1. **(20 points)** *Recall the inductive definition for the set, S, of all strings in $\{0,1\}^*$ with an equal number of 1's and 0's (see the September 8 lecture notes): w is in S iff*

   - *$w = \epsilon$; or*
   - *There is a string x in S such that $w = 0x1$ or $w = 1x0$; or*
   - *There are strings x and y in S such that $w = xy$.*

   (a) **(10 points)** *Give an inductive definition for a set, T, that contains all strings that have more 1's than 0's.*

   Solution: String w is in T iff

   - There are strings x and y in S such that $w = x1y$, where S is the set of all string with an equal number of ones and zeros as defined in the problem statement.
   - There are strings x and y in T such that $w = xy$.

   (b) **(10 points)** *Give a proof that your solution to part (a) is correct.*

   Solution:

   Let $numOne(w)$ denote the number of 1's in string w and $numZero(w)$ denote the number of 0's. We prove that T is the set of all strings that have more 1's than 0's by showing the set inclusion in each directions.

   Every string in T has more 1's than 0s:

   Proof by induction on the derivation of the string.

   Let $w \in T$ be a string. There are two cases to consider:

   $\exists x, y \in S.\ w = x1y$:

1.	$numZero(x) = numOne(x)$,	S is the set of strings with an equal number of 0's and 1's.
2.	$numZero(y) = numOne(y)$,	same as for step 1
3.	$numZero(w) = numZero(x) + numZero(y)$,	w = x1y
4.	$numOne(w) = numOne(x) + 1 + numOne(y)$,	w = x1y
5.	$numOne(w) = numZero(w) + 1$,	subsitition, 1-4
5.	$numOne(w) > numZero(s)$,	step 4

   It is also acceptable to write the equivalent proof in English:

   It was shown (in the Sept. 11 notes) that for any string s in S, the number of 0's and 1's in s are equal. Thus, x has an equal number of 0's and 1's as does y. The number of 0's in w is the total number of 0's in x and y. The number of 1's in w is one greater than the total number in x and y. Thus, The number of 1's in w is one greater than the number of 0's in w which means that w has more 1's than 0's.

   $\exists x, y \in T.\ w = xy$:

1.	$numOne(x) > numZero(x)$,	induction hypothesis: $x \in T$
2.	$numOne(y) > numZero(y)$,	induction hypothesis: $x \in T$
3.	$numOne(w) = numOne(x) + numOne(y)$,	w = xy
4.	$numZero(w) = numZero(x) + numZero(y)$,	w = xy
5.	$numOne(w) > numZero(w)$,	substitution and addition, 1-4

   Again, a proof written in English prose acceptable. The use of the induction hypothesis should be clearly indicated.

   We've shown for both cases that $numOne(w) > numZero(w)$. Therefore, every string in T has more ones than zeros.

Every string that has more 1's than 0's is in T:

Let w be a string that has more 1's than 0's. Let x be the shortest prefix of w that has more 1's than zeros – note that w has this property so such a prefix must exist. Furthermore, x must have exactly one more 1 than 0, and x must end with a 1. Thus, we can choose u such that $x = u1$, and $u \in S$. Now, choose y such that $w = xy$. Note that $numOne(y) \geq numZero(y)$. We consider two cases:

$numOne(y) = numZero(y)$: This means that $y \in S$. We now have $w = u1y$ with $u, y \in S$. Thus, the first case in the definition of T applies, and $w \in T$.

$numOne(y) > numZero(y)$: This means that $y \in T$. Furthermore, $x = u1\epsilon$, and u and ϵ are both in S. Therefore, $x \in T$ by the first case in the definition of T. Having shown that x and y are both in T, we conclude that $xy \in T$ using hte second case in the definition of T. This shows that $w \in T$.

We've shown for both cases that $w \in T$. Therefore, every string in that has more 1's than 0's is in T.

We've shown that every string in T has more ones than zeros and that every string that has more ones than zeros is in T. Thus, T is the set of all strings that have more ones than zeros.

I've been careful to put "wrap-up" statements at the end of each part of the proof. Acceptable solutions can omit those when they are clear and be somewhat less detailed than mine.

