CSE 311 Section 09

Models of Computation

Administrivia

Announcements & Reminders

- HW8
  - Due Yesterday
- HW9
  - Due **Thursday,** March 13th @ **11:00 PM**

Relations!

Relations

R is **reflexive** iff $(a,a) \in R$ for every $a \in A$

R is **symmetric** iff $(a,b) \in R$ implies $(b, a) \in R$

R is **antisymmetric** iff $(a,b) \in R$ and $a \neq b$ implies $(b,a) \notin R$

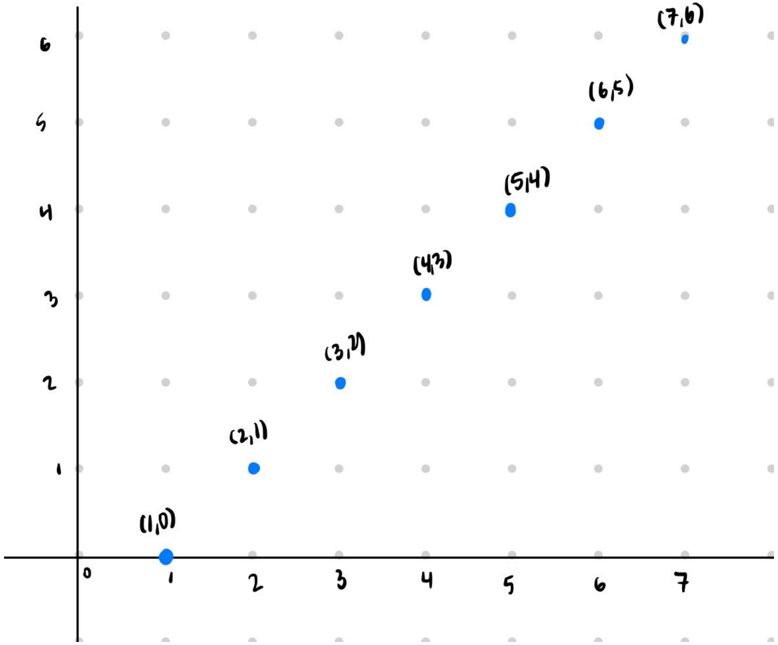
R is **transitive** iff $(a,b) \in R$ and $(b, c) \in R$ implies $(a, c) \in R$

Problem 1a

Let $R = \{(x, y) \ : \ x = y + 1\}$ on $\mathbb{N}$.

Problem 1a

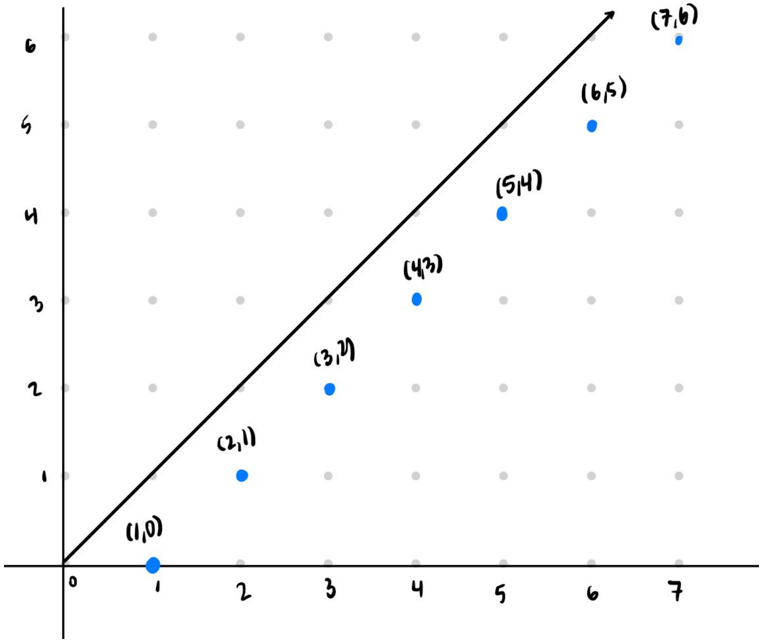
Let $R = \{(x, y) \ : \ x = y + 1\}$ on $\mathbb{N}$.



