Give context-free grammars for each of the following languages.

1. \( \{0^{2n}1^n \mid n \geq 0\} \)

   Solution: \( S \to \varepsilon | \emptyset S 1 \)

2. \( \{0^m1^n \mid m \neq 2n \} \)

   [Hint: If \( m \neq 2n \), then either \( m < 2n \) or \( m > 2n \).]

   Solution: To simplify notation, let \( \Delta(w) = \#(\emptyset, w) - 2\#(1, w) \). Our solution follows the following logic. Let \( w \) be an arbitrary string in this language.

   - Because \( \Delta(w) \neq 0 \), then either \( \Delta(w) > 0 \) or \( \Delta(w) < 0 \).
   - If \( \Delta(w) > 0 \), then \( w = \emptyset^i z \) for some integer \( i > 0 \) and some suffix \( z \) with \( \Delta(z) = 0 \).
   - If \( \Delta(w) < 0 \), then \( w = x 1^j \) for some integer \( j > 0 \) and some prefix \( x \) with either \( \Delta(x) = 0 \) or \( \Delta(x) = 1 \).
   - Substrings with \( \Delta = 0 \) is generated by the previous grammar; we need only a small tweak to generate substrings with \( \Delta = 1 \).

   Here is one way to encode this case analysis as a CFG. The nonterminals \( M \) and \( L \) generate all strings where the number of \( \emptyset \)s is More or Less than twice the number of \( 1 \)s, respectively. The last nonterminal generates strings with \( \Delta = 0 \) or \( \Delta = 1 \).

   \[
   \begin{align*}
   S & \rightarrow M \mid L & \{0^m1^n \mid m \neq 2n\} \\
   M & \rightarrow \emptyset M \mid \emptyset E \{0^m1^n \mid m > 2n\} \\
   L & \rightarrow L1 \mid E1 & \{0^m1^n \mid m < 2n\} \\
   E & \rightarrow \varepsilon \mid \emptyset \emptyset E1 \{0^m1^n \mid m = 2n \text{ or } 2n+1\}
   \end{align*}
   \]

   Here is a different correct solution using the same logic. We either identify a non-empty prefix of \( \emptyset \)s or a non-empty suffix of \( 1 \)s, so that the rest of the string is as “balanced” as possible. We also generate strings with \( \Delta = 1 \) using a separate non-terminal.

   \[
   \begin{align*}
   S & \rightarrow AE \mid EB \mid FB & \{0^m1^n \mid m \neq 2n\} \\
   A & \rightarrow \emptyset \mid \emptyset A & \emptyset^+ = \{\emptyset^i \mid i \geq 1\} \\
   B & \rightarrow 1 \mid 1B & 1^+ = \{1^j \mid j \geq 1\} \\
   E & \rightarrow \varepsilon \mid \emptyset \emptyset E1 & \{0^m1^n \mid m = 2n\} \\
   F & \rightarrow \emptyset E & \{0^m1^n \mid m = 2n+1\}
   \end{align*}
   \]

   Alternatively, we can separately generate all strings of the form \( 0^{\text{odd}} 1^* \), so that we don’t have to worry about the case \( \Delta = 1 \) separately.

   \[
   \begin{align*}
   S & \rightarrow D \mid M \mid L & \{0^m1^n \mid m \neq 2n\} \\
   D & \rightarrow \emptyset \mid \emptyset \emptyset D \mid D1 & \{0^m1^n \mid m \text{ is odd}\} \\
   M & \rightarrow \emptyset M \mid \emptyset E & \{0^m1^n \mid m > 2n\} \\
   L & \rightarrow L1 \mid E1 & \{0^m1^n \mid m < 2n \text{ and } m \text{ is even}\} \\
   E & \rightarrow \varepsilon \mid \emptyset \emptyset E1 & \{0^m1^n \mid m = 2n\}
   \end{align*}
   \]

\[\blacksquare\]
Solution: Intuitively, we can parse any string \( w \in L \) as follows. First, remove the first \( 2k \) 0s and the last \( k \) 1s, for the largest possible value of \( k \). The remaining string cannot be empty, and it must consist entirely of 0s, entirely of 1s, or a single 0 followed by 1s.

\[
S \rightarrow 00S1 | A | B | C \quad \{0^n1^m | m \neq 2n\}
\]

\[
A \rightarrow 0 | 0A \quad 0^+
\]

\[
B \rightarrow 1 | 1B \quad 1^+
\]

\[
C \rightarrow 0 | 0B \quad 01^*
\]