2. **(20 points)** *Let $\Sigma = \{0, 1, 2\}$. Let $\subseteq \Sigma^* H$ be the language that contains a string w iff*

- *$w = \epsilon$; or*
- *There are strings x and y in H such that $w \in \{0x1y2, 0x2y1, 1x0y2, 1x2y0, 2x0y1, 2x1y0\}$.*

(a) **(10 points)** *Prove that for each string, w in H, the number of 0's, 1's and 2's in w are all equal to each other.*

Solution: Let $numZero(w)$, $numOne(w)$ and $numTwo(w)$ denote respectively the number of 0's, 1's and 2's in w. Let $w \in H$ be a string. To show that $numZero(w) = numOne(w) = numTwo(w)$, there are two cases to consider according to the definition of H:

$w = \epsilon$: $numZero(w) = numOne(w) = numTwo(w) = 0$.

$w \in \{0x1y2, 0x2y1, 1x0y2, 1x2y0, 2x0y1, 2x1y0\}$: We consider the case where $w = 0x1y2$, the other cases are equivalent. We have:

$$\begin{array}{rrcll}
1. & numZero(w) & = & 1 + numZero(x) + numZero(y), & w = 0x1y2 \\
2. & numOne(w) & = & 1 + numOne(x) + numOne(y), & w = 0x1y2 \\
 & & = & 1 + numZero(x) + numZero(y), & \text{induction hypothesis:} \\
 & & & & \quad numOne(x) = numZero(x) \\
 & & & & \quad \text{and } numOne(y) = numZero(y) \\
 & & = & numZero(w), & \text{substitution, step 1} \\
3. & numTwo(w) & = & 1 + numTwo(x) + numTwo(y), & w = 0x1y2 \\
 & & = & 1 + numZero(x) + numZero(y), & \text{induction hypothesis:} \\
 & & & & \quad numTwo(x) = numZero(x) \\
 & & & & \quad \text{and } numTwo(y) = numZero(y) \\
 & & = & numZero(w), & \text{substitution, step 1} \\
4. & \multicolumn{3}{l}{numZero(w) = numOne(w) = numTwo(w),} & \text{steps 2 \& 3}
\end{array}$$

This completes the proof.

(b) **(10 points)** *Does H contain all strings that have an equal number of 0's, 1's and 2's? Give a short proof for your answer.*

Solution: H **does not** contain all strings that have an equal number of 0's, 1's and 2. For example, H does not include the string **012210**.

Proof: The first rule for H produces the empty string. All strings produced by the second rule have first and last symbols that differ. Neither rule can produce the string **012210**.

An acceptable proof would be:

There are no strings in H for which the first and last symbol are the same.

or

If $w \in H$ and $w \neq \epsilon$, then the first and last symbols of w are different.

3. **(30 points)** *Let $\Sigma = \{\mathsf{a}, \mathsf{b}\}$. Figure 1 depicts three finite state machines that read inputs from this alphabet. Let L_a, L_b, and L_c be the languages accepted by DFA (a), DFA (b), and DFA (c) respectively.*

   (a) **(9 points)** *For each of L_a, L_b, and L_c, list three strings in Σ^* that are in the language and three strings in Σ^* that are not in the language.*

   Solution:

   L_a: The strings a, aa and aaa are in L_a.
   The strings b, ab and bb are not in L_a.

   L_b: The strings aa, baa and baabaa are in L_b.
   The strings b, ab and bb are not in L_b.

   L_c: The strings aaa, aaaa and baaa are in L_c.
   The strings b, ab and bb are not in L_c.

   (b) **(12 points)** *Write a short description of each of the languages, L_a, L_b and L_c.*

   Solution:

   L_a: $w \in L_a$ iff w ends with an a.

   L_b: $w \in L_b$ iff w ends with two a's followed by zero or more repetitions of ba.
   It is not correct to say that L_b is the set of all strings that end with two a's. For example, the string aaba is in L_b, but it does not end with two a's.

   L_c: For this one, it's convenient to define two other languages first. Let L_{ba} be the language of all strings consisting of zero or more repetitions of ba; for example ϵ, ba, and babababa are in L_{ba}. Let L_{bba-a} be the language of all strings of the form bba y a where $y \in L_{ba}$.
   Using these definitions, a string w is in L_c iff w ends with a suffix of the form aa y a z where $y \in L_{ba}$ and is the concatenation of zero or more strings from L_{ba} or L_{bba-a}.
   Explanation: Let z be the suffix of as string w as described above. The aa at the beginning of z moves the machine to state 2. The string y moves the machine back and forth between states 1 and 2 any number of times (perhaps zero), ending in state 2. The next a moves the machine to state 3. Once the machine has reached state 3, any string from L_{ba} moves the machine back and forth between states 2 and 3 any number of times (perhaps zero). Likewise, as string from L_{bba-a} brings the machine back to state 1 (with the bb) then forward to state 2 (with the a) and eventually back to state 3 (with the final a).
   Note that we don't have to worry about strings that take the machine all the way back to state 0 – we can just start again with a later suffix.