We can graph the points of R

Problem 1a

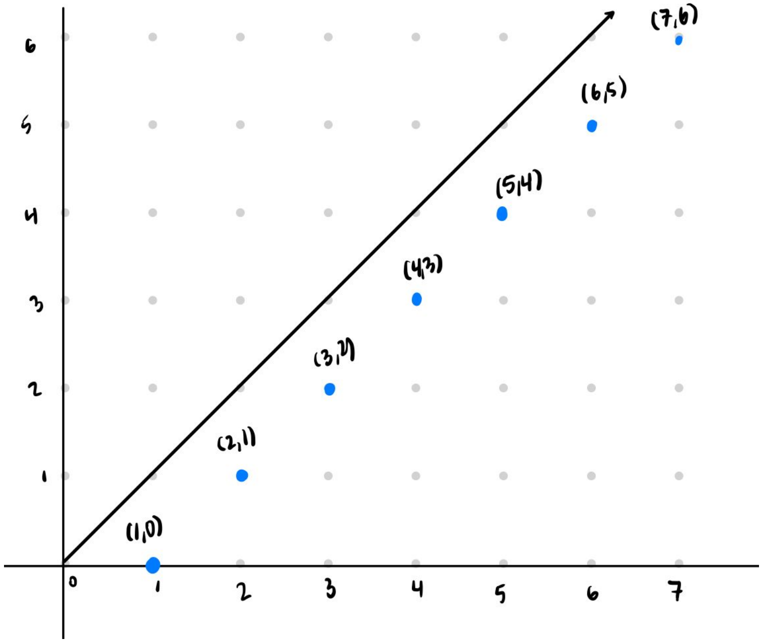
Let $R = \{(x, y) \ : \ x = y + 1\}$ on $\mathbb{N}$.



If all points on the line of y = x are in the relation then the relation is reflexive

Problem 1a

Let $R = \{(x, y) \ : \ x = y + 1\}$ on $\mathbb{N}$.

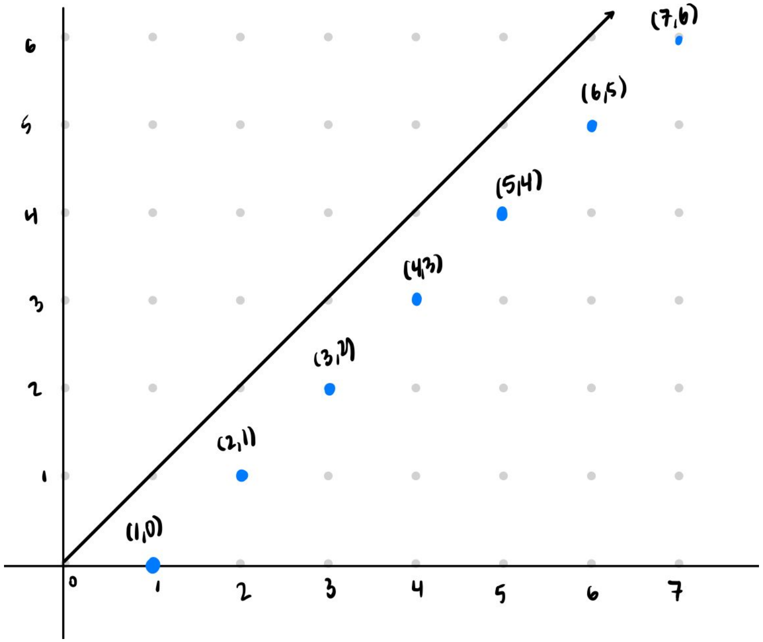


If all points on the line of y = x are in the relation then the relation is reflexive

The relation is not reflexive!

Problem 1a

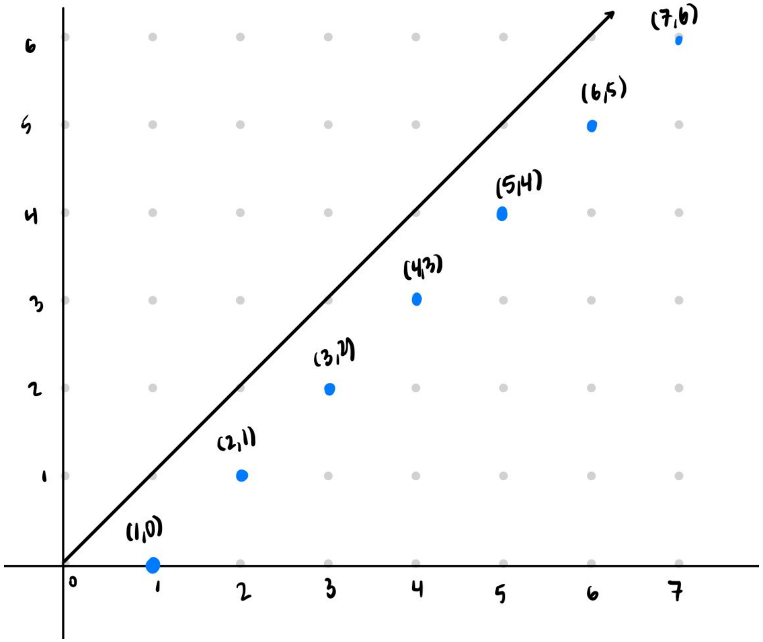
Let $R = \{(x, y) \ : \ x = y + 1\}$ on $\mathbb{N}$.



