1. (a) Prove that the following languages are not regular by providing a fooling set. You need to provide an infinite set and also prove that it is a valid fooling set for the given language. Alternatively, you can describe a fooling set \( F_n \) of size \( n \) for every \( n > 0 \) and prove its validity.

i. \( L = \{0^i 1^j 2^k | i + j = k + 1\} \).

ii. Recall that a block in a string is a maximal non-empty substring of identical symbols. Let \( L \) be the set of all strings in \( \{0, 1\}^*\) that contain two non-empty blocks of \( 0s \) of unequal length. For example, \( L \) contains the strings \( 011001111 \) and \( 00100100111000100 \) but does not contain the strings \( 000110001100011 \) and \( 00000000111 \).

iii. \( L = \{0^\lceil n \log_2 n \rceil | n \geq 1\} \).

(b) Let \( L_k = \{w \in \{0, 1\}^*: |w| \geq 2k \text{ and last } 2k \text{ characters of } w \text{ have unequal number of } 0s \text{ and } 1s\}. \) If \( k = 3 \) then \( 0001111 \) and \( 01000110 \) are in \( L_3 \) while \( 010011 \) and \( 000111000 \) are not. Describe a fooling set for \( L_k \) of size at least \( 2^k \) and prove that it is valid.

Not to submit for grading: Design an NFA for \( L_k \) with \( O(k^2) \) states.

(c) Suppose \( L \) is not regular and \( L' \) is a finite language. Prove that \( L \setminus L' \) is not regular. Give a simple example of a non-regular language \( L \) and a regular language \( L' \) such that \( L \setminus L' \) is regular.

2. Describe a context free grammar for the following languages. Clearly explain how they work and the role of each non-terminal. Unclear grammars will receive little to no credit.

(a) \( L = \{a^i b^j c^k d^\ell | i + j = k + \ell\} \)

(b) \( L = \{0^i 1^j 2^k | k = 3(i + j)\} \)

(c) \( L = \{x_1 \# x_2 \# \ldots \# x_k | k \geq 1, \text{each } x_i \in \{0, 1\}^*, \text{and for some } i \text{ and } j, x_i = x_j^R\} \). Note that \( i \) can be equal to \( j \) in the definition and there can be multiple pairs that satisfy the condition. Here the terminal set \( T \) is \( \{0, 1, \#\} \).

(d) \( L = \{0, 1\}^* \setminus \{1^n 0^n | n \geq 0\} \), in other words the complement of the language \( L' = \{1^n 0^n | n \geq 0\} \). Note that \( L' \) is not regular but context free. The complement of a context free language is not necessarily context free, but it is true for this particular language \( L' \).

3. Not to submit: Consider all regular expressions over an alphabet \( \Sigma \). Each regular expression is a string over a larger alphabet \( \Sigma' = \Sigma \cup \{\emptyset\text{-Symbol}, \epsilon\text{-Symbol}, +, (, )\} \). We use \( \emptyset\text{-Symbol} \) and \( \epsilon\text{-Symbol} \) in place of \( \emptyset \) and \( \epsilon \) to avoid confusion with overloading; technically one should do it with \( +, (, ) \) as well. Let \( R_\Sigma \) be the language of regular expressions over \( \Sigma \).

(a) Prove that \( R_\Sigma \) is not regular.

(b) Describe a context free grammar (CFG) for \( R_\Sigma \) which will prove that it is a CFL.

This shows that we need more expressive languages than regular languages to describe regular expressions.
Solved problem

4. Let $L$ be the set of all strings over $\{0, 1\}^*$ with exactly twice as many $0$s as $1$s.

(a) Describe a CFG for the language $L$.

[Hint: For any string $u$ define $\Delta(u) = \#(0, u) - 2\#(1, u)$. Introduce intermediate variables that derive strings with $\Delta(u) = 1$ and $\Delta(u) = -1$ and use them to define a non-terminal that generates $L$.]

Solution: $S \rightarrow \varepsilon \mid SS \mid 00S1 \mid 0S1S0 \mid 1S00$

(b) Prove that your grammar $G$ is correct. As usual, you need to prove both $L \subseteq L(G)$ and $L(G) \subseteq L$.

[Hint: Let $u_{\leq i}$ denote the prefix of $u$ of length $i$. If $\Delta(u) = 1$, what can you say about the smallest $i$ for which $\Delta(u_{\leq i}) = 1$? How does $u$ split up at that position? If $\Delta(u) = -1$, what can you say about the smallest $i$ such that $\Delta(u_{\leq i}) = -1$?]

Solution: We separately prove $L \subseteq L(G)$ and $L(G) \subseteq L$ as follows:

Claim 1. $L(G) \subseteq L$, that is, every string in $L(G)$ has exactly twice as many $0$s as $1$s.

Proof: As suggested by the hint, for any string $u$, let $\Delta(u) = \#(0, u) - 2\#(1, u)$. We need to prove that $\Delta(w) = 0$ for every string $w \in L(G)$.

Let $w$ be an arbitrary string in $L(G)$, and consider an arbitrary derivation of $w$ of length $k$. Assume that $\Delta(x) = 0$ for every string $x \in L(G)$ that can be derived with fewer than $k$ productions.\(^1\) There are five cases to consider, depending on the first production in the derivation of $w$.

