Finite Automata

Part One

Computability Theory

What problems can we solve with a computer?

What problems can we solve with a computer?

What kind of computer?

Two Challenges

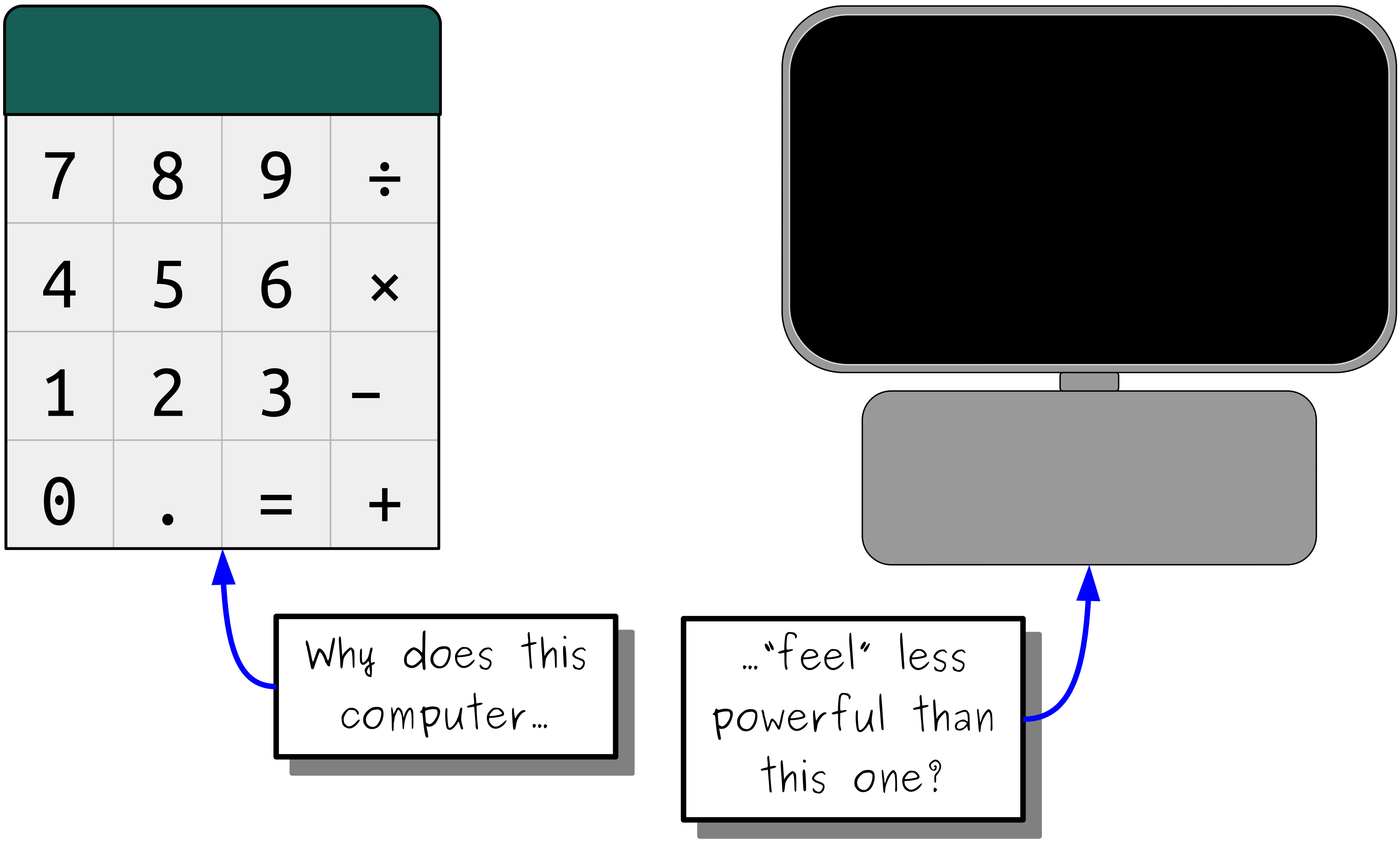
- Computers are dramatically better now than they've ever been, and that trend continues.
- Writing proofs on formal definitions is hard, and computers are *way* more complicated than sets, graphs, or functions.
- ***Key Question:*** How can we prove what computers can and can't do…
  - … so that our results are still true in 20 years?
  - … without multi-hundred page proofs?

Enter Automata

- An ***automaton*** (plural: ***automata***) is a mathematical model of a computing device.
- It's an ***abstraction*** of a real computer, the way that graphs are abstractions of social networks, transportation grids, etc.
- The automata we'll explore are
  - powerful enough to capture huge classes of computing devices, yet
  - simple enough that we can reason about them in a small space.
- They're also fascinating and useful in their own rights. More on that later.

What do these automata look like?

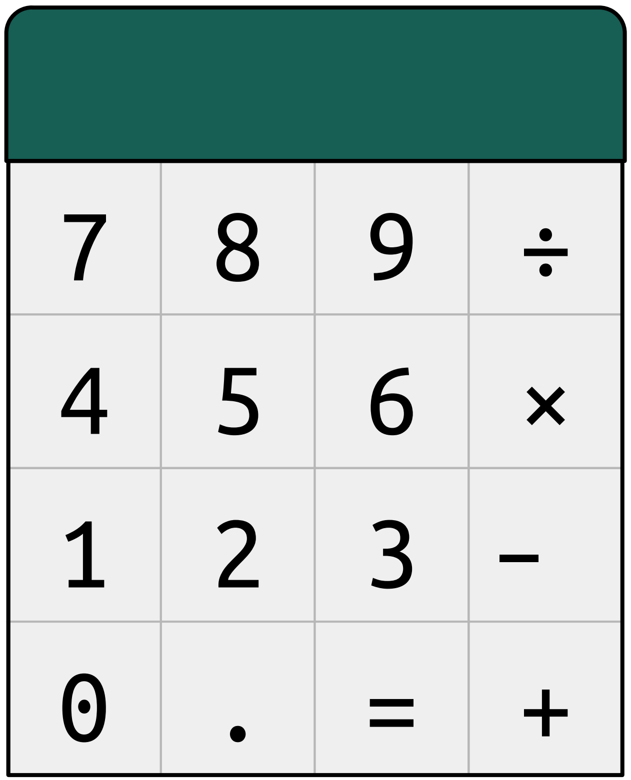
A Tale of Two Computers

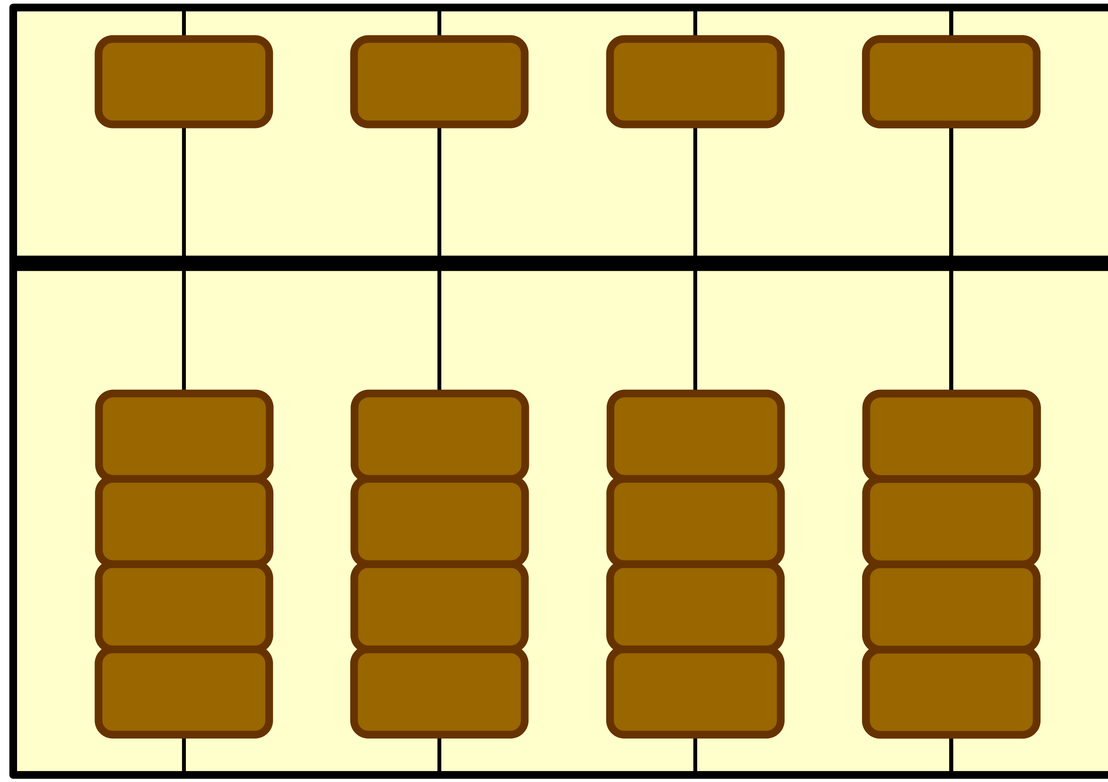

7 8 9 ÷
4 5 6 ×
1 2 3 −
0 . = +
Why does this computer…
…“feel” less powerful than this one?

Calculators vs. Desktops

- A calculator has a ***small amount of memory***. A desktop computer has a ***large amount of memory***.
- A calculator performs a ***fixed set of functions***. A desktop is ***reprogrammable*** and can run many different programs.
- These two distinctions account for much of the difference between “calculator-like” computers and “desktop-esque” computers.
- In CS103, we’ll first explore “small-memory” computers in detail, then discuss “large-memory” computers in depth.

Computing with Finite Memory





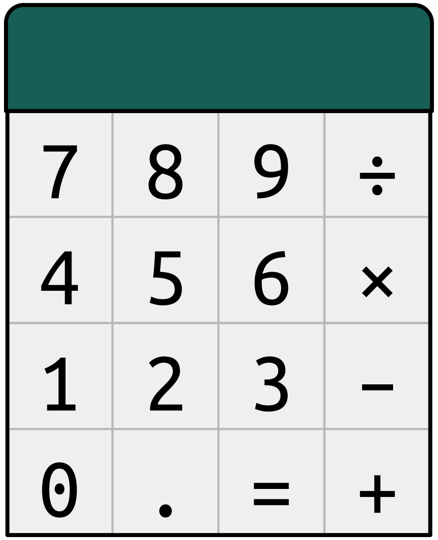
Data stored electronically.
Algorithm is in silicon.
Memory limited by display.

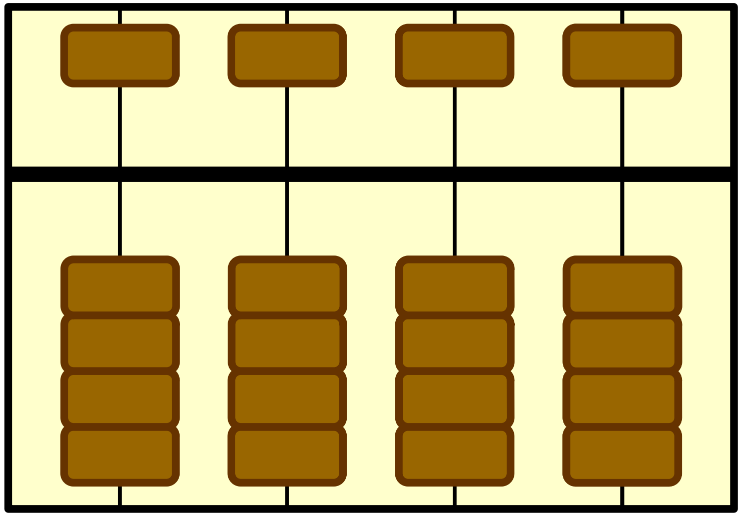
Data stored in wood.
Algorithm is in brain.
Memory limited by beads.

How do we model “memory” and “an algorithm” when they can take on so many forms?

What's in Common?

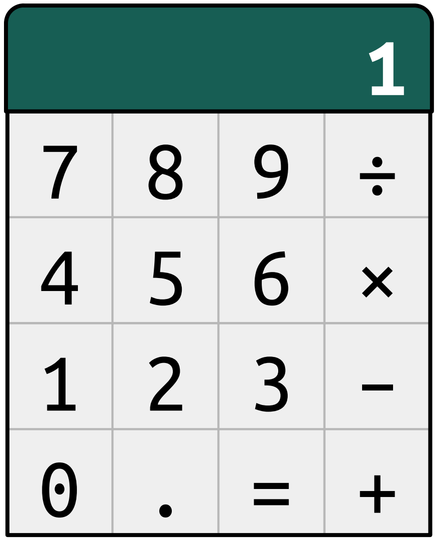
- These machines ***receive input*** from an external source.
- That input is provided ***sequentially***, one discrete unit at a time.
- Each input causes the device to ***change configuration***. This change, big or small, is where the computation happens.

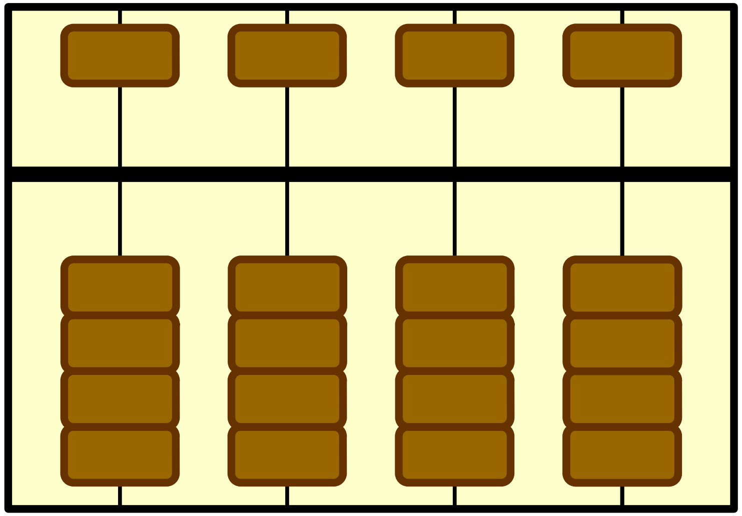




What's in Common?