If all points that are reflected across y = x are also in the relation, then the relation is symmetric

Problem 1a

Let $R = \{(x, y) \ : \ x = y + 1\}$ on $\mathbb{N}$.

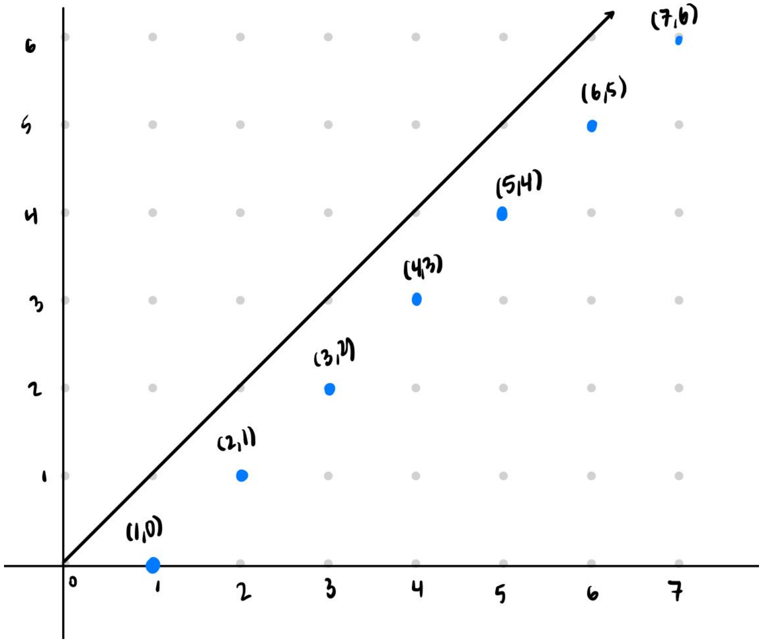


If all points that are reflected across y = x are also in the relation, then the relation is symmetric

The relation is not symmetric!

Problem 1a

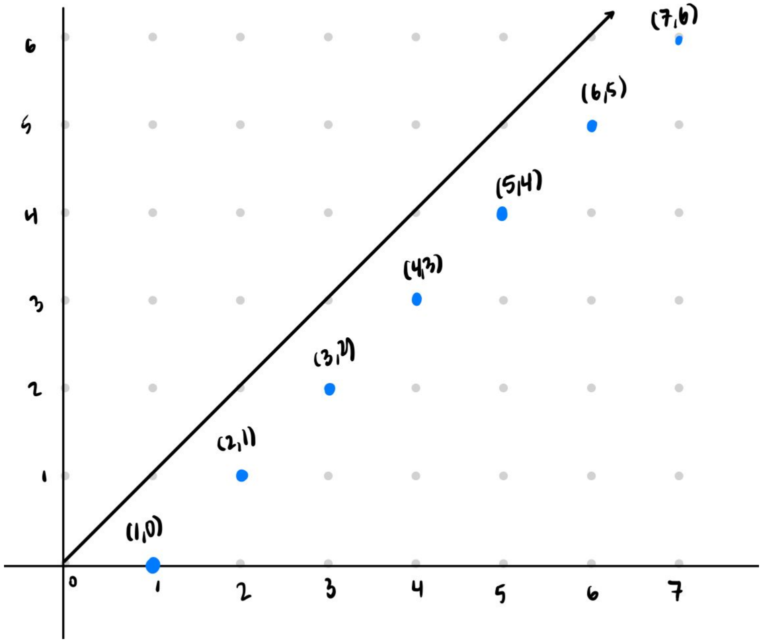
Let $R = \{(x, y)\ :\ x = y + 1\}$ on $\mathbb{N}$.



If all points that are reflected across y = x are not in the relation, then the relation is antisymmetric

Problem 1a

Let $R = \{(x, y) \ : \ x = y + 1\}$ on $\mathbb{N}$.



