A Turing Machine (TM) adds to a FA the capabilities to:

- change a char in the string on its tape,
- move a scanning head left (L) or right (R) or stay (S).

Text puts these in different instructions ("4-tuple notation") but we and Turing Kit combine them ("5-tuple notation").

**Def:** A TM is a 7-tuple \( M = (Q, \Sigma, \Gamma, \delta, s, \text{acc}, F) \) where:

- \( Q \) is a finite set of states,
- \( \Sigma \) is a finite input alphabet (or I/O alphabet),
- \( \Gamma \), which includes \( \Sigma \) and the blank \( '\_\' \), is the work alphabet,
- \( s \in Q \) is the start state,
- \( F \subseteq Q \) is the set of desired final states, aka accepting states.

Turing Kit allows \( F \) to be general, but we will design to the text's "normal form" when \( F = \{ \text{acc} \} \) and \( \text{grei} \) is the only other halting state.

And

\[
S \subseteq Q \times \Gamma \times \Gamma \times \{L,R,S,F\} \times Q
\]

Typical instruction:

\[
(p, \sigma \downarrow \sigma, d, D, q) \quad \text{where} \quad p \in Q \setminus \{ \text{acc}, \text{grei} \}, \quad q \in Q, \quad d \in \Gamma
\]

In a diagram:

\[
p \quad \downarrow \sigma \quad \eta \quad d \quad D \quad q
\]

The machine is deterministic (a DTM) if the instructions define \( S \) as a function

\[
S : \Phi \times \Gamma \rightarrow \Gamma \times \{L,R,S,F\} \times Q
\]

The rest of the lecture was a long demo of Turing machines with 1 and 2 tapes, incl. PDAs.

Given a DTM \( M \) and an input \( x \in \Sigma^* \), \( M(x) \) denotes a computation path that might:

- end in \( \text{acc} : M \) accepts \( x \) & \( x \in L(M) \),
- \( \text{grei} : M \) halts and rejects \( x \), OR \( M(x) \uparrow \) ("diverges"), also \( x \notin L(M) \).