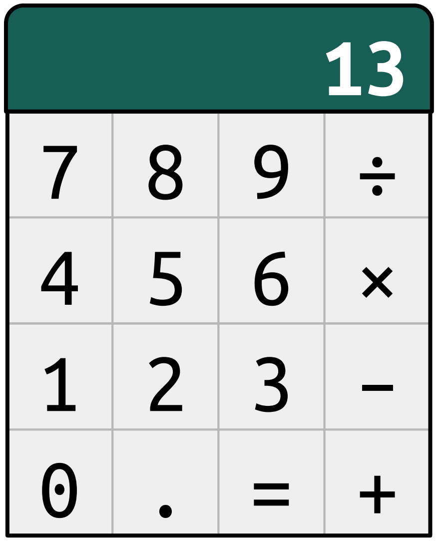
- These machines ***receive input*** from an external source.
- That input is provided ***sequentially***, one discrete unit at a time.
- Each input causes the device to ***change configuration***. This change, big or small, is where the computation happens.

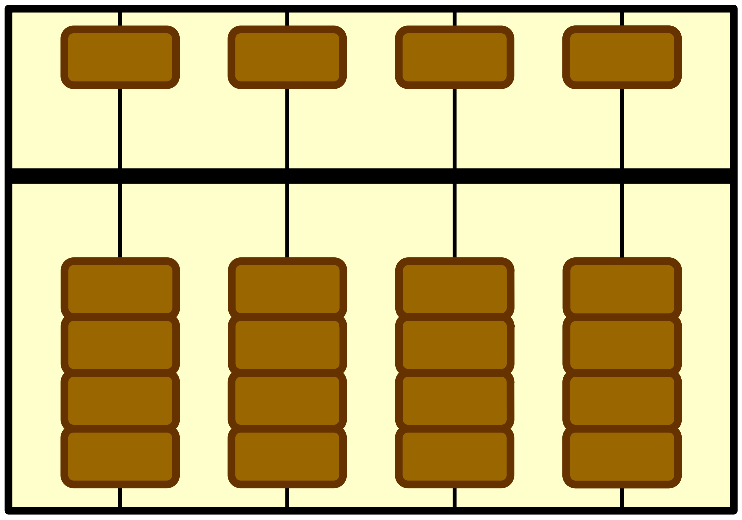




What's in Common?

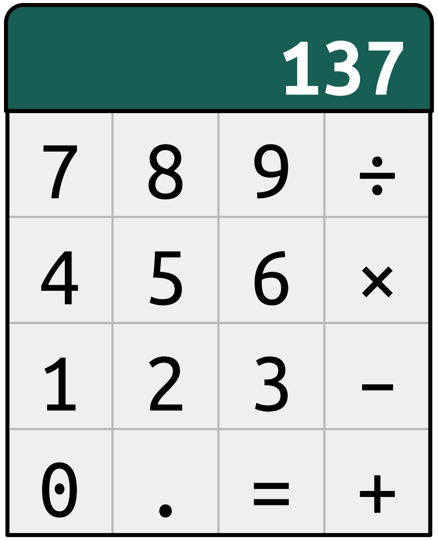
- These machines ***receive input*** from an external source.
- That input is provided ***sequentially***, one discrete unit at a time.
- Each input causes the device to ***change configuration***. This change, big or small, is where the computation happens.

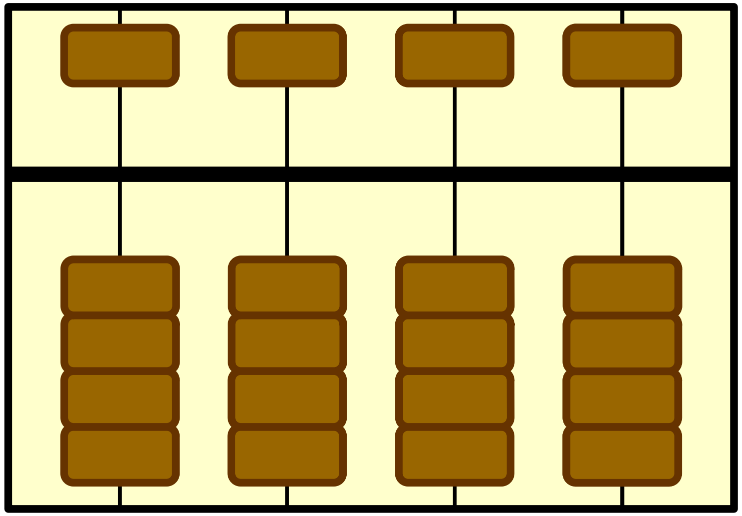




What's in Common?

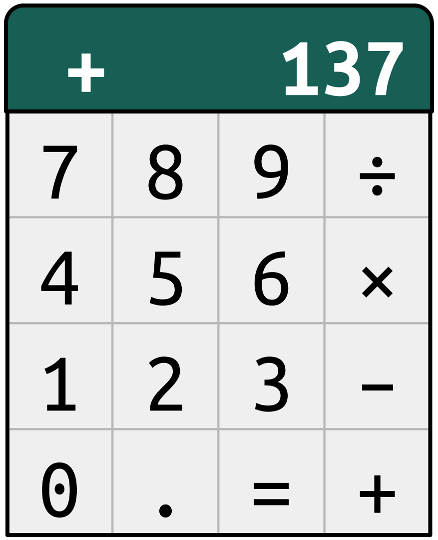
- These machines ***receive input*** from an external source.
- That input is provided ***sequentially***, one discrete unit at a time.
- Each input causes the device to ***change configuration***. This change, big or small, is where the computation happens.

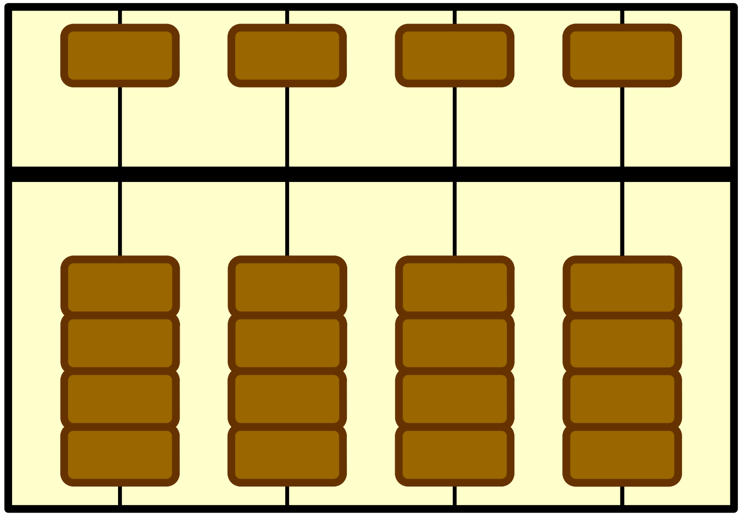




What's in Common?

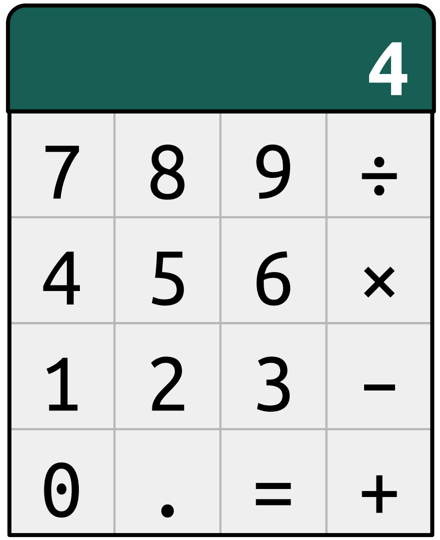
- These machines ***receive input*** from an external source.
- That input is provided ***sequentially***, one discrete unit at a time.
- Each input causes the device to ***change configuration***. This change, big or small, is where the computation happens.

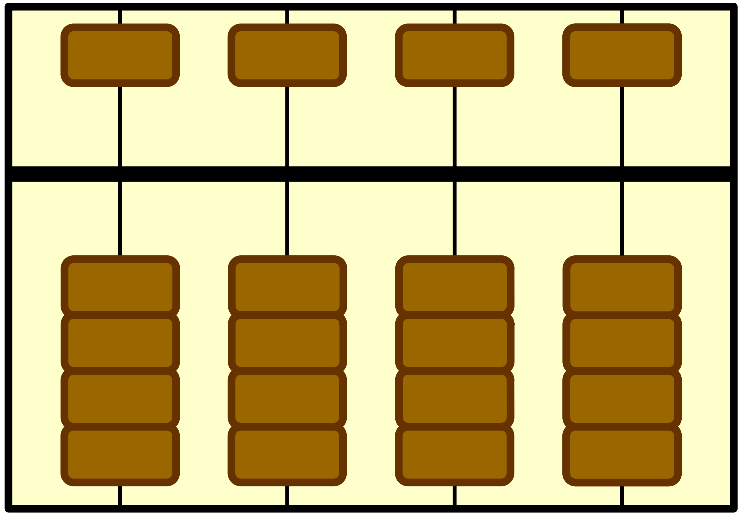




What's in Common?

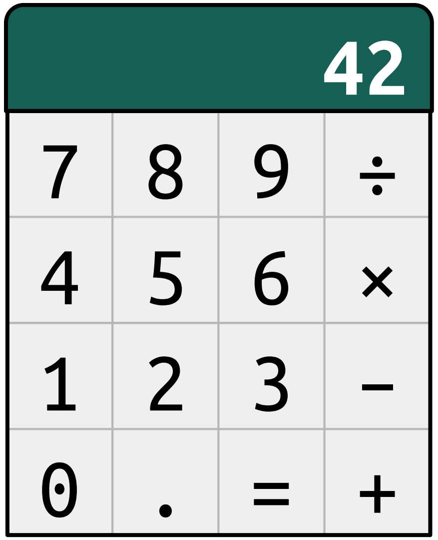
- These machines ***receive input*** from an external source.
- That input is provided ***sequentially***, one discrete unit at a time.
- Each input causes the device to ***change configuration***. This change, big or small, is where the computation happens.

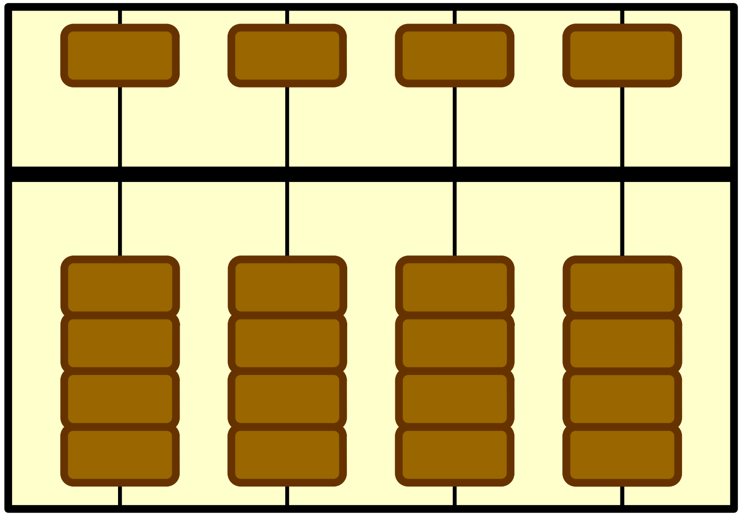




What’s in Common?

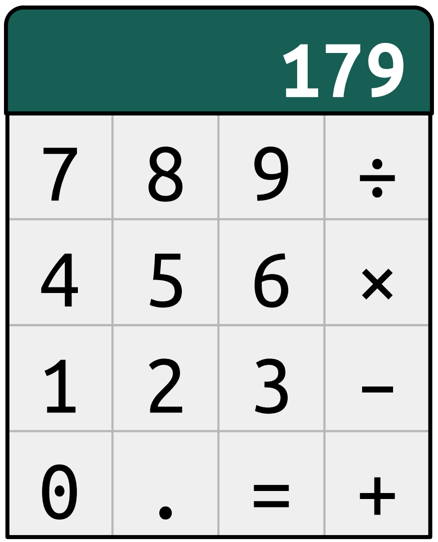
- These machines ***receive input*** from an external source.
- That input is provided ***sequentially***, one discrete unit at a time.
- Each input causes the device to ***change configuration***. This change, big or small, is where the computation happens.

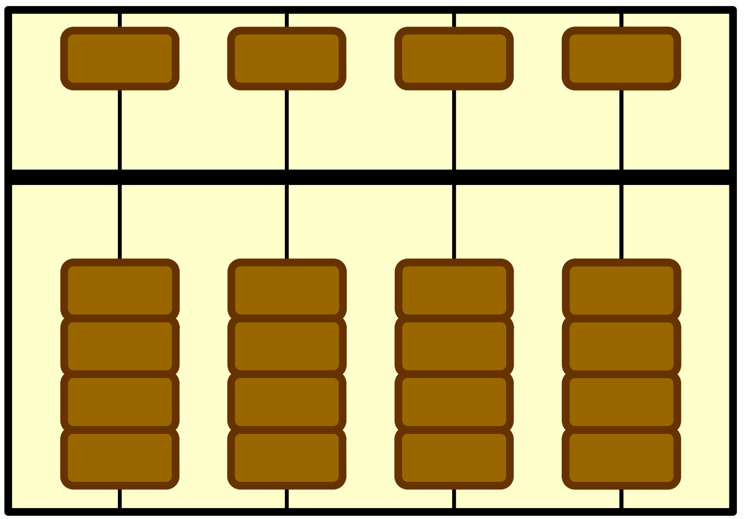




What’s in Common?