   By the time that this is posted, we will have seen regular expressions. I wrote my description without using regular expressions. Here's the same descriptions written as regular expressions:

$$
\begin{aligned}
L_a &= \Sigma^* \,\mathsf{a} \\
L_b &= \Sigma^* \,\mathsf{aa}\,(\mathsf{ba})^* \\
L_c &= \Sigma^* \,\mathsf{aa}\,(\mathsf{ba})^*\,\mathsf{a}\,(\mathsf{ba} \cup (\mathsf{bba}\,(\mathsf{ba})^*\,\mathsf{a}))^*
\end{aligned}
$$

(c) **(9 points)**
Is $L_a = L_b$, $L_a \subset L_b$, $L_a \supset L_b$, or none of these?
Is $L_b = L_c$, $L_b \subset L_c$, $L_b \supset L_c$, or none of these?
Is $L_a = L_c$, $L_a \subset L_c$, $L_a \supset L_c$, or none of these?
Give a short justification of your answers.

Solution: $L_a \supset L_b \supset L_c$.

Any string in L_b ends with an a and is therefore in L_a. Conversely, the string a is in L_a but not in L_b; thus the superset relation, $L_a \supset L_b$ is strict.

Let δ_b and δ_c be the state transition functions for DFA(a) and DFA(b) respectively. I'll now show by induction that for all strings, w,

$$\delta_c(0, w) - 1 \quad \leq \quad \delta_b(0, w) \quad \leq \quad \delta_c(0, w).$$

My proof is (of course) by induction – in this case on w.

$w = \epsilon$: $\delta_b(0, \epsilon) = 0 = \delta_c(0, \epsilon)$.

$w = x \cdot c$:

If $c =$ a and $\delta_b(0, x) < 2$, Both machines move one to the right and the induction hypothesis is maintained.

If $c =$ a and $\delta_b(0, x) = 2$, DFA(b) stays in state 2. DFA(c) must have been in state 2 or 3 after reading x, and moves to state 3 after reading the a. The induction hypothesis is maintained.

If $c =$ b and $\delta_b(0, x) > 0$, Both machines move one to the left and the induction hypothesis is maintained.

If $c =$ b and $\delta_b(0, x) = 0$, DFA(b) stays in state 0. DFA(c) must have been in state 0 or 1 after reading x, and moves to state 0 after reading the a. The induction hypothesis is maintained.

Now, consider $w \in L_c$. This means that $\delta_c(0, w) = 3$. By the result that we just proved by induction,

$$2 \quad \leq \quad \delta_b(0, w) \quad \leq \quad 3,$$

but $\delta_b(0, w)$ must be less than 3. Therefore, $\delta_b(0, w) = 2$ which means that DFA(b) accepts w. Therefore $w \in L_b$.

The string aa is in L_b but not in L_a. This shows that the superset relationship, $L_b \supset L_c$ is strict.

I'll also accept a solution that doesn't set up a formal induction proof. For example:

Let w be a string in L_c. As noted earlier, this means that w ends with two a's followed by zero or more repetitions of ba followed by an a followed by zero or more repetitions of ba. Note that we can find strings x and y such that

- $w = xy$;
- x ends with two a's followed by zero or more repetitions of ba followed by an a,
- y consists of zero or more repetitions of ba.

Any such x must end with two consecutive a's (just consider the cases for zero repetitions of ba and more than zero repetitions). Therefore, xy is a string that ends with two a's followed by zero or more repetitions of ba. Thus, $xy \in L_b$.

Of course, an example to show that the subset relationship is strict is still required.

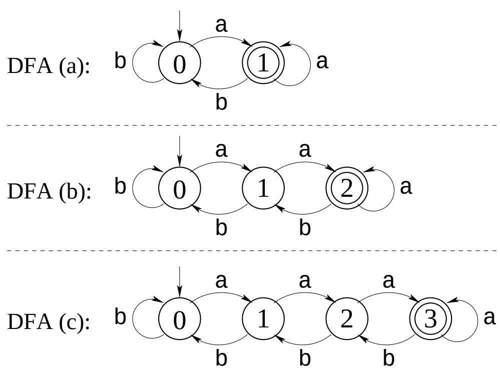

Figure 1: Finite state machines for question 3