After doing this tutorial you should be able to:

1. design a regular expression given an English specification
2. write an English description of the language of a given regular expression
3. reason and prove results about regular expressions
4. write and justify recursive definitions/processes

Problem 1. Let $\Sigma = \{a, b\}$. Write regular expressions for the following:

1. The set of strings ending in a.
2. The set of strings that have aba as a substring.
3. The set of strings of even length.
4. The set of strings with an even number of a's.
5. The set of strings that do not contain ab as a substring.
6. The set of strings not equal to ab. (Exam 2022)
7. The set of strings not containing bab as a substring (hard).

Problem 2. Let $\Sigma = \{0, 1\}$.

We use the following shorthand: Σ to mean $(0 \mid 1)$

Write English descriptions for the language of each of the following regular expressions:

1. $\Sigma^* 1 \Sigma \Sigma$.
2. $(\Sigma\Sigma\Sigma)^*$
3. $(\Sigma^* 0 \Sigma^* 1 \Sigma^*) \mid (\Sigma^* 1 \Sigma^* 0 \Sigma^*)$.
4. $(1^* 0 1^* 0 1^* 0)^* 1^*$
5. $(0^* 10)^* 0^* 110^* (010^*)^*$ (hard; hint: it is testing if a certain string occurs a certain number of times).

Problem 3. (Assignment Question in 2023, extra)

For each of the following three languages over $\Sigma = \{a, b\}$, provide a regular expression for the language.

1. The set of strings that contain $abbab$, in that order, but not necessarily consecutively.

   Said differently, the set of strings such that one can delete some amount of letters, possibly zero, to obtain $abbab$.

   For example, $aaababaababba$ is in the language, while $bbbababbbaa$ is not.

2. The set of strings of length 4 that contain exactly two bs.

   For example, $abba$ is in the language, while $abbb$ and abb are not.

3. The set of strings that contain exactly three as and an even number of bs.

   For example, $abababb$ is in the language, while $ababab$ is not.

Problem 4. Give a recursive procedure that decides, given R, if $L(R)$ contains the empty string. Make sure you understand why each case is correct.

Recall the recursive definition of regular expressions: a regular expression is one of $\emptyset, \epsilon, a \in \Sigma, (R_1|R_2), (R_1R_2), R_1^*$. Hence your procedure should handle these cases separately.

Problem 5. Regular expressions can be used to search text for patterns. How? By solving the membership problem for regular expressions. In this problem you will design an algorithm for this problem (over a fixed alphabet Σ). You will write a recursive procedure Match(R, x) that takes as input a regular expression R over Σ and a string $x \in \Sigma^*$, and returns 1 if $x \in L(R)$, and 0 otherwise.

For instance, say $\Sigma = \{0, 1\}$ and $R = (0^*1^*)\,|(1^*0^*)$. If $x = 00111$ then the algorithm returns 1, and if $x = 001100$ then the algorithm returns 0.

1. Provide (high-level) pseudocode for your algorithm and very briefly describe the main idea(s) in plain English.
2. Make sure you understand why your algorithm is correct.

Hint: do not worry about finding an algorithm with good complexity (i.e., fast or memory efficient). Just get a correct algorithm. Later in the course we will see how to solve the membership problem in polynomial time. For a history of regular-expression matching see https://www.youtube.com/watch?v=NTfOnGZUZDk

Problem 6. (Assignment Question in 2022)

Let $\Sigma = \{a, b\}$. For each of the following regular expressions, provide a clear and concise English description for the language of that regular expression. No justification is necessary.

1. $(a|b)^*((aaa)|(bbb))(a|b)$
2. $b^*(abb^*)^*(a|\epsilon)$

Problem 7. (Assignment Question in 2022)

Let $\Sigma = \{a, b\}$. For each of the following languages, provide a regular expression for that language. No justification is necessary.

1. The set of strings that contain at most two b.
2. The set of strings that have even length or contain no a, but not both.
3. The set of strings that contain exactly two out of the four length 2 strings (aa, ab, ba, bb) as substrings.

Problem 8. (Assignment Question in 2022) (hard) Give a recursive procedure that decides, given R, if $L(R)$ is infinite. Explain why each case is correct.

Hint: It is likely necessary to design a procedure that obtains and uses some additional information about the subexpressions beyond whether their languages are infinite. Be very careful with regards to edge cases, such as the empty set.