- If $w = \varepsilon$, then $\#(\varepsilon, w) = \#(1, w) = 0$ by definition, so $\Delta(w) = 0$.
- Suppose the derivation begins $S \rightarrow SS \twoheadrightarrow^* w$. Then $w = xy$ for some strings $x, y \in L(G)$, each of which can be derived with fewer than $k$ productions. The inductive hypothesis implies $\Delta(x) = \Delta(y) = 0$. It immediately follows that $\Delta(w) = 0$.\(^2\)
- Suppose the derivation begins $S \rightarrow 00S1 \twoheadrightarrow^* w$. Then $w = 00x1$ for some string $x \in L(G)$. The inductive hypothesis implies $\Delta(x) = 0$. It immediately follows that $\Delta(w) = 0$.
- Suppose the derivation begins $S \rightarrow 1S00 \twoheadrightarrow^* w$. Then $w = 1x00$ for some string $x \in L(G)$. The inductive hypothesis implies $\Delta(x) = 0$. It immediately follows that $\Delta(w) = 0$.
- Suppose the derivation begins $S \rightarrow 0S1S1 \twoheadrightarrow^* w$. Then $w = 0x1y0$ for some strings $x, y \in L(G)$. The inductive hypothesis implies $\Delta(x) = \Delta(y) = 0$. It immediately follows that $\Delta(w) = 0$.

In all cases, we conclude that $\Delta(w) = 0$, as required. \(\square\)

Claim 2. $L \subseteq L(G)$; that is, $G$ generates every binary string with exactly twice as many $0$s as $1$s.

---

\(^1\)Alternatively: Consider the shortest derivation of $w$, and assume $\Delta(x) = 0$ for every string $x \in L(G)$ such that $|x| < |w|$.

\(^2\)Alternatively: Suppose the shortest derivation of $w$ begins $S \rightarrow SS \twoheadrightarrow^* w$. Then $w = xy$ for some strings $x, y \in L(G)$. Neither $x$ or $y$ can be empty, because otherwise we could shorten the derivation of $w$. Thus, $x$ and $y$ are both shorter than $w$, so the induction hypothesis implies. . . . We need some way to deal with the decompositions $w = \varepsilon \ast w$ and $w = w \ast \varepsilon$, which are both consistent with the production $S \rightarrow SS$, without falling into an infinite loop.
Proof: As suggested by the hint, for any string $u$, let $\Delta(u) = #(0, u) - 2#(1, u)$. For any string $u$ and any integer $0 \leq i \leq |u|$, let $u_i$ denote the $i$th symbol in $u$, and let $u_{\leq i}$ denote the prefix of $u$ of length $i$.

Let $w$ be an arbitrary binary string with twice as many 0s as 1s. Assume that $G$ generates every binary string $x$ that is shorter than $w$ and has twice as many 0s as 1s. There are two cases to consider:

- If $w = \epsilon$, then $\epsilon \in L(G)$ because of the production $S \to \epsilon$.
- Suppose $w$ is non-empty. To simplify notation, let $\Delta_i = \Delta(w_{\leq i})$ for every index $i$, and observe that $\Delta_0 = \Delta_{|w|} = 0$. There are several subcases to consider:
  - Suppose $\Delta_i = 0$ for some index $0 < i < |w|$. Then we can write $w = xy$, where $x$ and $y$ are non-empty strings with $\Delta(x) = \Delta(y) = 0$. The induction hypothesis implies that $x, y \in L(G)$, and thus the production rule $S \to SS$ implies that $w \in L(G)$.
  - Suppose $\Delta_i > 0$ for all $0 < i < |w|$. Then $w$ must begin with 00, since otherwise $\Delta_1 = -2$ or $\Delta_2 = -1$, and the last symbol in $w$ must be 1, since otherwise $\Delta_{|w| - 1} = -1$. Thus, we can write $w = 00x1$ for some binary string $x$. We easily observe that $\Delta(x) = 0$, so the induction hypothesis implies $x \in L(G)$, and thus the production rule $S \to 00S1$ implies $w \in L(G)$.
  - Suppose $\Delta_i < 0$ for all $0 < i < |w|$. A symmetric argument to the previous case implies $w = 1x00$ for some binary string $x$ with $\Delta(x) = 0$. The induction hypothesis implies $x \in L(G)$, and thus the production rule $S \to 1S00$ implies $w \in L(G)$.
  - Finally, suppose none of the previous cases applies: $\Delta_i < 0$ and $\Delta_j > 0$ for some indices $i$ and $j$, but $\Delta_i \neq 0$ for all $0 < i < |w|$.

  Let $i$ be the smallest index such that $\Delta_i < 0$. Because $\Delta_j$ either increases by 1 or decreases by 2 when we increment $j$, for all indices $0 < j < |w|$, we must have $\Delta_j > 0$ if $j < i$ and $\Delta_j < 0$ if $j \geq i$.

  In other words, there is a unique index $i$ such that $\Delta_{i-1} > 0$ and $\Delta_i < 0$. In particular, we have $\Delta_1 > 0$ and $\Delta_{|w| - 1} < 0$. Thus, we can write $w = 0x1y0$ for some binary strings $x$ and $y$, where $|0x1| = i$.

  We easily observe that $\Delta(x) = \Delta(y) = 0$, so the inductive hypothesis implies $x, y \in L(G)$, and thus the production rule $S \to 0S1S0$ implies $w \in L(G)$.

In all cases, we conclude that $G$ generates $w$. \qed

Together, Claim 1 and Claim 2 imply $L = L(G)$.

Rubric: 10 points:
- part (a) = 4 points. As usual, this is not the only correct grammar.
- part (b) = 6 points = 3 points for $\subseteq$ + 3 points for $\supseteq$, each using the standard induction template (scaled).