Let $L$ be an arbitrary regular language over the alphabet $\Sigma = \{0, 1\}$. Prove that the following languages are also regular. (You probably won’t get to all of these.)

1. $\text{FlipOdds}(L) := \{\text{flipOdds}(w) \mid w \in L\}$, where the function $\text{flipOdds}$ inverts every odd-indexed bit in $w$. For example:

$$\text{flipOdds}(0000111101010101) = 1010010111111111$$

**Solution:** Let $M = (Q, s, A, \delta)$ be a DFA that accepts $L$. We construct a new DFA $M' = (Q', s', A', \delta')$ that accepts $\text{FlipOdds}(L)$ as follows.

Intuitively, $M'$ receives some string $\text{flipOdds}(w)$ as input, restores every other bit to obtain $w$, and simulates $M$ on the restored string $w$.

Each state $(q, \text{flip})$ of $M'$ indicates that $M$ is in state $q$, and we need to flip the next input bit if $\text{flip} = \text{TRUE}$.

$$Q' = Q \times \{\text{TRUE, FALSE}\}$$
$$s' = (s, \text{TRUE})$$
$$A' =$$
$$\delta'((q, \text{flip}), a) =$$

2. $\text{UnflipOdd1s}(L) := \{w \in \Sigma^* \mid \text{flipOdd1s}(w) \in L\}$, where the function $\text{flipOdd1s}$ inverts every other 1 bit of its input string, starting with the first 1. For example:

$$\text{flipOdd1s}(000011110101010101) = 000010100010001$$

**Solution:** Let $M = (Q, s, A, \delta)$ be a DFA that accepts $L$. We construct a new DFA $M' = (Q', s', A', \delta')$ that accepts $\text{UnflipOdd1s}(L)$ as follows.

Intuitively, $M'$ receives some string $w$ as input, flips every other 1 bit, and simulates $M$ on the transformed string.

Each state $(q, \text{flip})$ of $M'$ indicates that $M$ is in state $q$, and we need to flip the next 1 bit of and only if $\text{flip} = \text{TRUE}$.

$$Q' = Q \times \{\text{TRUE, FALSE}\}$$
$$s' = (s, \text{TRUE})$$
$$A' =$$
$$\delta'((q, \text{flip}), a) =$$
3. **FlipOdd1s(L)** := \{flipOdd1s(w) | w ∈ L\}, where the function **flipOdd1** is defined as in the previous problem.

**Solution:** Let \( M = (Q, s, A, \delta) \) be a DFA that accepts \( L \). We construct a new NFA \( M' = (Q', s', A', \delta') \) that accepts **FlipOdd1s(L)** as follows.

Intuitively, \( M' \) receives some string **flipOdd1s(w)** as input, guesses which 0 bits to restore to 1s, and simulates \( M \) on the restored string \( w \). No string in **FlipOdd1s(L)** has two 1s in a row, so if \( M' \) ever sees 11, it rejects.

Each state \((q, flip)\) of \( M' \) indicates that \( M \) is in state \( q \), and we need to flip a 0 bit before the next 1 if \( flip = \text{TRUE} \).

\[
Q' = Q \times \{\text{True, False}\} \\
s' = (s, \text{True}) \\
A' = \\
\delta'((q, \text{flip}), a) =
\]

4. Prove that the language **insert1(L)** := \{x1y | x y ∈ L\} is regular.

Intuitively, **insert1(L)** is the set of all strings that can be obtained from strings in \( L \) by inserting exactly one 1. For example, if \( L = \{\varepsilon, 00K!\} \), then **insert1(L)** = \{1, 100K!, 010K!, 001K!, 00K1!, 00K!1\}.

**Solution:** Let \( M = (\Sigma, Q, s, A, \delta) \) be a DFA that accepts \( L \). We construct an NFA \( M' = (\Sigma, Q', s', A', \delta') \) that accepts **insert1(L)** as follows.

Intuitively, \( M' \) nondeterministically chooses a 1 in the input string to ignore, and simulates \( M \) running on the rest of the input string.

- The state \((q, \text{before})\) means (the simulation of) \( M \) is in state \( q \) and \( M' \) has not yet skipped over a 1.
- The state \((q, \text{after})\) means (the simulation of) \( M \) is in state \( q \) and \( M' \) has already skipped over a 1.

\[
Q' = Q \times \{\text{before, after}\} \\
s' = (s, \text{before}) \\
A' = \\
\delta'((q, \text{before}), a) = \\
\delta'((q, \text{after}), a) =
\]
5. Prove that the language \( \text{delete}_1(L) := \{xy \mid x1y \in L\} \) is regular.

Intuitively, \( \text{delete}_1(L) \) is the set of all strings that can be obtained from strings in \( L \) by deleting exactly one 1. For example, if \( L = \{101101,00,\epsilon\} \), then \( \text{delete}_1(L) = \{01101,10101,10110\} \).

6. Consider the following recursively defined function on strings:

\[
stutter(w) := \begin{cases} 
\epsilon & \text{if } w = \epsilon \\
\epsilon & \text{if } w = ax \text{ for some symbol } a \text{ and some string } x \\
ba \cdot stutter(x) & \text{if } w = \epsilon \\
ba \cdot stutter(x) & \text{if } w = ax \text{ for some symbol } a \text{ and some string } x \\
\end{cases}
\]

Intuitively, \( stutter(w) \) doubles every symbol in \( w \). For example:

- \( stutter(\text{PRESTO}) = \text{PPRREESSTTOO} \)
- \( stutter(\text{HOCUS}\diamond\text{POCUS}) = \text{HHOOCCUUSS}\diamond\diamond\text{PP00CCUUSS} \)

(a) Prove that the language \( stutter^{-1}(L) := \{w \mid stutter(w) \in L\} \) is regular.

(b) Prove that the language \( stutter(L) := \{stutter(w) \mid w \in L\} \) is regular.

7. Consider the following recursively defined function on strings:

\[
evens(w) := \begin{cases} 
\epsilon & \text{if } w = \epsilon \\
\epsilon & \text{if } w = a \text{ for some symbol } a \\
b \cdot evens(x) & \text{if } w = \epsilon \\
b \cdot evens(x) & \text{if } w = abx \text{ for some symbols } a \text{ and } b \text{ and some string } x \\
\end{cases}
\]

Intuitively, \( evens(w) \) skips over every other symbol in \( w \). For example:

- \( evens(\text{EXPELLIARMUS}) = \text{XELAMS} \)
- \( evens(\text{AVADA}\diamond\text{KEDAVRA}) = \text{VD}\diamond\text{EAR} \).

(a) Prove that the language \( evens^{-1}(L) := \{w \mid evens(w) \in L\} \) is regular.

(b) Prove that the language \( evens(L) := \{evens(w) \mid w \in L\} \) is regular.
You may find it helpful to imagine these transformations concretely on the following DFA for the language specified by the regular expression $00^*11^*$.