- The homework is due on April 16, 23:59pm. Please submit your solutions to Gradescope.
- Starting from Homework 1, all homework sets allow *group submissions* up to 2 people. Please write down the names of the members *very clearly* on the first page of your solutions.
- Answer the questions in a way that is clear, correct, convincing, and concise. The level of details to aim for is that your peers in this class should be convinced by your solutions.
- You can use any statements proved during the working sessions/lectures without proofs in your solutions.
- You might notice the difficulty of the homework problems are much higher than the worksheets. *This is by design.* These problems are meant to stretch your ability and solidify your understanding of the core concepts.
- You are expected to spend a reasonable amount of time (measured in hours) working on these problems. Remember you are allowed to utilize any resources. Make sure to cite all the people/webpages/source of infomation that helped.
- Some problems are marked with a *star*; these are more challenging (and fun) extra credit problems. They are optional and do not count toward raw grades.

---

1. ***Register machine.*** A *k-register machine* is a DFA augmented with extra k-bits of memory called *register*; the transition of the DFA is determined by *both* the input bit and the current values in the registers. Formally, a ***k-register machine M*** consists of the following components:

   - a set of *states* Q,
   - a *starting state* $s \in Q$,
   - a set of *accepting states* $A \subseteq Q$,
   - *alphabet set* Σ,
   - a set of ***registers*** $\boldsymbol{R} = \underbrace{\Sigma \times \cdots \times \Sigma}_{k \text{ times}}$, taking values over Σ,
   - a ***transition function*** $\boldsymbol{\delta : Q \times \Sigma \times R \to Q \times R}$; that is, given the *current state* q, an *input character* a, and *current values of the k registers* $r := (r_1, \ldots, r_k)$, the transition function outputs the next state $\delta(q, a, r)$, and write the new values $r' := (r'_1, \ldots, r'_k)$ into the registers.

   We can define the extended transition function δ^* similarly to the regular DFA. Just like regular DFAs, a k-register machine M ***accepts*** string w if $\delta^*(s, w) \in A$. The language of a k-register machine M is defined to be

   $$L(M) := \left\{ w \in \Sigma^* : M \text{ accepts } w \right\}.$$

   Prove that given any language L of some k-register DFA M for constant k, one can construct a regular DFA D that accepts the same set of strings in L. In notation,

   $$\forall \ k\text{-register DFA } M, \quad \exists \text{ DFA } D \quad \text{such that} \quad L(D) = L(M).$$

2. ***Erasing digit sequence.*** Let the input be a string of digits from `0` to `9` (in other words, the alphabet set Σ is $\{0, \ldots, 9\}$). The ERASE function is defined as follows:

```
ERASE(w):
  input: digit string w
  digit string r ← ε
  while w is not empty:
      d ← first digit of w
      remove the first digit of w
      r ← r · d   ⟨⟨append d after r⟩⟩
      if there are at least d digits left in w:
          remove d digits from w
      else:
          return fail
  return r
```

A digit string w is ***erasable*** if ERASE(w) successfully returns another digit string. For example, string $w =$ `314159265358979323846264338327950288419` is erasable because

ERASE(w) = 3~~141~~5~~92653~~5~~89793~~2~~38~~4~~6264~~3~~383~~2~~79~~5~~02884~~1~~9~~ = `355243251`.

Construct DFAs that recognize the following languages.

(a) $\{w \in \Sigma^* : w \text{ is erasable}\}$

(b) $\{w \in \Sigma^* : \text{both } w \text{ and } \textsc{Erase}(w) \text{ are erasable}\}$

[It is not *sufficient to just draw the diagram; you* must *explain your construction, especially what each state represents, in English. (This is equivalent to commenting your code with the meaning of each variable.) Remember your job is to* ***convince*** *the reader that your construction is correct. Alternatively, you may describe the DFAs using the formal tuple $(Q, s, A, \Sigma, \delta)$. But you still need to explain your construction. Answers without English explanations will receive no credit, even when the answers are correct.]*

*3. ***Wait, this is regular?*** Design a regular expresion for the language containing binary representations of positive integers divisible by 3.