If all points that are reflected across y = x are not in the relation, then the relation is antisymmetric

The relation is antisymmetric!

Problem 1a

Let $R = \{(x, y) \ : \ x = y + 1\}$ on $\mathbb{N}$.

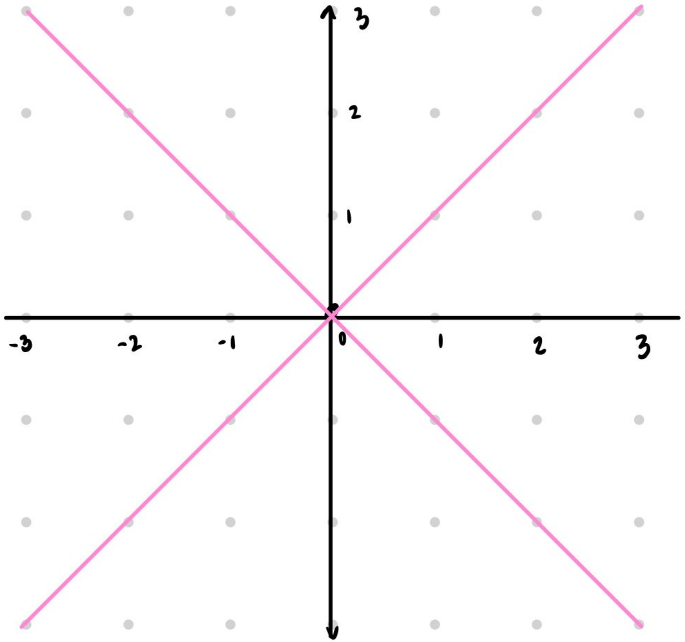
not reflexive (counterexample: $(1,1) \notin R$), not symmetric (counterexample: $(2,1) \in R$ but $(1,2) \notin R$), antisymmetric, not transitive (counterexample: $(3,2) \in R$ and $(2,1) \in R$, but $(3,1) \notin R$

Problem 1b

Let $R = \{(x, y) \ : \ x^2 = y^2\}$ on $\mathbb{R}$.

Problem 1b

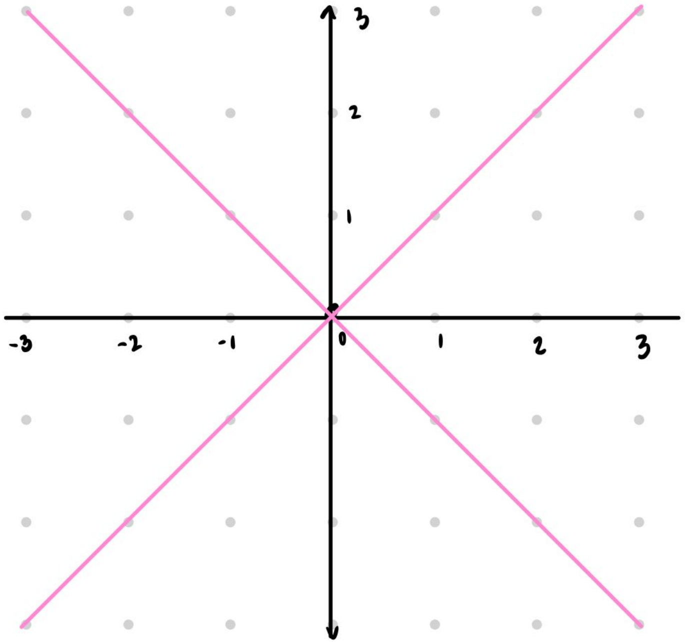
Let $R = \{(x, y) \ : \ x^2 = y^2\}$ on $\mathbb{R}$.



We can graph the points of R

Problem 1b

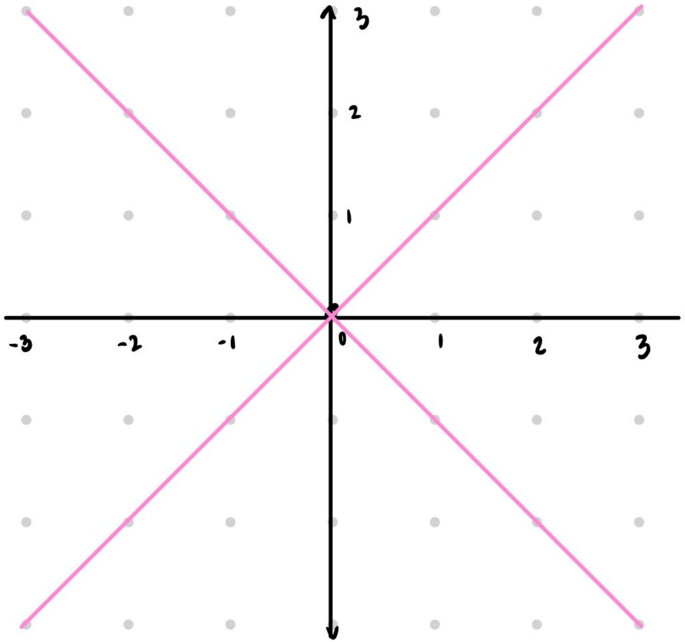
Let $R = \{(x, y) \ : \ x^2 = y^2\}$ on $\mathbb{R}$.



