- Find students around you to form a ***small group***.
- Work on the exercise problems in sequence. You are allowed to use ***all resources*** (including textbook and search engines) to help you solve the problems.
- After solving each problem, ***discuss*** your solution with other group members and exchange your ideas. You should try to poke holes in each other's argument and defend your own; this is a good way to learn about how to communicate.
- In daily life we are used to *not argue* with people and overlook the gaps in the statements. Well, not for this course. Being critical and patient when listening to others, ask questions to see if you understand them correctly, and provide well-thoughtout counter-arguments when you disagree. Be civilized and argue about ideas but don't make it personal.
- As you are trying to solve the problems you might have questions. Ask the questions within the group and see if others know the answer. If no ones knows, or the whole group is stuck and don't know how to proceed, Raise your hand and pull one of the ***course staff*** to help.
- After reaching concensus, ***write down*** the solutions *in your own words*. You are not allowed to copy sentences from others or TAs; everything you deliver must be from *you*. Use the writing tips provided on Canvas to help with your presentation.
- ***Submit*** your writeup with the proofread signature through Gradescope. You have *24 hours* to do so. The TA will grade them and provide feedback.

Our topic for this working session is the *DFA construction*.

Remember that DFA is just a directed graph with labels. The language of DFA M is defined to be

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

We are going to learn about two signature operations on DFAs: how to combine two languages by taking a *union*, and how to take a *complement*.

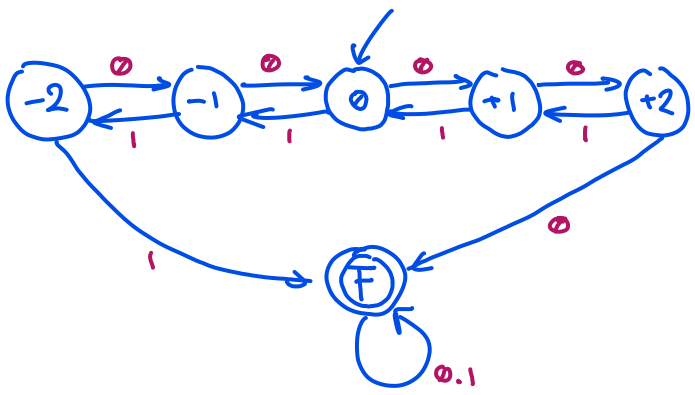
Example. Construct deterministic finite automata (DFAs) over alphabet $\{0, 1\}$:

- Set of all strings where some prefix has number of 0s and 1s differ by at least three.

Solution: First construct a DFA for the language

- Set of all strings where the number of 0s and 1s differ by at most two in every prefix,

then take the completement.



Construct deterministic finite automata (DFAs) for each of the following languages. No formal proofs required, but explanation of your construction in a few English sentences would be helpful. Throughout the exercise we assume the alphabet to be $\Sigma = \{0, 1\}$.

1. Set of all strings where number of `0`s is divisible by 5
2. Set of all strings with exactly a single `1`, and the number of `0`s is divisible by 5
3. Set of all strings containing `000` or `111` as *subsequences*
4. Set of all strings *not* containing `000` *nor* `111` as subsequences

---

To think about later: (No submissions needed)

5. Set of all strings that are *binary representations* of nonnegative integers divisible by 5
6. Set of all strings whose *reverses* are nonnegative integers divisible by 5 in binary

Conceptual question: Is there a general rule to express the *complement* of a language? That is, how do we write a regular expression for the set of strings *not* in a given regular language L?