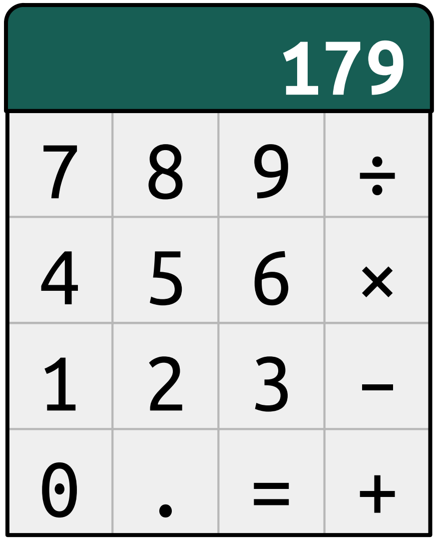
- These machines ***receive input*** from an external source.
- That input is provided ***sequentially***, one discrete unit at a time.
- Each input causes the device to ***change configuration***. This change, big or small, is where the computation happens.

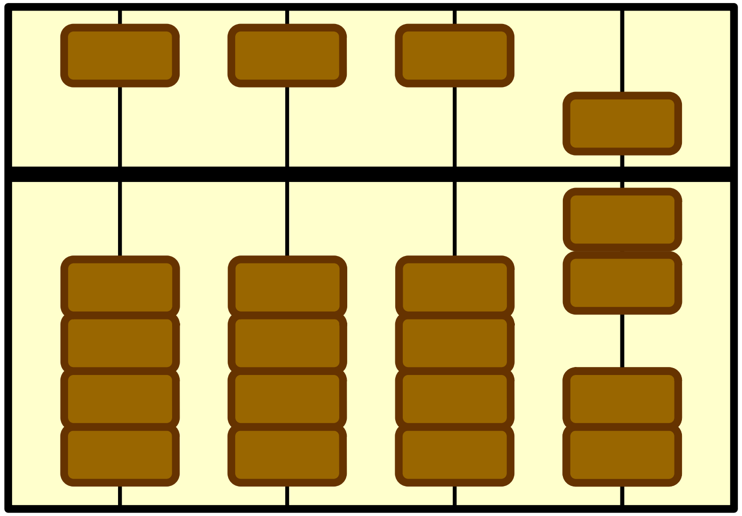




What's in Common?

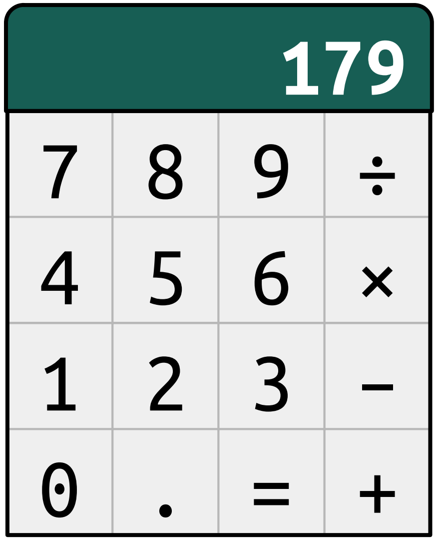
- These machines ***receive input*** from an external source.
- That input is provided ***sequentially***, one discrete unit at a time.
- Each input causes the device to ***change configuration***. This change, big or small, is where the computation happens.

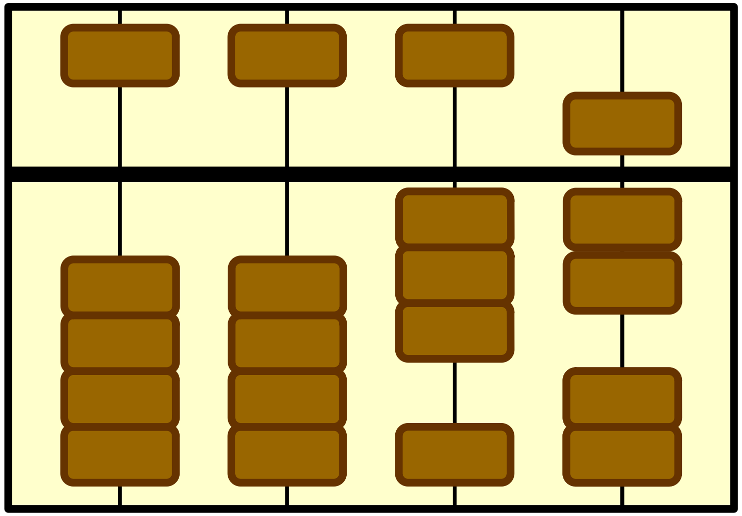




What’s in Common?

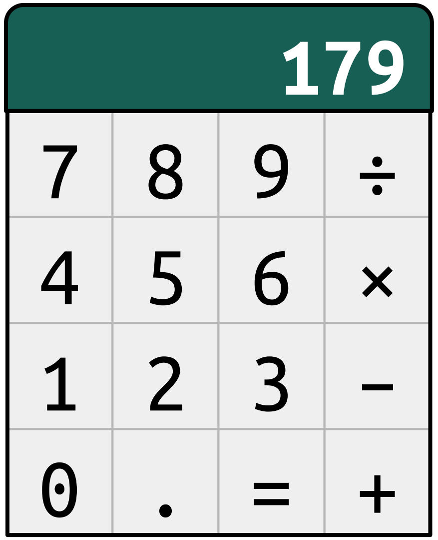
- These machines ***receive input*** from an external source.
- That input is provided ***sequentially***, one discrete unit at a time.
- Each input causes the device to ***change configuration***. This change, big or small, is where the computation happens.

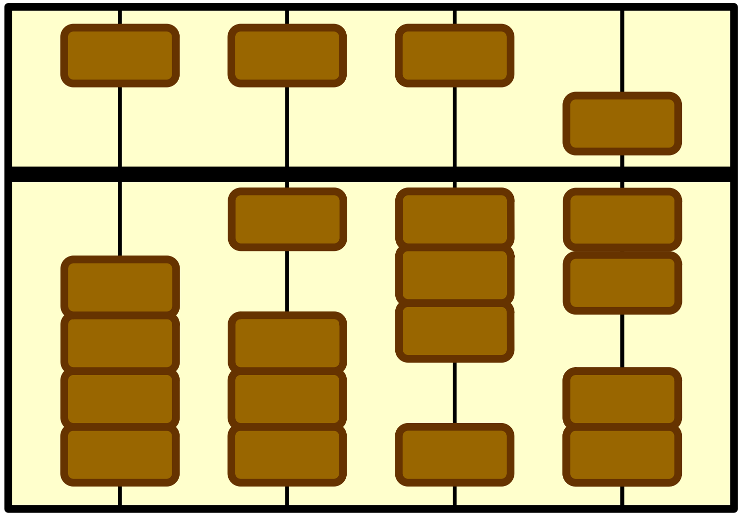




What's in Common?

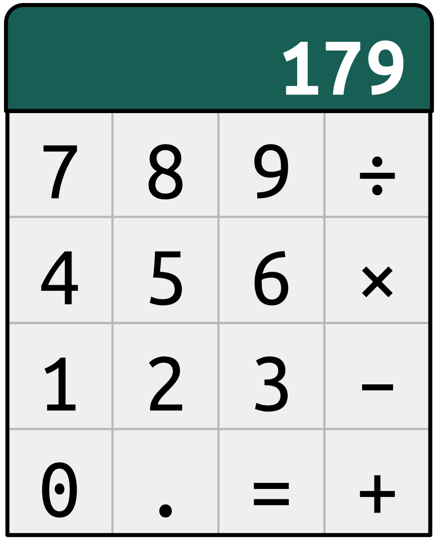
- These machines ***receive input*** from an external source.
- That input is provided ***sequentially***, one discrete unit at a time.
- Each input causes the device to ***change configuration***. This change, big or small, is where the computation happens.

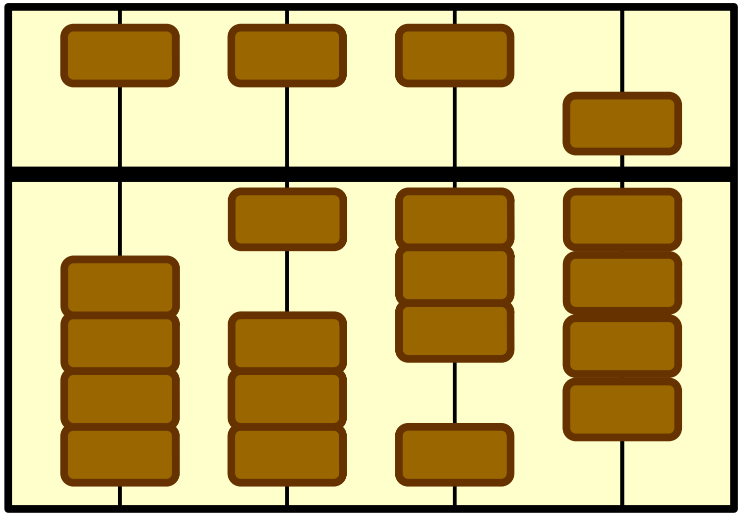




What's in Common?

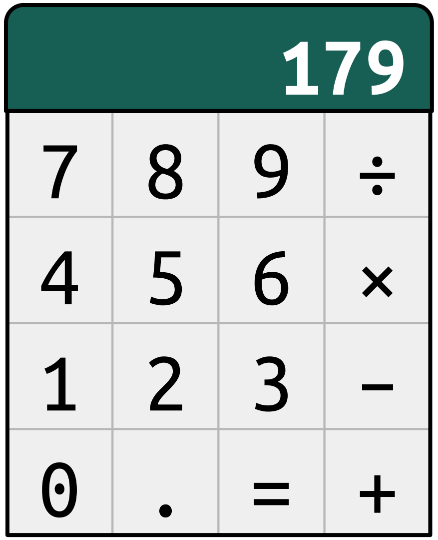
- These machines ***receive input*** from an external source.
- That input is provided ***sequentially***, one discrete unit at a time.
- Each input causes the device to ***change configuration***. This change, big or small, is where the computation happens.

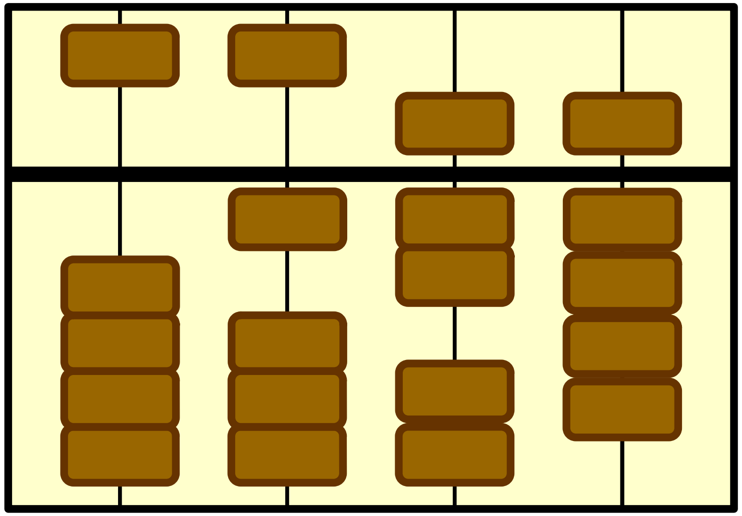




What's in Common?

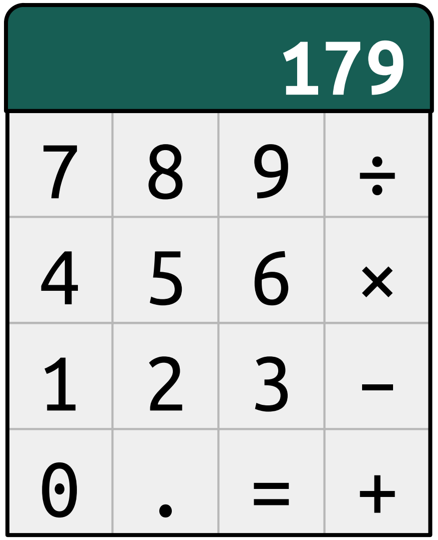
- These machines ***receive input*** from an external source.
- That input is provided ***sequentially***, one discrete unit at a time.
- Each input causes the device to ***change configuration***. This change, big or small, is where the computation happens.

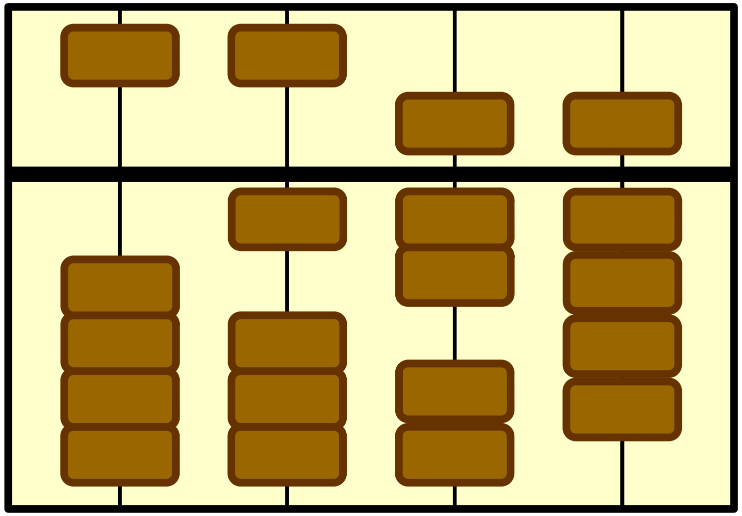




What's in Common?