If all points on the line of y = x are in the relation then the relation is reflexive

Problem 1b

Let $R = \{(x, y) \ : \ x^2 = y^2\}$ on $\mathbb{R}$.

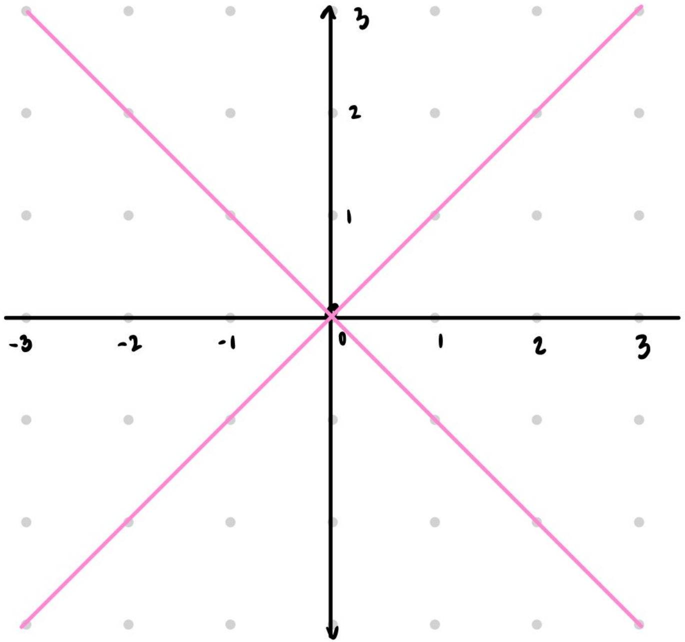


If all points on the line of y = x are in the relation then the relation is reflexive

The relation is reflexive!

Problem 1b

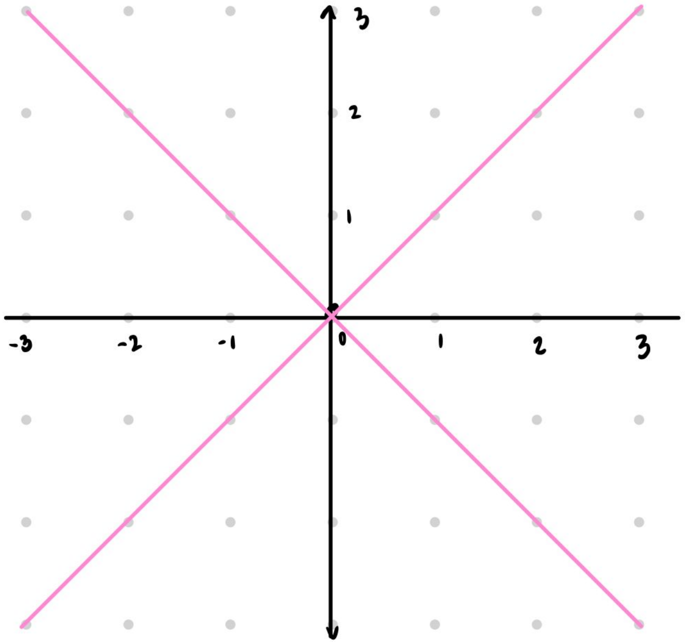
Let $R = \{(x, y)\ :\ x^2 = y^2\}$ on $\mathbb{R}$.



If all points that are reflected across y = x are also in the relation, then the relation is symmetric

Problem 1b

Let $R = \{(x, y) \ : \ x^2 = y^2\}$ on $\mathbb{R}$.

