generated by C. The induction hypothesis implies $x \in L$ and therefore $x = \mathtt{0}^a\mathtt{1}^b$ for some integers a and b where $b/2 < a < 2b$. Thus, $w = \mathtt{0}^{a+1}\mathtt{1}^{b+2}$. We easily confirm that $(b+2)/2 = b/2 + 1 < a+1 < 2b+4 = 2(b+4)$. It follows that $w \in L$.
  - Finally, $\mathtt{01} \in L$, because $1/2 < 1 < 2$.

  We conclude that every string generated by C is in L.
- Now let w be an arbitrary string in L; by definition $w = \mathtt{0}^a\mathtt{1}^b$ for some integers a and b such that $b/2 < a < 2b$. Assume without loss of generality that any string in L that is shorter than w is generated by C. There are several cases to consider.
  - If $a = 1$ and $b = 1$, then $w = \mathtt{01}$, which is clearly generated by C. In all other cases, we can assume $a > 1$ and $b > 1$.
  - Suppose $a = 2b-1$. Let $x = \mathtt{0}^{a-2}\mathtt{1}^{b-1}$. We easily verify that $(b-1)/2 < a-2 < 2(b-1)$. (The first inequality requires $b > 1$.) Thus, $x \in L$, so

the induction hypothesis implies x is generated by C. So $w = \texttt{00}x\texttt{1}$ is generated by C.

- Suppose $b = 2a-1$. Let $x = \texttt{0}^{a-1}\texttt{1}^{b-2}$. We easily verify that $(b-2)/2 < a-1 < 2(b-2)$. (The second inequality requires $a > 1$.) Thus, $x \in L$, so the induction hypothesis implies x is generated by C. So $w = \texttt{0}x\texttt{11}$ is generated by C.
- Finally, suppose $b < 2a-1$ and $a < 2b-1$. Let $x = \texttt{0}^{a-1}\texttt{1}^{b-1}$. We easily verify that the case boundaries imply $(b-1)/2 < a-1 < 2(b-1)$. Thus, $x \in L$, so the induction hypothesis implies x is generated by C. So $w = \texttt{0}x\texttt{1}$ is generated by C.

We conclude that every string in L is generated by C.

Rubric: 5 points = 2½ for grammar + 2½ for descriptions/justification. The formal proof of correctness (grayed out) is *not* required for full credit. This is not the only correct solution.

2. For any string w, let $contract(w)$ denote the string obtained by collapsing each maximal substring of equal symbols to one symbol. For example:

$$\begin{aligned} contract(010101) &= 010101 \\ contract(001110) &= 010 \\ contract(111111) &= 1 \\ contract(1) &= 1 \\ contract(\varepsilon) &= \varepsilon \end{aligned}$$

Prove that for every regular language L over the alphabet $\{0,1\}$, the following languages are also regular:

(a) $contract(L) = \{contract(w) \mid w \in L\}$

Solution: Let L be an arbitrary regular language, and let $M = (Q, s, A, \delta)$ be a DFA that accepts L. We construct an NFA $M' = (Q', s', A', \delta')$ with ε-transitions that accepts $contract(L)$ as follows:

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

s' is an explicit state in Q'

$$A' = \begin{cases} \{s'\} \cup (A \times \{0,1\}) & \text{if } s \in A \\ A \times \{0,1\} & \text{if } s \notin A \end{cases}$$

$$\delta'(s', \varepsilon) = \varnothing$$

$$\delta'(s', a) = \big\{(\delta(s,a), a)\big\} \qquad \text{for all } a \in \{0,1\}$$

$$\delta'((q,a), \varepsilon) = \big\{(\delta(q,a), a)\big\} \qquad \text{for all } q \in Q \text{ and } a \in \{0,1\}$$

$$\delta'((q,a), b) = \begin{cases} \varnothing & \text{if } a = b \\ \big\{(\delta(q,b), b)\big\} & \text{otherwise} \end{cases} \qquad \text{for all } q \in Q \text{ and } a, b \in \{0,1\}$$

Our machine M' simulates M, but whenever M' reads a symbol, it simulates M reading all possible non-empty runs of that symbol, and whenever M' reads two of the same symbol in a row, it rejects. Each state (q,a) of M' indicates that M is in state q and has just read symbol a. The ε-transition from (q,a) simulates M reading the same symbol a again. Finally, M' needs an explicit start state s' with no ε-transitions to reflect the fact that M has not read anything yet. ∎

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

(b) $contract^{-1}(L) = \{w \in \{0, 1\}^* \mid contract(w) \in L\}$

Solution: Let L be an arbitrary regular language, and let $M = (Q, s, A, \delta)$ be a DFA that accepts L. We construct a **DFA** $M' = (Q', s', A', \delta')$ that accepts $contract(L)$ as follows:

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

s' is an explicit state in Q'

$$A' = \begin{cases} \{s'\} \cup (A \times \{0, 1\}) & \text{if } s \in A \\ A \times \{0, 1\} & \text{if } s \notin A \end{cases}$$

$$\delta'(s', a) = (\delta(s, a), a) \qquad \text{for all } a \in \{0, 1\}$$

$$\delta'((q, a), b) = \begin{cases} q & \text{if } a = b \\ \delta(q, b) & \text{otherwise} \end{cases} \qquad \text{for all } q \in Q \text{ and } a, b \in \{0, 1\}$$

Our new DFA M' simulates M, but whenever its reads the first symbol in a run, it effectively ignores the rest of that run. Each state (q, a) of M' indicates that M is in state q and has just read symbol a; the transition function keeps M' in the same state if it reads the same symbol again. Finally, M' needs an explicit start state s' to reflect the fact that M has not read anything yet. ■

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

3. For any string w, let $oneswap(w)$ be the set of all strings obtained by swapping exactly one pair of adjacent symbols in w. For any language L, define a new language $oneswap(L)$ as follows:

$$oneswap(L) := \bigcup_{w \in L} oneswap(w)$$

Prove that if L is a regular language over the alphabet $\{0, 1\}$, the language $oneswap(L)$ is also regular.

Solution: Let L be an arbitrary regular language, and let $M = (Q, s, A, \delta)$ be a DFA that accepts L. We construct an NFA $M' = (Q', s', A', \delta')$ ***without*** ε-transitions that accepts $contract(L)$ as follows:

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

$$\delta'((q, notyet), a) = \big\{(\delta(q,a), notyet),\ (q,a)\big\} \qquad \text{for all } q \in Q \text{ and } a \in \{0,1\}$$
$$\delta'((q,a), b) = \big\{(\delta(\delta(q,b),a), done)\big\} \qquad \text{for all } q \in Q \text{ and } a, b \in \{0,1\}$$
$$\delta'((q, done), a) = \big\{(\delta(q,a), done)\big\} \qquad \text{for all } q \in Q \text{ and } a \in \{0,1\}$$

Our new DFA M' simulates M, but nondeterministically chooses a time to swap two adjacent input symbols. The states of M' have the following meanings:

- $(q, notyet)$ — M is in state q, and M' has not swapped anything
- $(q, 0)$ — M is in state q, and M' just read a 0, which it is about to swap with the next symbol
- $(q, 1)$ — M is in state q, and M' just read a 1, which it is about to swap with the next symbol
- $(q, done)$ — M is in state q, and M' has already swapped one pair of symbols

The only nondeterminism in M' is in deciding whether or not to swap the next two input symbols. ■

Rubric: 10 points: standard language transformation rubric. This is not the only correct solution.