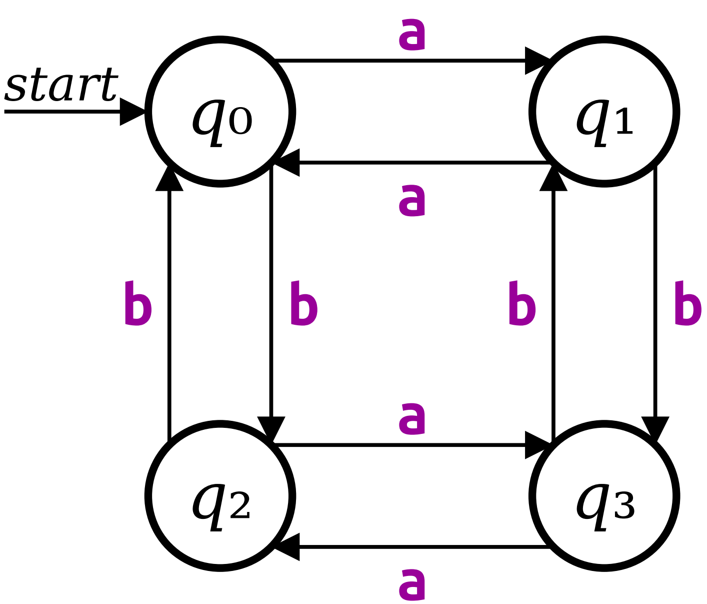
- These machines ***receive input*** from an external source.
- That input is provided ***sequentially***, one discrete unit at a time.
- Each input causes the device to ***change configuration***. This change, big or small, is where the computation happens.
- Once all input is provided, we can ***read off an answer*** based on the configuration of the device.





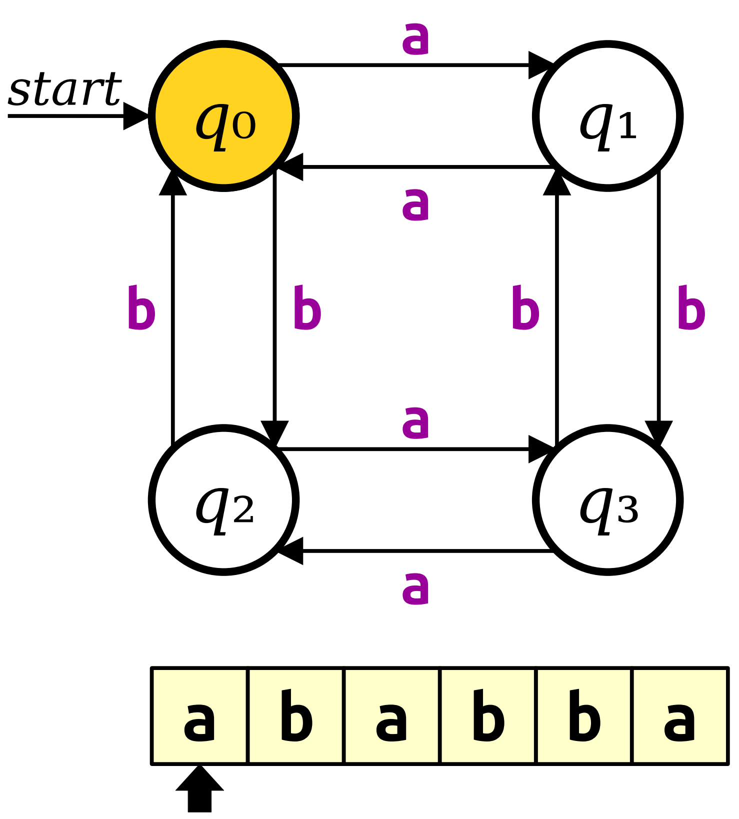
Modeling Finite Computation

- We will model a finite-memory computer as a collection of ***states*** linked by ***transitions***.
- Each state corresponds to one possible configuration of the device's memory.
- Each transition indicates how memory changes in response to inputs.
- Some state is designated as the ***start state***. The computation begins in that state.



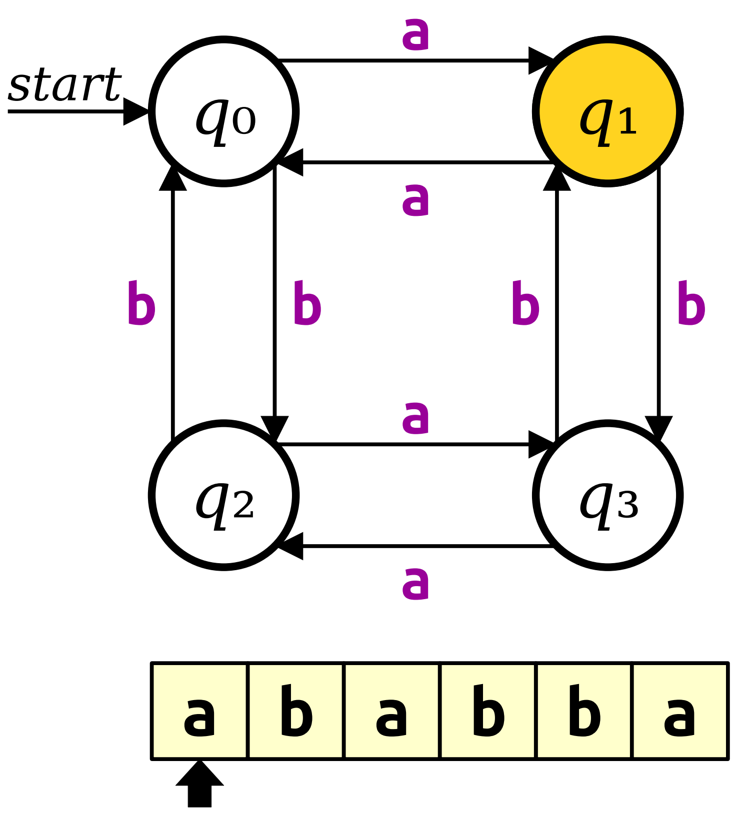
Modeling Finite Computation

- This device processes ***strings*** made of ***characters***.
  - Each character represents some external input to the device.
  - The string represents the full sequence of inputs to the device.
- To run this device, we begin in our start state and scan the input from left to right.
- Each time the machine sees a character, it ***changes state*** by following the transition labeled with that character.



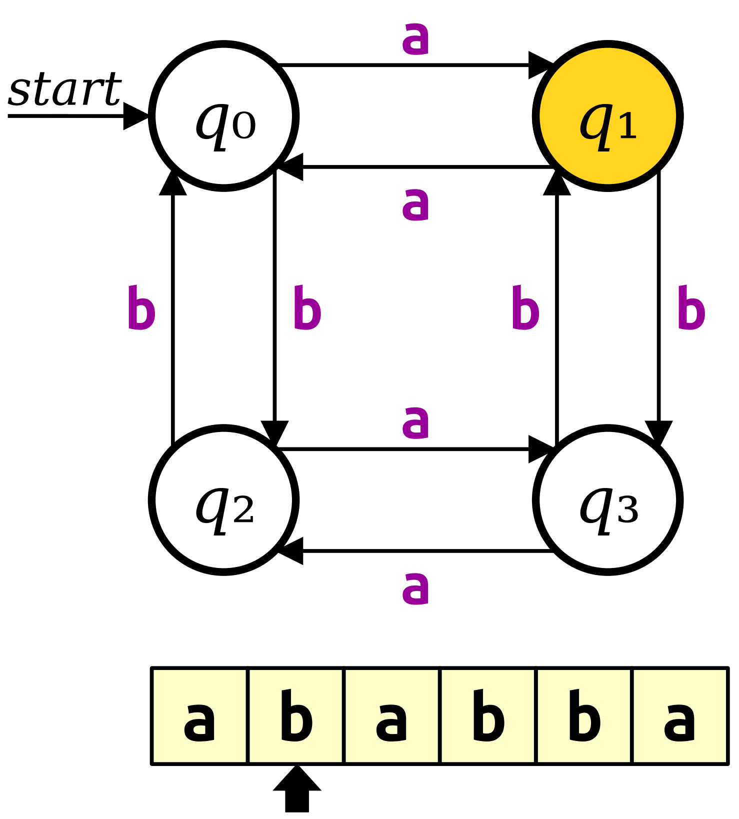
Modeling Finite Computation

- This device processes ***strings*** made of ***characters***.
  - Each character represents some external input to the device.
  - The string represents the full sequence of inputs to the device.
- To run this device, we begin in our start state and scan the input from left to right.
- Each time the machine sees a character, it ***changes state*** by following the transition labeled with that character.



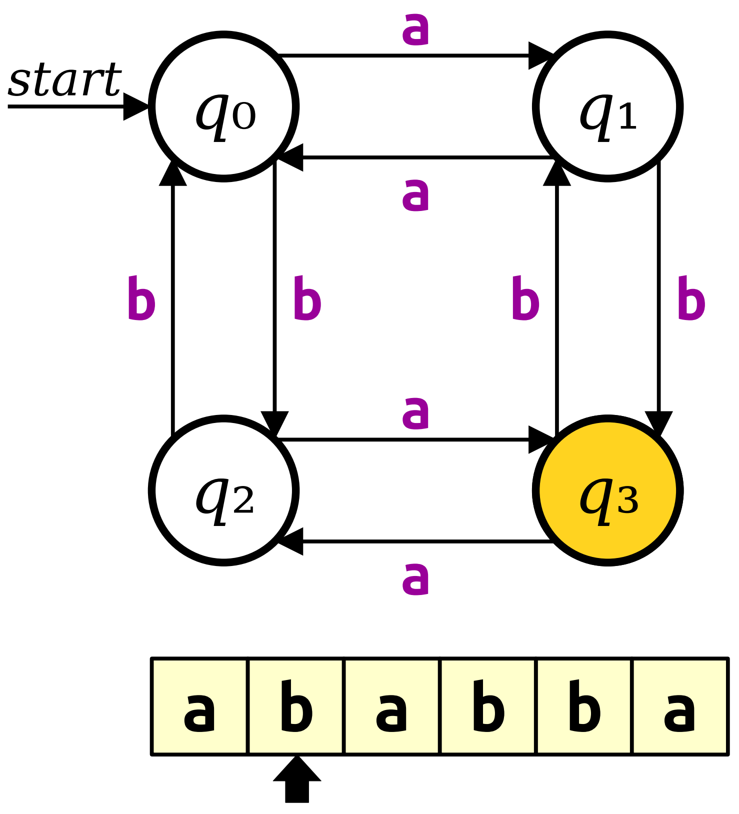
Modeling Finite Computation

- This device processes ***strings*** made of ***characters***.
  - Each character represents some external input to the device.
  - The string represents the full sequence of inputs to the device.
- To run this device, we begin in our start state and scan the input from left to right.
- Each time the machine sees a character, it ***changes state*** by following the transition labeled with that character.



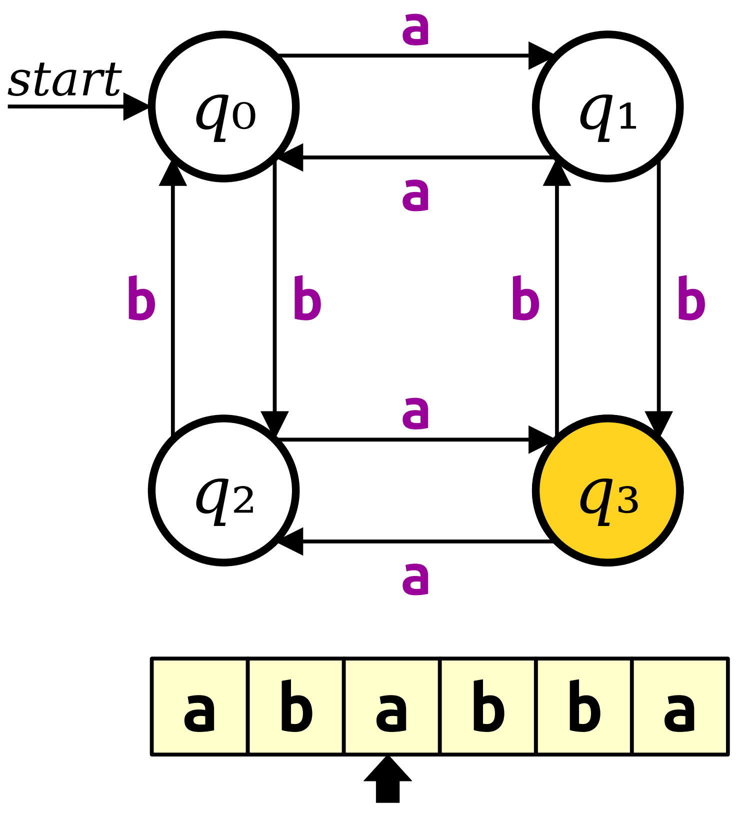
Modeling Finite Computation

- This device processes ***strings*** made of ***characters***.
  - Each character represents some external input to the device.
  - The string represents the full sequence of inputs to the device.
- To run this device, we begin in our start state and scan the input from left to right.
- Each time the machine sees a character, it ***changes state*** by following the transition labeled with that character.



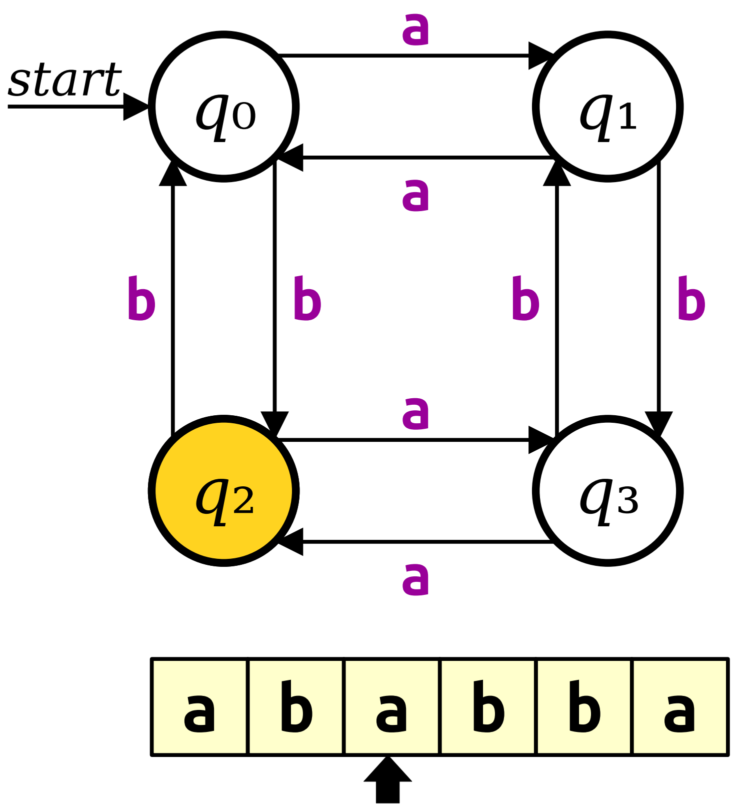
Modeling Finite Computation

- This device processes ***strings*** made of ***characters***.
  - Each character represents some external input to the device.
  - The string represents the full sequence of inputs to the device.
- To run this device, we begin in our start state and scan the input from left to right.
- Each time the machine sees a character, it ***changes state*** by following the transition labeled with that character.



