♢ **Best before:** November 7.

1. For each of these implications, state whether it holds for arbitrary regular expressions R, S and T. Justify your answer.

   (a) If $R + \emptyset \equiv S + \emptyset$, then $R \equiv S$.

   (b) If $R \equiv R^*$, then $R \equiv R + \epsilon$.

   (c) If $\mathcal{L}(R) \subseteq \mathcal{L}(S) \subseteq \mathcal{L}(T)$, then $(R^* + T)^* \equiv S^* + T^*$.

   (d) If $RTS \not\equiv STR$ and $\mathcal{L}(T)$ is infinite, then $RS \not\equiv SR$.

   (e) If $R^* \equiv S^*$, then $R \equiv S$.

   (f) An *implication* is a statement of the form "if *antecedent*, then *consequent*".
   Make up an implication about equivalence of regular expressions.
   Put another way, add another part to this question.
   As well, provide an answer to one of your classmate's additions to this question.

2. **Crunchy alert:** This question is somewhat open-ended. Just working on understanding what is being asked will improve your mathematical maturity.

   Let A_{17} be the set of all languages over alphabet $\{0, 1\}$ that contain at least one string of length 17, but none longer. Then every language in A_{17} is regular (why?). For any DFSA M, let size(M) be the number of states in M. For each language $L \in A_{17}$, let D_L be the set of all DFSAs that accept L, and let $m(L) = \min_{M \in D_L} \text{size}(M)$. I.e., to get $m(L)$, we would need to consider every DFSA in D_L, and find one with the fewest states.

   (best case) Give a language $L \in A_{17}$ such that $m(L)$ is minimized over all $L \in A_{17}$.
   I.e., Find a language $L \in A_{17}$ such that $m(L) \leq m(L')$, for every $L' \in A_{17}$.

   (worst case) Give a language $L \in A_{17}$ such that $m(L)$ is maximized over all $L \in A_{17}$.
   I.e., Find a language $L \in A_{17}$ such that $m(L) \geq m(L')$, for every $L' \in A_{17}$.

   Advice: Try the question with 17 replaced by $1, 2, 3, \cdots$ and see if you can spot a pattern, then generalized your results.

3. (a) Do exercise 6 on page 237 of the course notes (about $\emptyset$-free regular expressions).

   (b) How would you define the set of ϵ-free regular expressions and what similar property can you prove about them?

4. Let Σ be an alphabet. Consider the following operations defined on strings from Σ^*.

   - Informally, for two strings x, y of equal length, we define $\lambda(x, y)$ to be the string obtained by interleaving the symbols of x and y.[1] Formally, if $x = a_1 \cdots a_n$ and $y = b_1 \cdots b_n$, where $n \in \mathbb{N}$ and each $a_i, b_j \in \Sigma$, then $\lambda(x, y) = a_1 b_1 \cdots a_n b_n$. For example, if $x = inter$ and $y = leave$, then $\lambda(x, y) = ilnetaevre$ and $\lambda(y, x) = lienatveer$.
   - Informally, for a nonempty string x, we define $\rho(x)$ to be the string obtained by moving the last symbol in x to the first position.[2] Formally, if $x = x'c$, where $x' \in \Sigma^*$ and $c \in \Sigma$, then $\rho(x) = cx'$; and if $x = \epsilon$, then $\rho(x) = \epsilon$. For example, if $x = rotate$, then $\rho(x) = erotat$.

   Suppose L and L' are regular languages. Prove that the following languages are also regular.

[1] In case you didn't know, λ is the lower case Greek letter *lambda*.
[2] In case you didn't know, ρ is the lower case Greek letter *rho* (pronounced with a silent 'h').

(a) Interleave$(L, L') = \{x \in \Sigma^* : x = \lambda(y, z)$ for some $y \in L$ and $z \in L'$, where $|y| = |z|\}$.

(b) Rotate$(L) = \{x \in \Sigma^* : x = \rho(y)$ for some $y \in L\}$.

(c) *For those who like a challenge!* ReverseRotate$(L) = \{y \in \Sigma^* : \rho(y) = x$ for some $x \in L\}$.

5. For each of the following languages, construct a DFSA that accepts it and a regular expression that denotes it. Prove that your automata and regular expressions are correct. Use as few states as possible in your DFSA.

   (a) $L_1 = \{x : 00$ is a substring of x, but 11 is not$\}$.

   (b) $L_2 = \{x : x$ represents, in base 2, a non-negative integer i such that $i \bmod 3 = 0\}$.
   (For convenience, we let the empty string represent zero.)
   Beware: The regular expression for this language is not easy to find.

   (c) $L_3 = \{x :$ the number of odd parity substrings that can be found in x is odd$\}$.
   Recall that a binary string has odd parity iff it has an odd number of 1s.

6. (a) Prove that the language $\{0\,1^k0 : k$ is a power of 2$\}$ over alphabet $\{0, 1\}$ is **not** regular.
   "k is a power of 2" means for some $j \in \mathbb{N}$, $k = 2^j$.

   (b) Make up a language that is **not** regular. Put another way, add another part to this question.
   As well, use the Pumping Lemma to prove one of your classmate's languages to be **not** regular.

   (c) Make up a language operation that does **not** preserve regular languages. Prove your claim.

7. Do exercise 13 on page 239 of the course notes (about converses of closure properties).