3. \( \{0, 1\}^* \setminus \{0^{2n}1^n | n \geq 0\} \)

Solution: This language is the union of the previous language and the complement of \( 0^*1^* \), which is \((0 + 1)^*10(0 + 1)^*\).

\[
S \rightarrow T | X \quad \{0, 1\}^* \setminus \{0^{2n}1^n | n \geq 0\}
\]

\[
T \rightarrow 00T1 | A | B | C \quad \{0^n1^m | m \neq 2n\}
\]

\[
A \rightarrow 0 | 0A \quad 0^+
\]

\[
B \rightarrow 1 | 1B \quad 1^+
\]

\[
C \rightarrow 0 | 0B \quad 01^*
\]

\[
X \rightarrow Z10Z \quad (0 + 1)^*10(0 + 1)^*
\]

\[
Z \rightarrow \varepsilon | 0Z | 1Z \quad (0 + 1)^*
\]
Work on these later:

4. \( \{ w \in \{0, 1\}^* \mid \#(\emptyset, w) = 2 \cdot \#(1, w) \} \) — Binary strings where the number of 0s is exactly twice the number of 1s.

**Solution:** \( S \rightarrow \varepsilon \mid SS \mid 00S1 \mid 0S1S0 \mid 1S00 \).

Here is a sketch of a correctness proof; a more detailed proof appears in the homework.

For any string \( w \), let \( \Delta(w) = \#(\emptyset, w) - 2 \cdot \#(1, w) \). Suppose \( w \) is a binary string such that \( \Delta(w) = 0 \). Suppose \( w \) is nonempty and has no non-empty proper prefix \( x \) such that \( \Delta(x) = 0 \). There are three possibilities to consider:

- Suppose \( \Delta(x) > 0 \) for every proper prefix \( x \) of \( w \). In this case, \( w \) must start with \( \emptyset \emptyset \) and end with \( 1 \). Thus, \( w = \emptyset \emptyset x 1 \) for some string \( x \in L \).
- Suppose \( \Delta(x) < 0 \) for every proper prefix \( x \) of \( w \). In this case, \( w \) must start with \( 1 \) and end with \( \emptyset \emptyset \). Let \( x \) be the shortest non-empty proper prefix with \( \Delta(x) = 1 \). Thus, \( w = 1X\emptyset \emptyset \) for some string \( x \in L \).
- Finally, suppose \( \Delta(x) > 0 \) for some prefix \( x \) and \( \Delta(x') < 0 \) for some longer proper prefix \( x' \). Let \( x' \) be the shortest non-empty proper prefix of \( w \) with \( \Delta < 0 \). Then \( x' = \emptyset y 1 \) for some substring \( y \) with \( \Delta(y) = 0 \), and thus \( w = \emptyset y 1z\emptyset \) for some strings \( y, z \in L \).

5. \( \{0, 1\}^* \setminus \{ww \mid w \in \{0, 1\}^*\} \).

**Solution:** All strings of odd length are in \( L \).

Let \( w \) be any even-length string in \( L \), and let \( m = |w|/2 \). For some index \( i \leq m \), we have \( w_i \neq w_{m+i} \). Thus, \( w \) can be written as either \( x1y\emptyset z \) or \( x\emptyset y1z \) for some substrings \( x, y, z \) such that \( |x| = i - 1, |y| = m - 1, \) and \( |z| = m - i \). We can further decompose \( y \) into a prefix of length \( i - 1 \) and a suffix of length \( m - i \). So we can write any even-length string \( w \in L \) as either \( x1x'z'\emptyset z \) or \( x\emptyset x'z'1z \), for some strings \( x, x', z, z' \) with \( |x| = |x'| = i - 1 \) and \( |z| = |z'| = m - i \). Said more simply, we can divide \( w \) into two odd-length strings, one with a \( \emptyset \) at its center, and the other with a \( 1 \) at its center.

\[
\begin{align*}
S & \rightarrow AB \mid BA \mid A \mid B & \text{strings not of the form } ww \\
A & \rightarrow \emptyset \mid \Sigma A \Sigma & \text{odd-length strings with } \emptyset \text{ at center} \\
B & \rightarrow 1 \mid \Sigma B \Sigma & \text{odd-length strings with } 1 \text{ at center} \\
\Sigma & \rightarrow \emptyset \mid 1 & \text{single character}
\end{align*}
\]