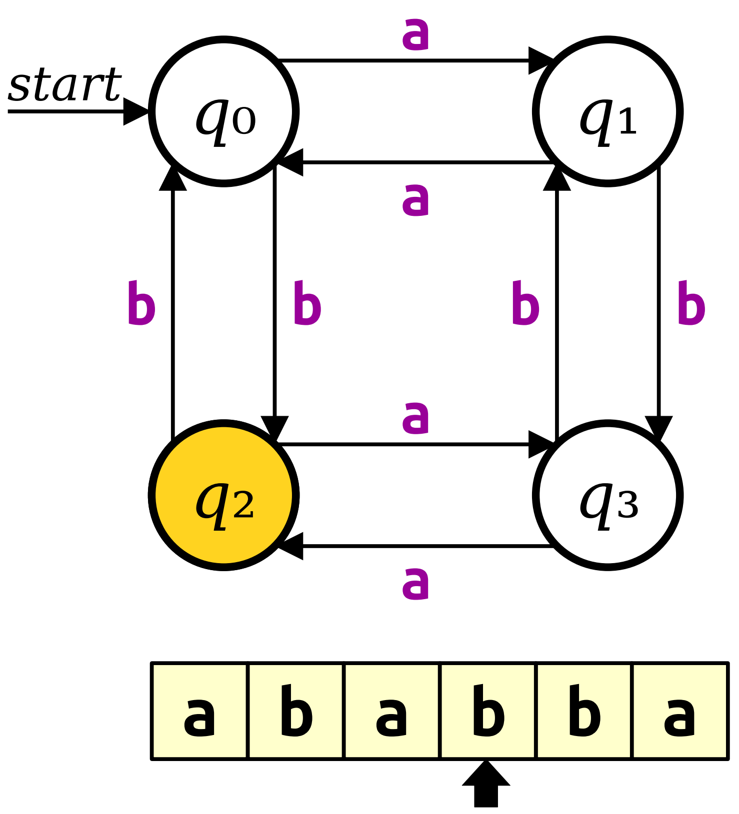
Modeling Finite Computation

- This device processes ***strings*** made of ***characters***.
  - Each character represents some external input to the device.
  - The string represents the full sequence of inputs to the device.
- To run this device, we begin in our start state and scan the input from left to right.
- Each time the machine sees a character, it ***changes state*** by following the transition labeled with that character.



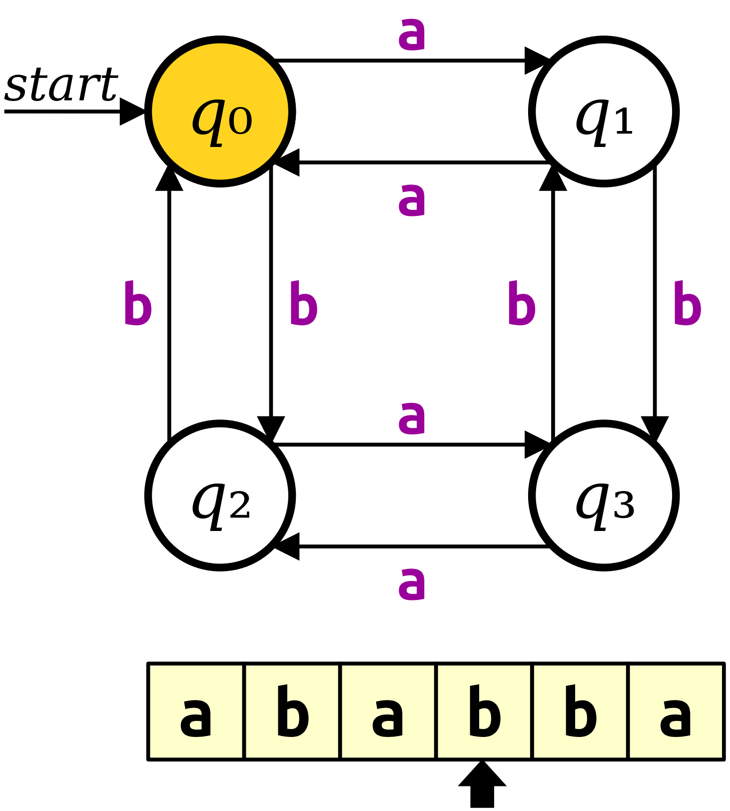
Modeling Finite Computation

- This device processes ***strings*** made of ***characters***.
  - Each character represents some external input to the device.
  - The string represents the full sequence of inputs to the device.
- To run this device, we begin in our start state and scan the input from left to right.
- Each time the machine sees a character, it ***changes state*** by following the transition labeled with that character.



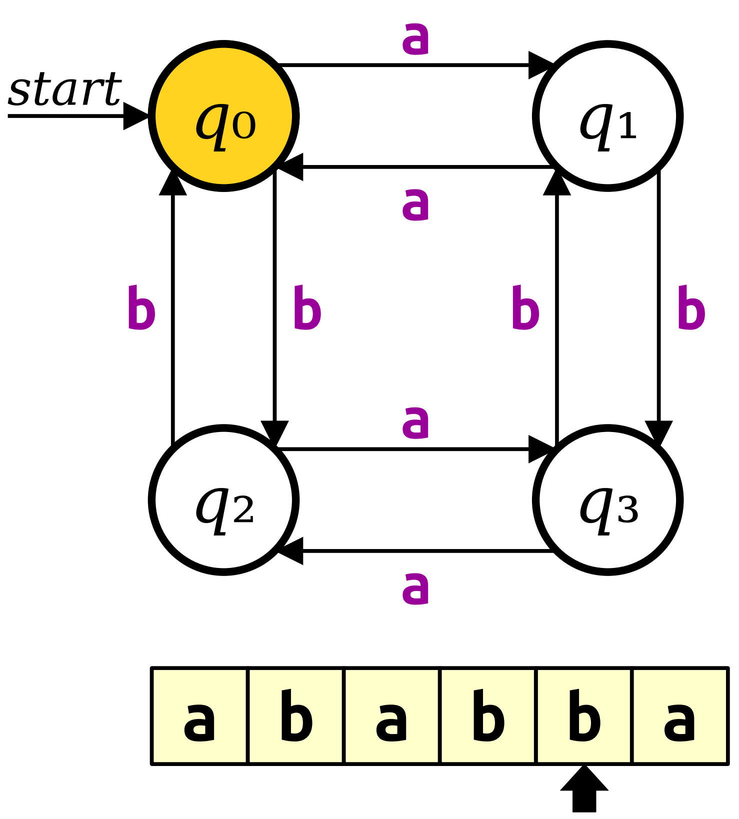
Modeling Finite Computation

- This device processes ***strings*** made of ***characters***.
  - Each character represents some external input to the device.
  - The string represents the full sequence of inputs to the device.
- To run this device, we begin in our start state and scan the input from left to right.
- Each time the machine sees a character, it ***changes state*** by following the transition labeled with that character.



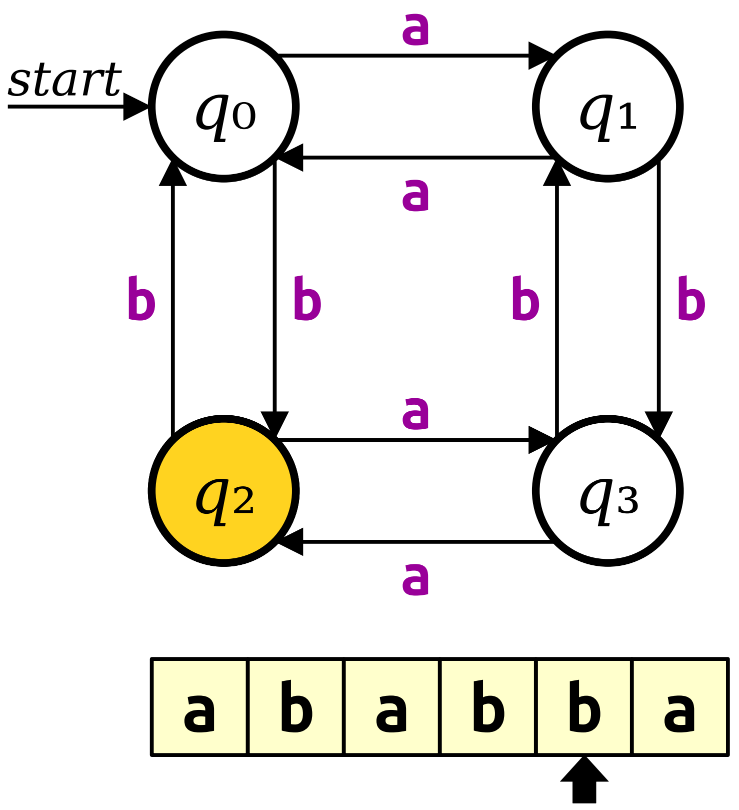
Modeling Finite Computation

- This device processes ***strings*** made of ***characters***.
  - Each character represents some external input to the device.
  - The string represents the full sequence of inputs to the device.
- To run this device, we begin in our start state and scan the input from left to right.
- Each time the machine sees a character, it ***changes state*** by following the transition labeled with that character.



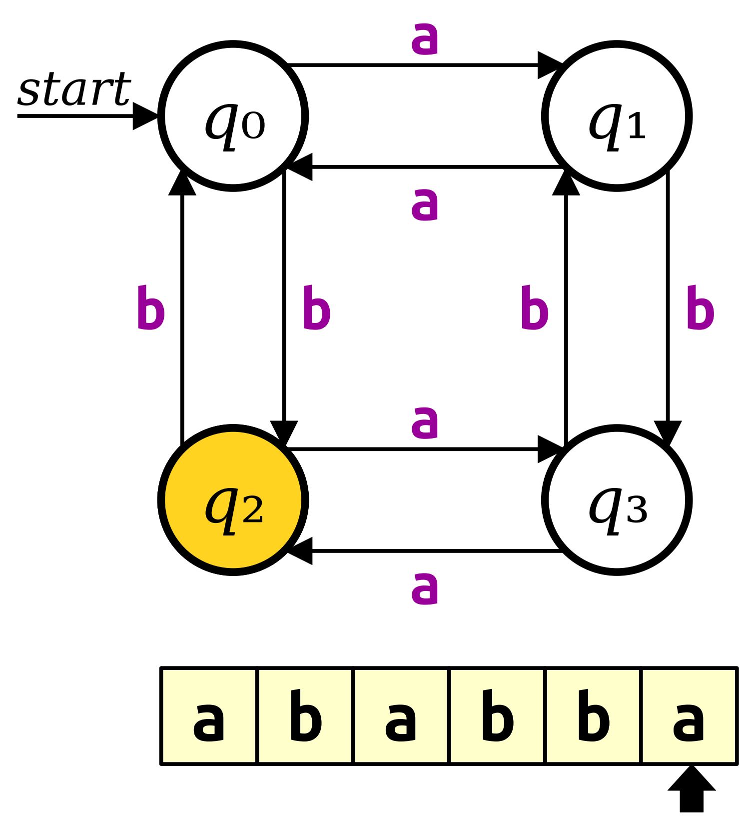
Modeling Finite Computation

- This device processes ***strings*** made of ***characters***.
  - Each character represents some external input to the device.
  - The string represents the full sequence of inputs to the device.
- To run this device, we begin in our start state and scan the input from left to right.
- Each time the machine sees a character, it ***changes state*** by following the transition labeled with that character.



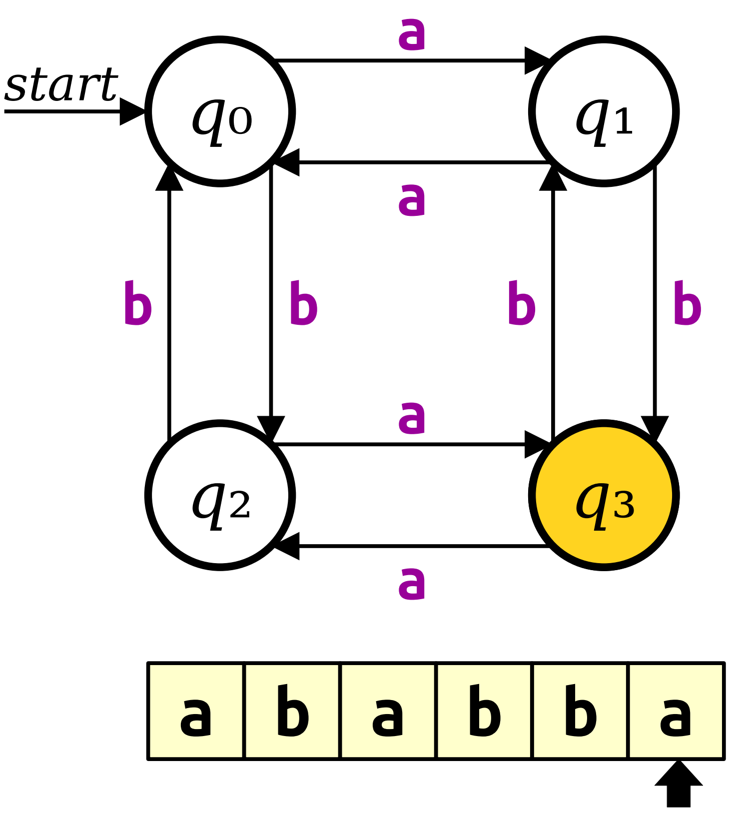
Modeling Finite Computation

- This device processes ***strings*** made of ***characters***.
  - Each character represents some external input to the device.
  - The string represents the full sequence of inputs to the device.
- To run this device, we begin in our start state and scan the input from left to right.
- Each time the machine sees a character, it ***changes state*** by following the transition labeled with that character.