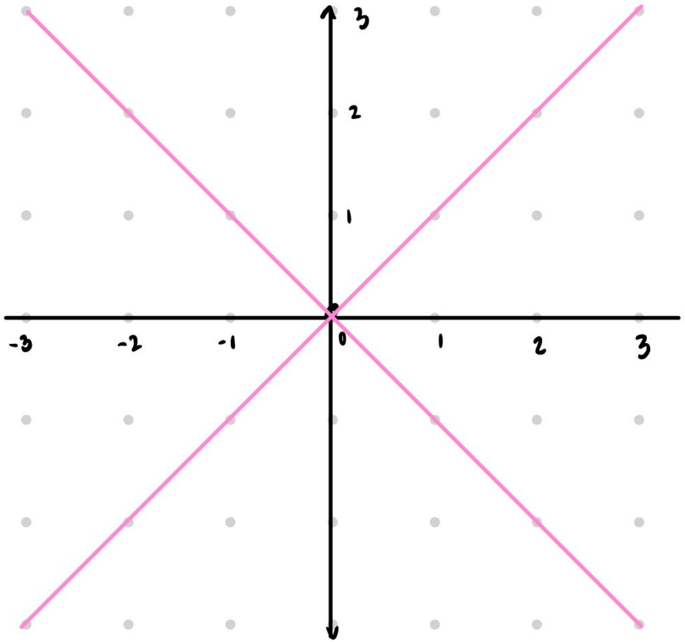


If all points that are reflected across y = x are also in the relation, then the relation is symmetric

The relation is symmetric!

Problem 1b

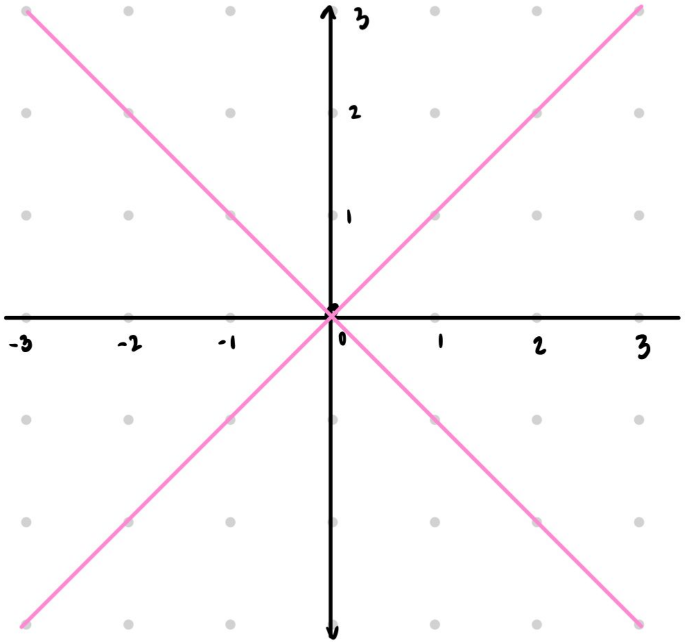
Let $R = \{(x, y) \ : \ x^2 = y^2\}$ on $\mathbb{R}$.



If all points that are reflected across y = x are not in the relation, then the relation is antisymmetric

Problem 1b

Let $R = \{(x, y) : x^2 = y^2\}$ on $\mathbb{R}$.



If all points that are reflected across y = x are not in the relation, then the relation is antisymmetric

The relation is not antisymmetric!

Problem 1b

Let $R = \{(x, y) \ : \ x^2 = y^2\}$ on $\mathbb{R}$.

reflexive, symmetric, not antisymmetric (counterexample: $(-2, 2) \in R$ and $(2, -2) \in R$ but $2 \neq -2$), transitive

Proving Relations!

Problem 3c

Let $\Sigma = \{0,1\}$. Define the relation R on Σ^* by $(x,y) \in R$ if and only if len(xy) is even. (Here xy is notation for the concatenation of the two strings x and y and len refers to the length of the string.)

Hint: In your proofs below, you may use the fact from lecture that len$(xy) =$ len$(x) +$ len(y).

Prove that R is transitive.

Problem 3c

Let $\Sigma = \{0, 1\}$. Define the relation R on Σ^* by $(x, y) \in R$ if and only if len(xy) is even. (Here xy is notation for the concatenation of the two strings x and y and len refers to the length of the string.)

Hint: In your proofs below, you may use the fact from lecture that len$(xy) =$ len$(x) +$ len(y).

Prove that R is transitive.

Formally the claim is: $\forall a, b, c \in \Sigma^*[((a, b) \in R \wedge (b, c) \in R) \rightarrow (a, c) \in R]$

Problem 3c

Prove that R is transitive.

Let a, b, and c be arbitrary elements of Σ^*. Suppose $(a, b) \in R$ and $(b, c) \in R$.

