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 during the lab session.)

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:

$$\text{flipOdds}(0001111010100) = 10100101111110$$

**Solution:** Let $M = (Q, s, A, \delta)$ be an arbitrary 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 $\text{flip} = \text{TRUE}$.

$$Q' = Q \times \{$\text{TRUE}, \text{FALSE}$\}$$
$$s' = (s, \text{TRUE})$$
$$A' =$$
$$\delta'((q, \text{flip}), a) =$$
2. \text{UNFLIPODD}1s(L) := \{w \in \Sigma^* \mid \text{flipOdd}_1s(w) \in L\}, where the function \text{flipOdd}_1 inverts every other 1 bit of its input string, starting with the first 1. For example:

\[
\text{flipOdd}_1s(00001110010100) = 0000010100001000
\]

\textbf{Solution:} Let \(M = (Q, s, A, \delta)\) be an arbitrary DFA that accepts \(L\). We construct a new DFA \(M' = (Q', s', A', \delta')\) that accepts \(\text{UNFLIPODD}1s(L)\) as follows.

Intuitively, \(M'\) receives some string \(w\) as input, flips every other 1 bit, and then 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 if and only if \(\text{flip} = \text{True}\).

\[
Q' = Q \times \{\text{True}, \text{False}\} \\
s' = (s, \text{True}) \\
A' = \\
\delta'((q, \text{flip}), a) = 
\]
3. \( \text{FlipOdd}1s(L) := \{\text{flipOdd}1s(w) \mid w \in L\} \), where the function \( \text{flipOdd}1 \) is defined as in the previous problem.

**Solution:** Let \( M = (Q, s, A, \delta) \) be an arbitrary DFA that accepts \( L \). We construct a new NFA \( M' = (Q', s', A', \delta') \) that accepts \( \text{FlipOdd}1s(L) \) as follows.

Intuitively, \( M' \) receives some string \( \text{flipOdd}1s(w) \) as input, guesses which 0 bits to restore to 1s, and simulates \( M \) on the restored string \( w \). No string in \( \text{FlipOdd}1s(L) \) has two 1s in a row, so if \( M' \) ever sees 11, it must reject.

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

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


4. **Shuffle**($L$) := \{\text{shuffle}(w, x) \mid w, x \in L \text{ and } |w| = |x|\}, where the function shuffle is defined recursively as follows:

\[
\text{shuffle}(w, x) := \begin{cases} 
  x & \text{if } w = \varepsilon \\
  a \cdot \text{shuffle}(x, y) & \text{if } w = ay \text{ for some } a \in \Sigma \text{ and some } y \in \Sigma^*
\end{cases}
\]

For example, \text{shuffle}(0001101, 1111001) = 01010111100011.

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

Intuitively, $M'$ reads the string \text{shuffle}(w, x) as input, splits the string into the subsequences $w$ and $x$, and passes those strings to two independent copies of $M$. Let $M_1$ denote the copy that processes the first string $w$, and let $M_2$ denote the copy that processes the second string $x$.

Each state $(q_1, q_2, \text{next})$ indicates that machine $M_1$ is in state $q_1$, machine $M_2$ is in state $q_2$, and \text{next} indicates whether $M_1$ or $M_2$ receives the next input bit.

\[
Q' = Q \times Q \times \{1, 2\} \\
\delta'(\langle q_1, q_2, \text{next}, a \rangle) = \\
A' = (s, s, 1)
\]