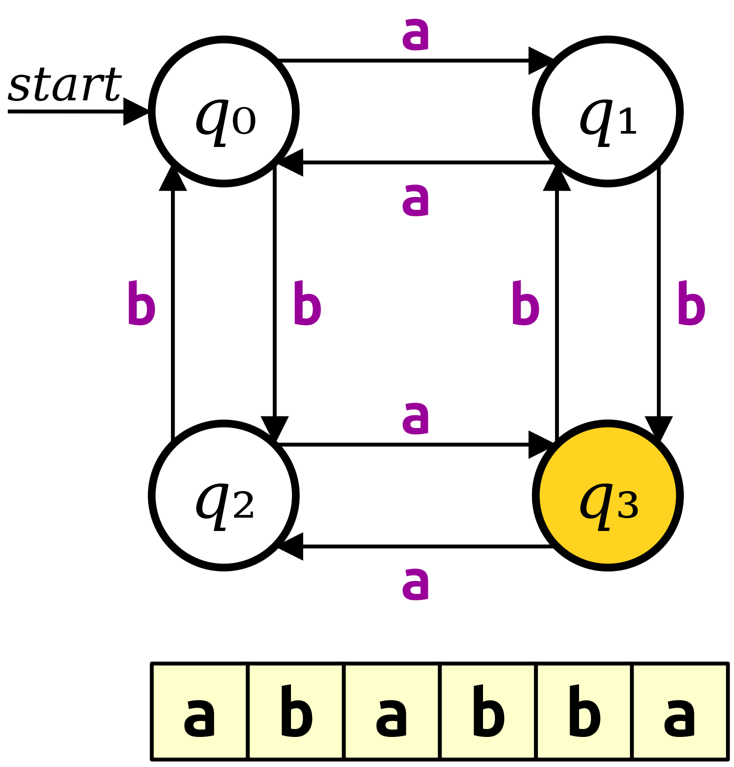
Modeling Finite Computation

- This device processes ***strings*** made of ***characters***.
  - Each character represents some external input to the device.
  - The string represents the full sequence of inputs to the device.
- To run this device, we begin in our start state and scan the input from left to right.
- Each time the machine sees a character, it ***changes state*** by following the transition labeled with that character.



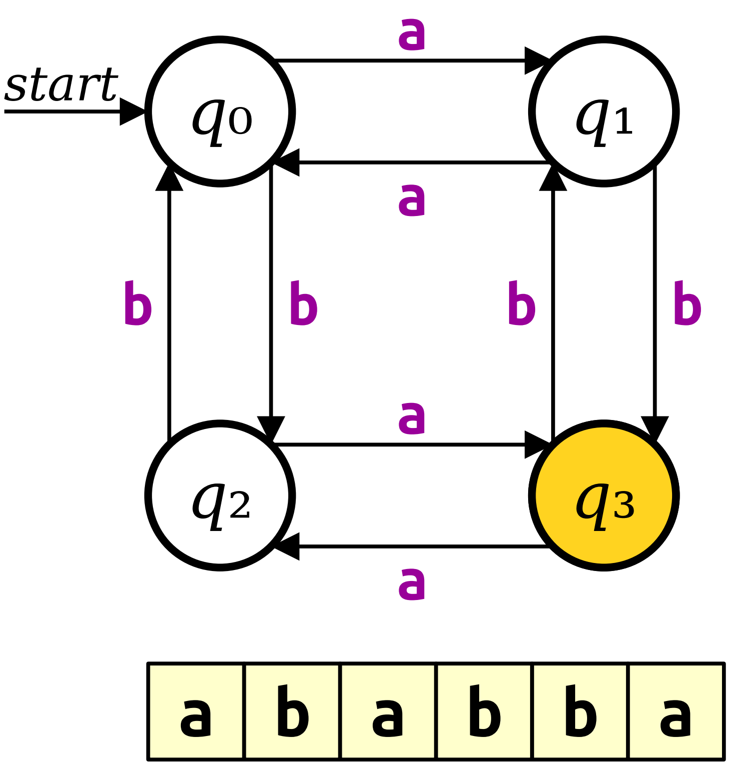
Modeling Finite Computation

- This device processes ***strings*** made of ***characters***.
  - Each character represents some external input to the device.
  - The string represents the full sequence of inputs to the device.
- To run this device, we begin in our start state and scan the input from left to right.
- Each time the machine sees a character, it ***changes state*** by following the transition labeled with that character.



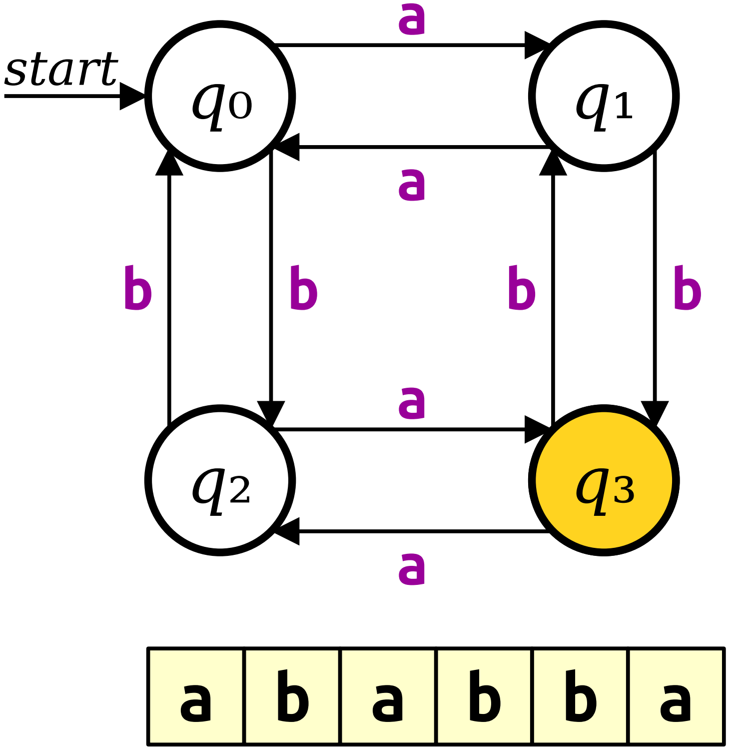
Modeling Finite Computation

- Once we've finished entering all the characters of our input, we need to obtain the result of the computation.
- In general, computers can produce all sorts of things as the result of a computation: a number, a piece of text, etc.
- As a simplifying assumption, we'll assume that we just need to get a single bit of output. That is, our machines will just say `YES` or `NO`.
- (This can be generalized – come talk to me after class if you're curious how!)



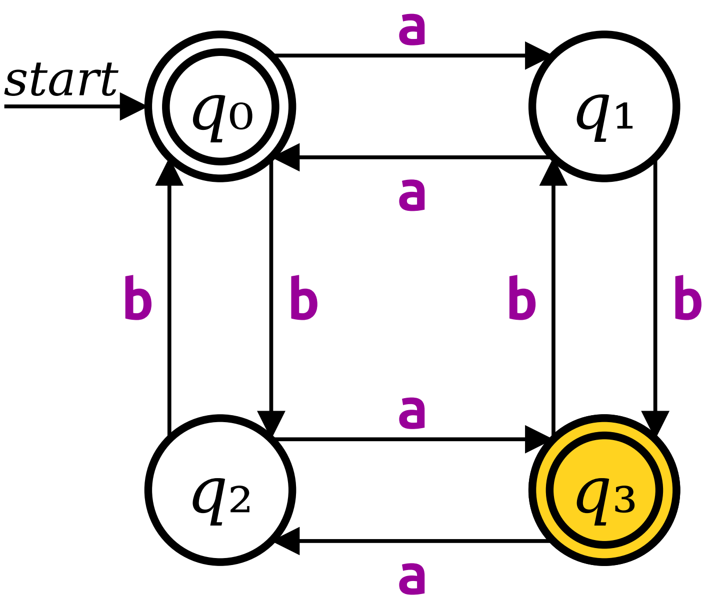
Modeling Finite Computation

- Some of the states in our computational device will be marked as ***accepting states***. These are denoted with a double ring.



Modeling Finite Computation

- Some of the states in our computational device will be marked as ***accepting states***. These are denoted with a double ring.
- If the device ends in an accepting state after seeing all the input, ***accepts*** the input (says `YES`)
- If the device does not end in an accepting state after seeing all the input, it ***rejects*** the input (says `NO`).



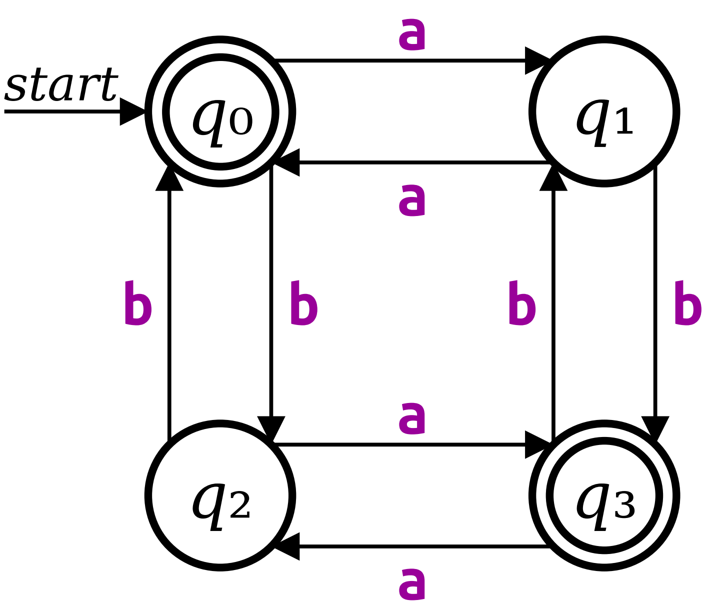
Modeling Finite Computation

- Try it yourself! Which of these strings does this device accept?

aab

aabb

abbababba



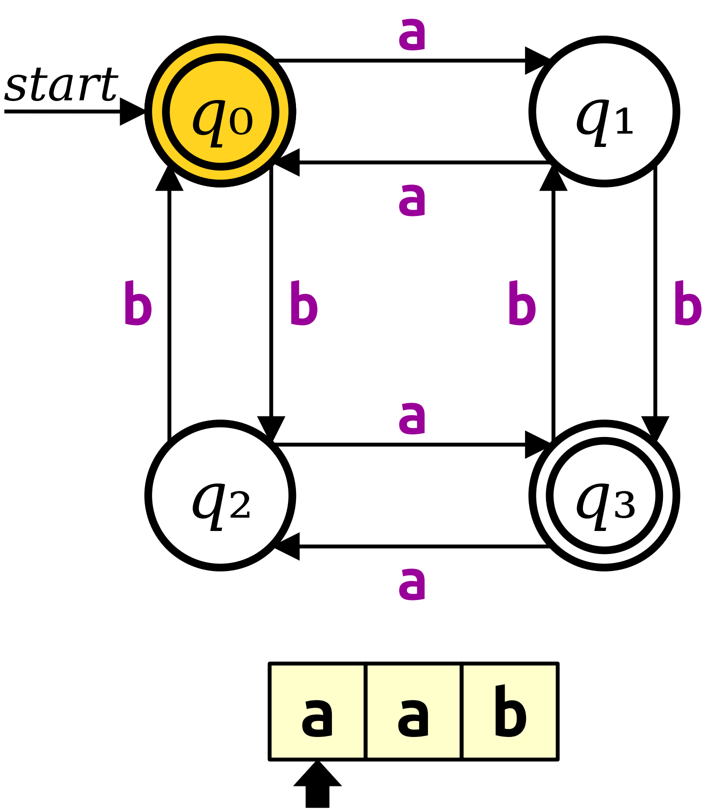
Modeling Finite Computation

- Try it yourself! Which of these strings does this device accept?

aab

aabb

abbababba



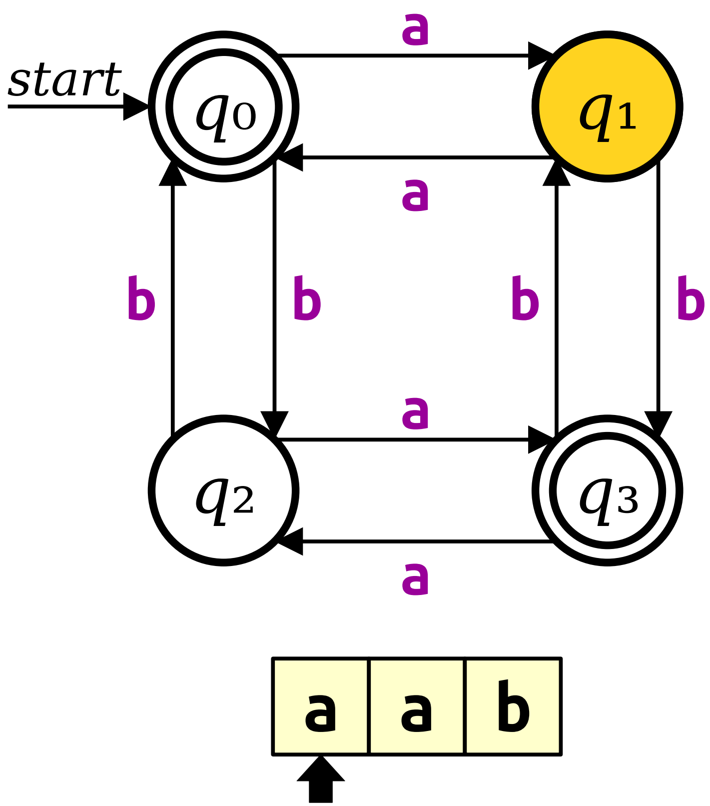
Modeling Finite Computation

- Try it yourself! Which of these strings does this device accept?

