1. For the sake of clarity, we will introduce the following notation. Given a finite multiset M (a set which allows multiple elements), we will denote $P(M)$ the set consisting of all unique permutations of M. To be more precise, we define a multiset M over an alphabet of allowable items Σ as $M = \Sigma \times \mathbb{N}$, such that if $(s_1, n) = (s_2, n)$, then $s_1 = s_2$, and if $(s, n) \in M$, then $(s', k) \in M$ for all $k < n$. Thus the second parameter can be thought of as the index of the first parameter in M. Let φ be the function that accepts a finite multiset and spits out its elements in order. In particular, $\varphi(\{(s_0, 0), (s_1, 1), \ldots, (s_n, n)\}) = s_0 s_1 \cdots s_n$. Then we can define $P(M) = \left\{\varphi(\{(s, \sigma(n)) : (s, n) \in M\}) : \sigma \in S_{|M|}\right\}$, where S_n denotes the standard symmetric group consisting of the $n!$ permutations of the numbers $1, 2, \ldots, n$. As a small example, $P(2, 1, 1) = \{211, 121, 112\}$.

   (a) We have the symbols $\Sigma = \{0, 1\}$ and the following non-terminals $\Gamma = \{S, A\}$, described with the production rules

   $$S \to 1A \,|\, 0S0$$
   $$A \to 1 \,|\, 0A0$$

   A represents the language $\left\{0^b 1 0^c : b = c\right\}$ and S represents the language $\left\{0^a 1 0^b 1 0^c : a + b = c\right\}$.

   (b) We have the symbols $\Sigma = \{0, 1\}$ and a single non-terminal $\Gamma = \{S\}$ described with the following production rule,

   $$S \to P(\{S, S, 0, 0, 1\}) \,|\, P(\{S, S, 0, 1\}) \,|\, P(\{S, 1\}) \,|\, \varepsilon,$$

   where all the elements of each set can be interpreted as a production that is or'ed with the rest. Namely, if $x \in X$, then $X = x \,|\, X \setminus \{x\}$. We claim that the language of S is precisely the language that was asked. Namely, every production rule in S produces something with no more than twice as many zeroes as ones. It is immediately clear that ε satisfies this condition, and via an induction argument, all productions of S also satisfy this condition. However, it is not immediately clear that all strings that satisfy this condition can be produced in this fashion. On the other hand, the problem statement does not ask us to prove that the languages we claim each production rule describes is correct, so we will not do that and instead just assert that this is indeed the desired language.

   (c) We have the symbols $\Sigma = \{0, 1\}$ and the following non-terminals $\Gamma = \{S, A, B, C\}$, described with the production rules

   $$S \to 0A0 \,|\, 1B1 \,|\, 1C0 \,|\, 0C1 \,|\, 0 \,|\, 1 \,|\, \varepsilon$$
   $$A \to P(\{C, 1, C, 1, C, 0, C\})$$
   $$B \to P(\{C, 0, C, 0, C, 1, C\})$$
   $$C \to C0C1C \,|\, C1C0C \,|\, \varepsilon$$

   The language produced by C consists of all strings with equal number of 1's and 0's. The language produced by B has one more 0 than 1. Similarly, the language produced by A has one more 1 than 0. We then invoke Piazza post 341, so we do not have to comprehensively prove the following statements. The productions $1C0$ and $0C1$ have the same number of 0's and 1's, and they start and end with different characters, so they have the same number of 00's and 11's. Similarly, for a string surrounded by 0's on both sides, $0A0$ has one more 0 than 1, and we get an equal number of 00's and 11's. $1B1$ is analogous to $0A0$. The rest are just corner cases, but S is indeed the language where there are equal amounts of occurrences of 00 and 11.

2. We make the following observation. For a string $w \in \Sigma^*$ of length n, where $\Sigma = 0, 1$, $w+1$ either does not induce a carry operation, or causes a carry operation, flipping the last character of w (which is 1), to 0, and recursing into the length $n-1$ prefix of w. $\varepsilon + 1$ was vacuously defined to be ε.

   Given a DFA $(\Sigma, Q, \delta, q_0, A)$ for an arbitrary regular language L, we will construct an NFA $(\Sigma, Q', \delta', q_0', A')$ that accepts $\text{inc}(L)$ by non-deterministically guessing which bit the carry operation "stopped" on. We define

$$\begin{aligned} Q' &= Q \times \{\omega, \Omega\} \\ q_0' &= (q_0, \omega) \\ A' &= A \times \{\Omega\}, \end{aligned}$$

   and give the transition function δ' as follows,

$$\begin{aligned} \delta'((q,\omega),0) &= \{(\delta(q,0),\omega)\} \\ \delta'((q,\omega),1) &= \{(\delta(q,1),\omega),(\delta(q,0),\Omega)\} \\ \delta'((q,\Omega),0) &= \{(\delta(q,1),\Omega)\} \\ \delta'((q,\Omega),1) &= \varnothing \\ \delta'(q,\varepsilon) &= \begin{cases} (q_0,\Omega) & q = q_0' \\ \varnothing & q \neq q_0' \end{cases}. \end{aligned}$$

   Our new states are a product of the old states and two new "meta-states", that describe if we are attemping to reverse enginner the carry operation or not. ω represents no (carry ends somewhere after this character) and Ω represents yes (carry ends somewhere before this character). Notice that in the carry operation in this binary increment, the largest block of 1s at the end of the string will become a block of 0's and the 0 immediately preceeding the block (if it exists) becomes a 1.

   Therefore, whenever we read a 0 and we think that the series of carries ended sometime afterwards, it is impossible for this 0 to have been produced from a carry operation. Likewise, whenever we read a 1 and we think that the series of carries ended sometime later, this 1 could be the result of a carry, so we branch off into two universes. Do note that when we operate in the Ω universe, we have to flip the bits received from $\text{inc}(L')$ because of the series of bit flips that occur in the increment process. As long as we keep reading 0's in the Ω world, we can only believe that this 0 is a result of a carry, but once we read a 1, we know for sure that this is not the case (again, because of the series of bit flips), so we immediately fail. Additionally, we introduce an ε-transition at the start because the final bit flip $(0 \to 1)$ may have been truncated.

   Finally, this NFA only accepts when our guessed string for L accepts and we have accounted for the bit flips associated with increments.

3. Let $(\Sigma, Q_1, \delta_1, q_1, A_1)$ be a DFA for L and let $(\Sigma, Q_2, \delta_2, q_2, A_2)$ be a DFA for L', where $\Sigma = \{0, 1\}$. We give a new NFA for shuffles(L, L') by $(\Sigma, \Delta, Q_1 \times Q_2, (q_1, q_2), A_1 \times A_2)$ as follows,

$$\Delta((q, q'), a) = \{(\delta_1(q), q'), (q, \delta_2(q')\}\,.$$

This NFA transitions by either pretending the read character is from L or L', but never both. State (q, q') means that we have read until state q for L and read until state q' for L'. The NFA only accepts when we have accepted a string in L and a string in L'.