Problem 3c

Prove that R is transitive.

Let a, b, and c be arbitrary elements of Σ^*. Suppose $(a, b) \in R$ and $(b, c) \in R$. Then

$$\text{len}(ac) = \text{len}(a) + \text{len}(c)$$

Problem 3c

Prove that R is transitive.

Let a, b, and c be arbitrary elements of Σ^*. Suppose $(a, b) \in R$ and $(b, c) \in R$. Then

$$\begin{aligned} \mathsf{len}(ac) &= \mathsf{len}(a) + \mathsf{len}(c) \\ &= (\mathsf{len}(a) + \mathsf{len}(b)) + (\mathsf{len}(b) + \mathsf{len}(c)) - 2\mathsf{len}(b) \end{aligned}$$

Problem 3c

Prove that R is transitive.

Let a, b, and c be arbitrary elements of Σ^*. Suppose $(a, b) \in R$ and $(b, c) \in R$.
Then

$$\begin{aligned}
\text{len}(ac) &= \text{len}(a) + \text{len}(c) \\
&= (\text{len}(a) + \text{len}(b)) + (\text{len}(b) + \text{len}(c)) - 2\text{len}(b) \\
&= \text{len}(ab) + \text{len}(bc) - 2\text{len}(b)
\end{aligned}$$

Problem 3c

Prove that R is transitive.

Let a, b, and c be arbitrary elements of Σ^*. Suppose $(a, b) \in R$ and $(b, c) \in R$.
Then

$$\begin{aligned} \text{len}(ac) &= \text{len}(a) + \text{len}(c) \\ &= (\text{len}(a) + \text{len}(b)) + (\text{len}(b) + \text{len}(c)) - 2\text{len}(b) \\ &= \text{len}(ab) + \text{len}(bc) - 2\text{len}(b) \end{aligned}$$

Thus $\text{len}(ac)$ is the sum of three terms: $\text{len}(ab)$, $\text{len}(bc)$, and $-2\text{len}(b)$. By definition of R, we have that $\text{len}(ab)$ is even and $\text{len}(bc)$ is even. The first two terms are even by assumption, and the third is even by definition of even. Since, the sum of even numbers is even, so it follows that $\text{len}(ac)$ is even. By definition of R, it follows that $(a, c) \in R$. Since a, b, and c were arbitrary, it follows by definition of transitivity that R is transitive.

Deterministic Finite Automata

Deterministic Finite Automata

- A DFA is a finite-state machine that accepts or rejects a given string of symbols, by running through a state sequence uniquely determined by the string.

- In other words:
  - Our machine is going to get a string as input. It will read one character at a time and update "its state."
  - At every step, the machine thinks of itself as in one of the (finite number) vertices. When it reads the character, it follows the arrow labeled with that character to its next state.
  - Start at the "start state" (unlabeled, incoming arrow).
  - After you've read the last character, accept the string if and only if you're in a "final state" (double circle).

- There is exactly one start state; can have as many accept states (aka final states) as you want – including none.

Problem 4 – DFAs, Stage 1

Construct DFAs to recognize each of the following languages.
Let $\Sigma = \{0, 1, 2, 3\}$.

b) All strings whose digits sum to an even number.

Work on this problem with the people around you.

Problem 4 – DFAs, Stage 1

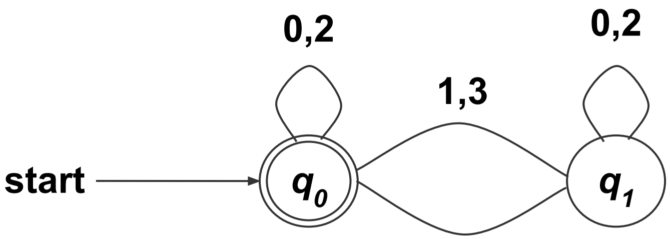
Let $\Sigma = \{0, 1, 2, 3\}$.

b) All strings whose digits sum to an even number.