aab

aabb

abbababba



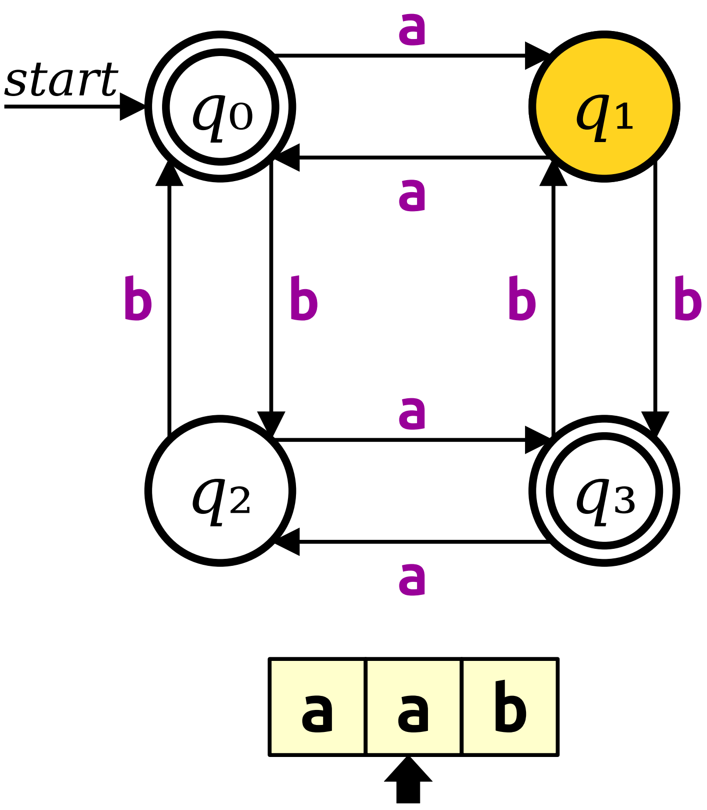
Modeling Finite Computation

- Try it yourself! Which of these strings does this device accept?

aab

aabb

abbababba



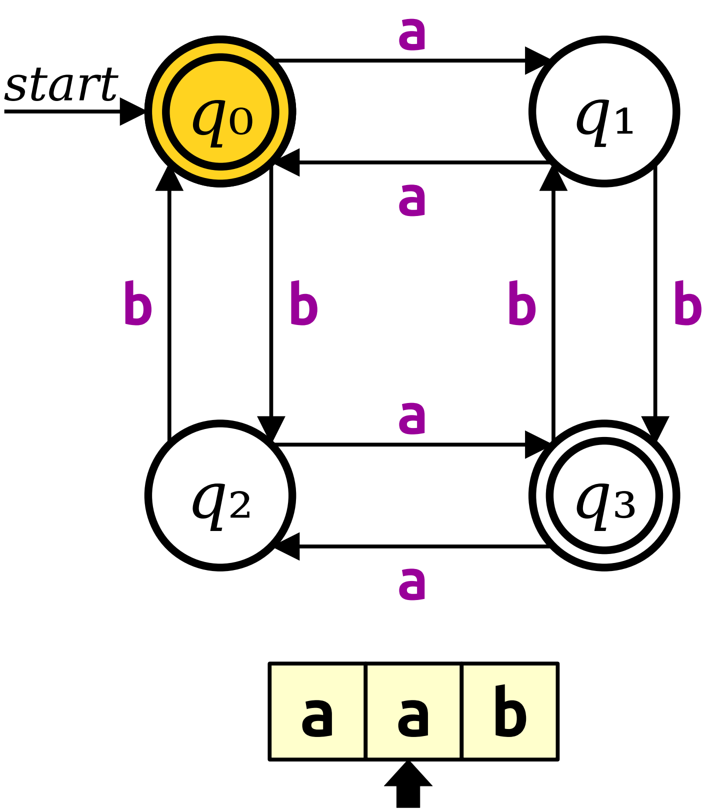
Modeling Finite Computation

- Try it yourself! Which of these strings does this device accept?

aab

aabb

abbababba



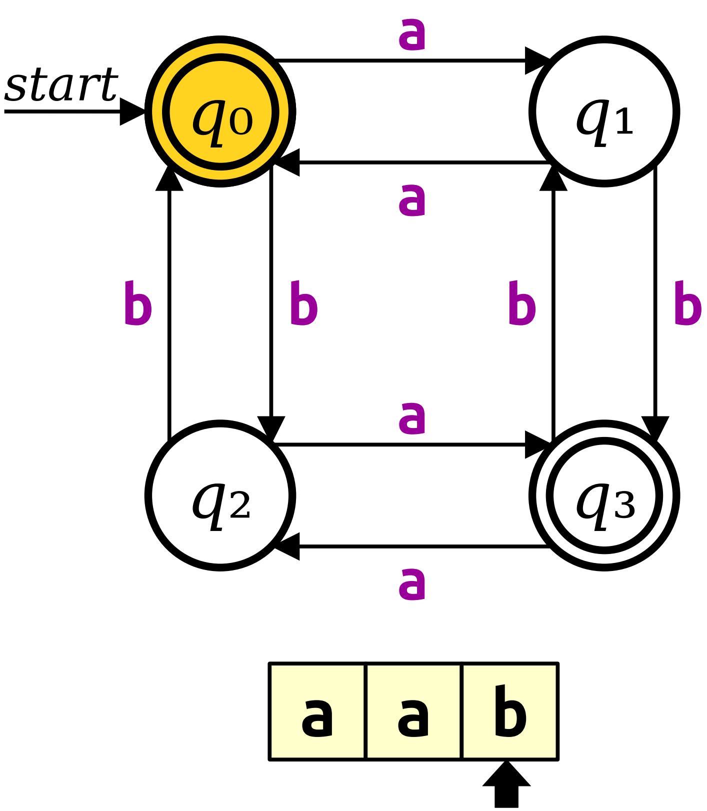
Modeling Finite Computation

- Try it yourself! Which of these strings does this device accept?

aab

aabb

abbababba



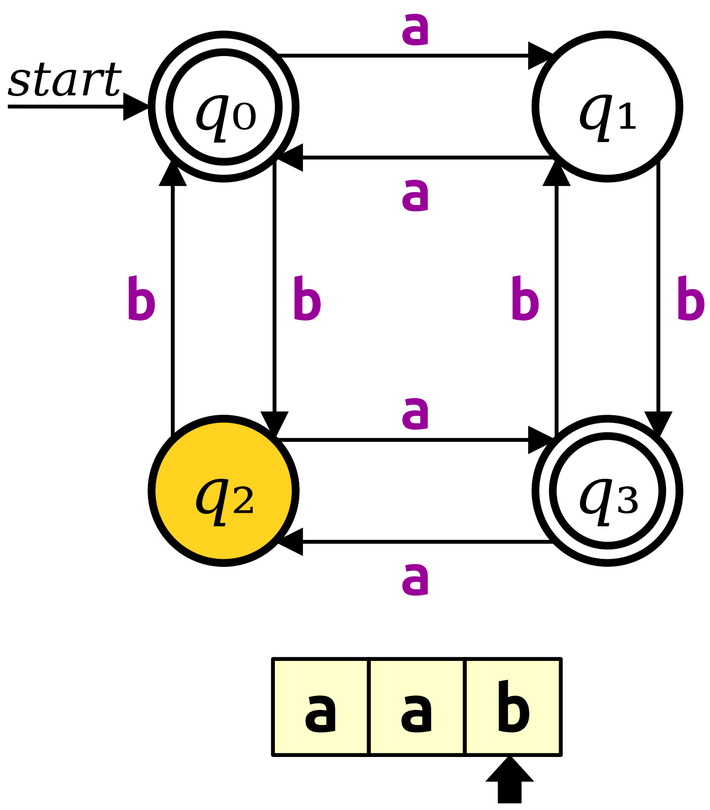
Modeling Finite Computation

- Try it yourself! Which of these strings does this device accept?

aab

aabb

abbababba



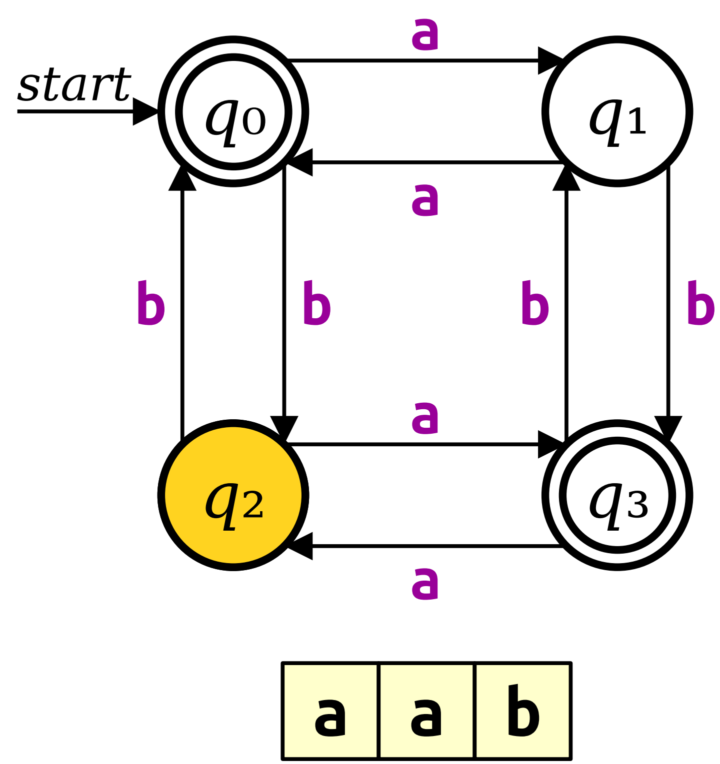
Modeling Finite Computation

- Try it yourself! Which of these strings does this device accept?

aab

aabb

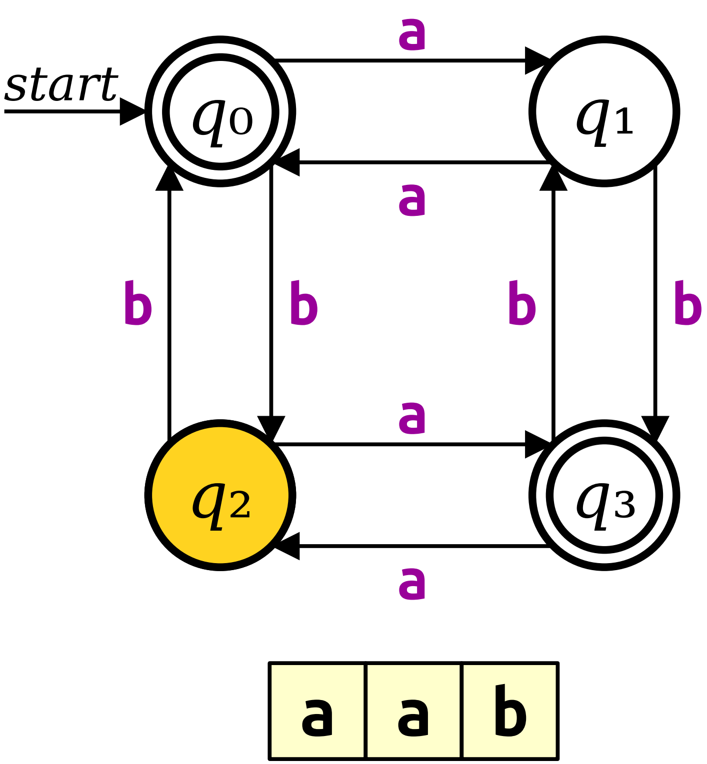
abbababba



Modeling Finite Computation

- Try it yourself! Which of these strings does this device accept?

aab



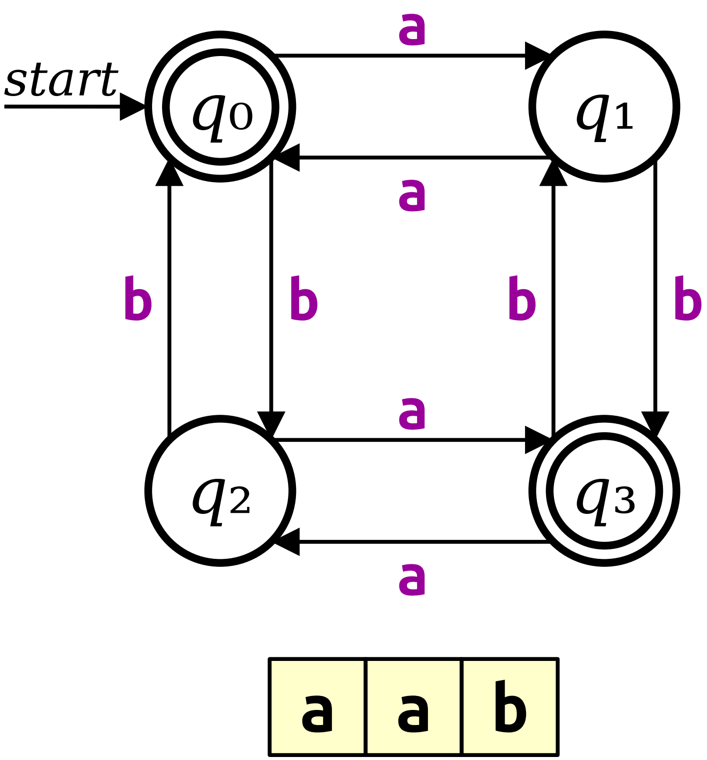
Modeling Finite Computation

- Try it yourself! Which of these strings does this device accept?

aab

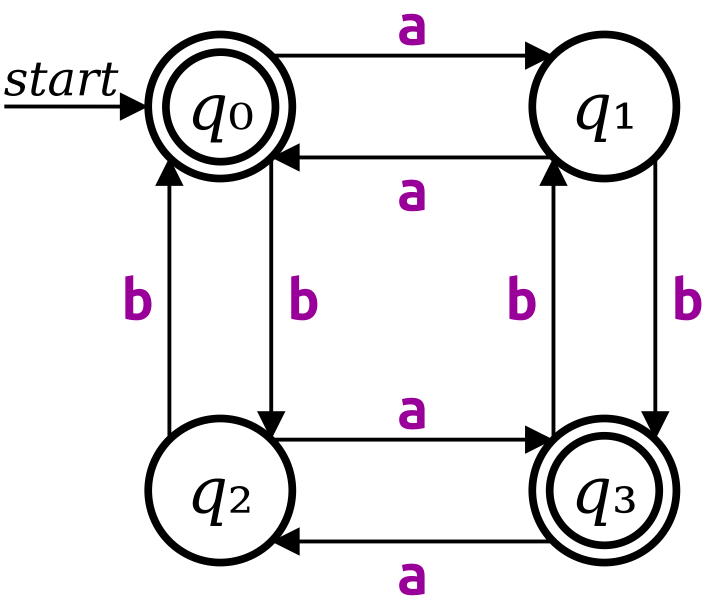
aabb

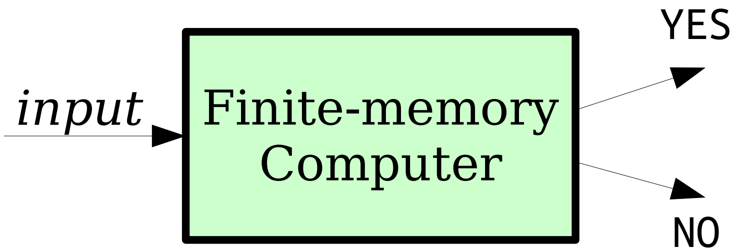
abbababba



Finite Automata

- This type of computational device is called a ***finite automaton*** (plural: ***finite automata***).
- Finite automata model computers where (1) memory is finite and (2) the computation produces as YES/NO answer.
- In other words, finite automata model predicates, and do so with a fixed, finite amount of memory.





Formalizing Things

Strings

- An ***alphabet*** is a finite, nonempty set of symbols called ***characters***.
  - Typically, we use the symbol **Σ** to refer to an alphabet.
- A ***string over an alphabet Σ*** is a finite sequence of characters drawn from Σ.
- Example: Let Σ = {**a**, **b**}. Here are some strings over Σ:

a　　**aabaaabbabaaabaaaabbb**　　**abbababba**

- The ***empty string*** has no characters and is denoted **ε**.
- Calling attention to an earlier point: since all strings are finite sequences of characters from Σ, you cannot have a string of infinite length.

Languages

- A ***formal language*** is a set of strings.
- We say that L is a ***language over Σ*** if it is a set of strings over Σ.
- Example: The language of palindromes over Σ = {**a**, **b**, **c**} is the set
  - {ε, **a**, **b**, **c**, **aa**, **bb**, **cc**, **aaa**, **aba**, **aca**, **bab**, … }
- The set of all strings composed from letters in Σ is denoted **Σ***.
- Formally, we say that L is a language over Σ if $L \subseteq \Sigma^*$.

Mathematical Lookalikes

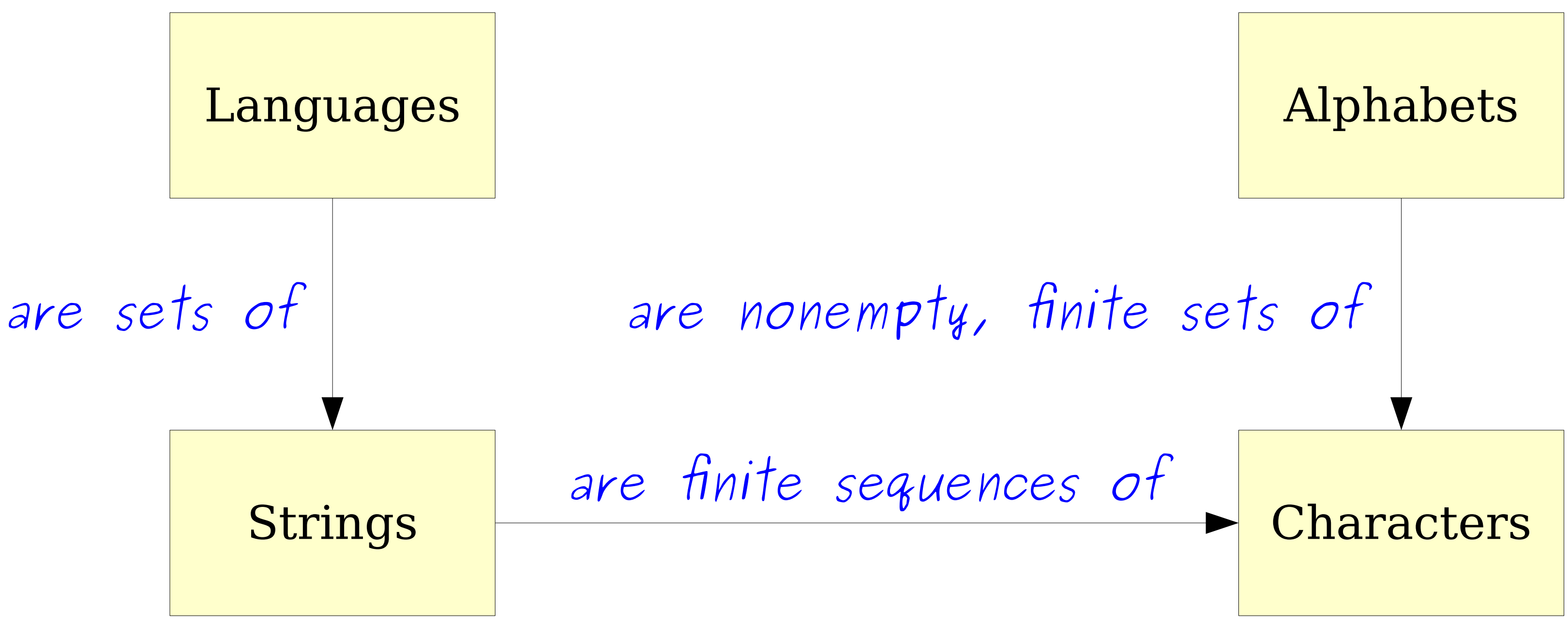
- We now have ∈, ε, Σ, and Σ*. Yikes!
- The symbol ∈ is the ***element-of*** relation.
- The symbol ε is the ***empty string***.
- The symbol Σ denotes an ***alphabet***.
- The expression Σ* means "all strings that can be made from characters in Σ."
- That lets us write things like

$$\text{We have } \varepsilon \in \Sigma^*, \text{ but } \varepsilon \notin \Sigma.$$

- Ever get confused? ***Just ask!***

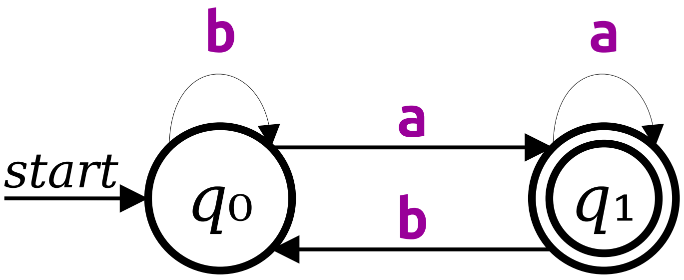
The Cast of Characters

- ***Languages*** are sets of strings.
- ***Strings*** are finite sequences of characters.
- ***Characters*** are individual symbols.
- ***Alphabets*** are sets of characters.



Finite Automata and Languages

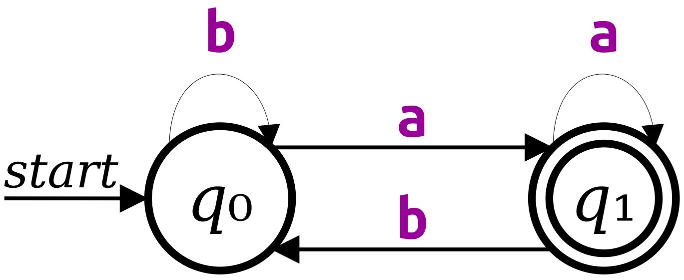
- Let A be an automaton that processes strings drawn from an alphabet Σ.
- The ***language of A***, denoted $\mathscr{L}(A)$, is the set of strings over Σ that A accepts:



$$\mathscr{L}(A) = \{\ w \in \Sigma^* \mid A \text{ accepts } w\ \}$$

Finite Automata and Languages

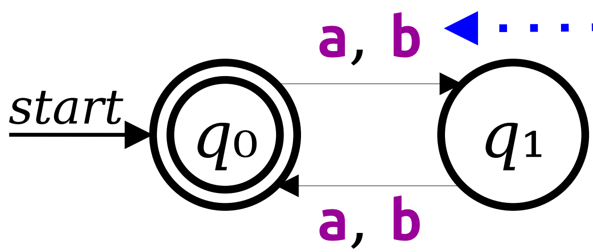
- Let D be the automaton shown to the right. It processes strings over {**a**, **b**}.
- Notice that D accepts all strings of **a**'s and **b**'s that end in **a** and rejects everything else.



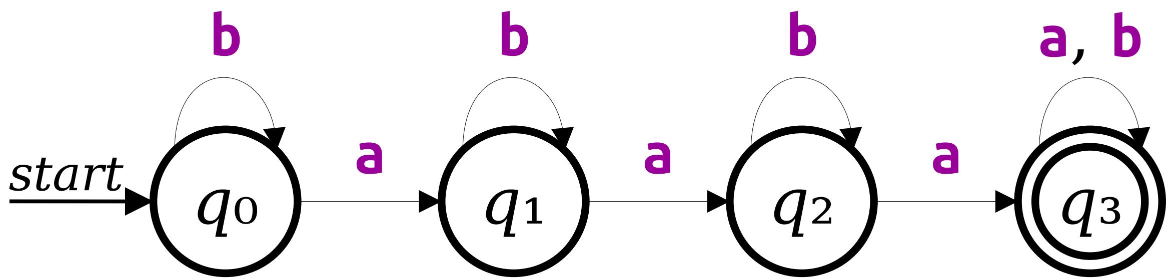
- So $\mathscr{L}(D) = \{ w \in \{\mathtt{a}, \mathtt{b}\}^* \mid w \text{ ends in } \mathtt{a} \}$.

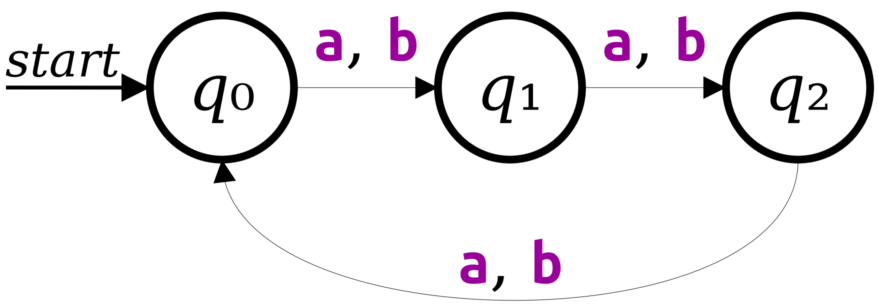
$$\mathscr{L}(A) = \{ w \in \Sigma^* \mid A \text{ accepts } w \}$$

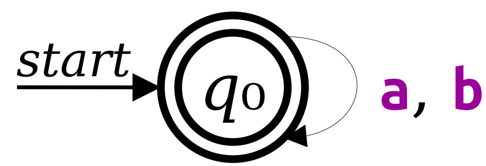
Finite Automata and Languages



This means "take this transition if you see an **a** or a **b**."







$$\mathscr{L}(A) = \{\ w \in \Sigma^* \mid A \text{ accepts } w\ \}$$

The Story So Far

- A ***finite automaton*** is a collection of ***states*** joined by ***transitions***.
- Some state is designated as the ***start state***.
- Some number of states are designated as ***accepting states***.
- The automaton processes a string by beginning in the start state and following the indicated transitions.
- If the automaton ends in an accepting state, it ***accepts*** the input.
- Otherwise, the automaton ***rejects*** the input.
- The ***language*** of an automaton is the set of strings it accepts.

Time-Out For Announcements!

Midterm: Revise and Resubmit

- We've finished grading the midterm exam. Scores and feedback are now available on Gradescope.
- Between now and Wednesday at 2:30PM, you may submit regrade requests if you believe we made any errors during grading.
  - This is for traditional regrade requests, as in "the course staff made an error grading this" rather than "I would like to resubmit this answer."
- ***Revise-and-resubmit runs from Thursday at 2:30PM to Monday at 2:30PM***. During that time, you may submit regrade requests in which you replace your old answers with new ones.
  - You can resubmit answers to any number of questions on the exam.
  - Please submit one regrade request per question where you want to resubmit your answer.
- We'll grade newly-submitted answers next week. Your new score is the maximum of your old score and 85% of your new score.
  - Your score cannot decrease.
  - The maximum possible score you can earn when resubmitting is 85%.

Next Time

- ***Regular Languages***
  - An important class of languages.
- ***Nondeterministic Computation***
  - Why must computation be linear?
- ***NFAs***
  - Automata with Magic Superpowers.