Problem 4 – DFAs, Stage 1

Let $\Sigma = \{0, 1, 2, 3\}$.

b) All strings whose digits sum to an even number.



q_0: strings whose sum of digits is even
q_1: strings whose sum of digits is odd

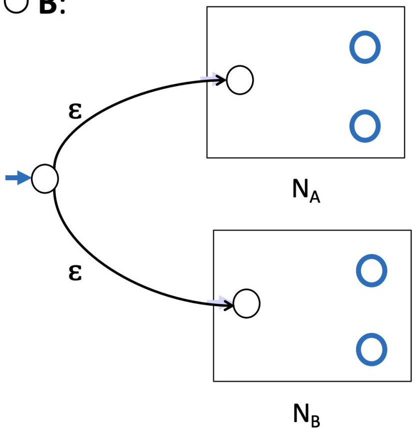
Nondeterministic Finite Automata

Nondeterministic Finite Automata

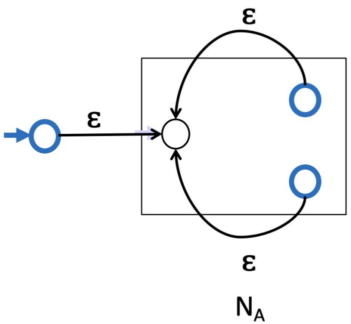
- Similar to DFAs, but with less restrictions.
  - From a given state, we'll allow any number of outgoing edges labeled with a given character. (In a DFA, we have only 1 outgoing edge labeled with each character).
  - The machine can follow any of them.
  - We'll have edges labeled with "ε" – the machine (optionally) can follow one of those without reading another character from the input.
  - If we "get stuck" i.e. the next character is a and there's no transition leaving our state labeled a, the computation dies.
- An NFA still has exactly one start state and any number of final states.
- The NFA accepts x if there is some path from a start state to a final state labeled with x.
- From a state, you can have 0,1, or many outgoing arrows labeled with a single character. You can choose any of them to build the required path.

Regex to NFA Conversion!

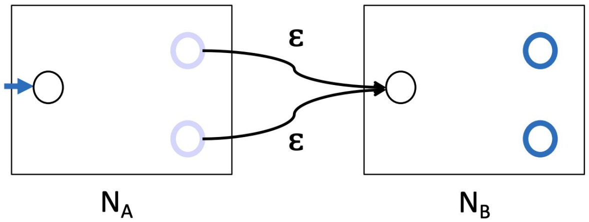
Case A ∪ B:



Case A*

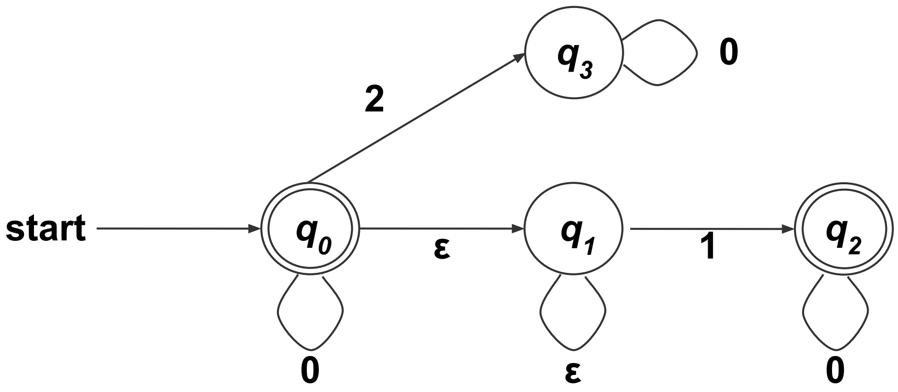


Case AB:



Task 6 – NFAs

a) What language does the following NFA accept?

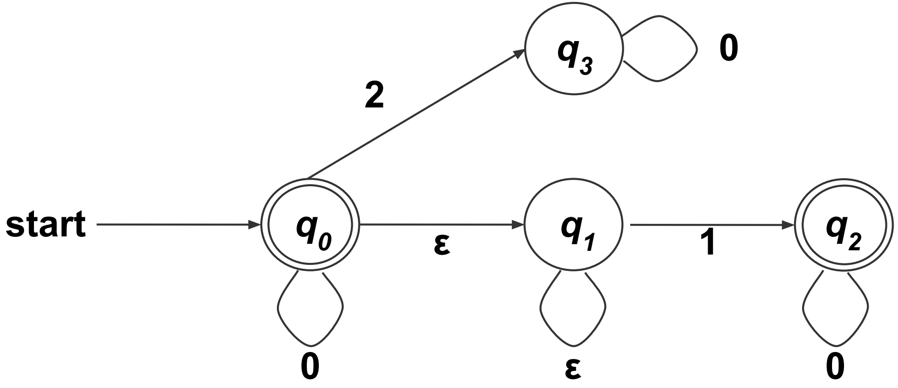


b) Create an NFA for the language “all binary strings that have a 1 as one of the last three digits”.

Work on this problem with the people around you.

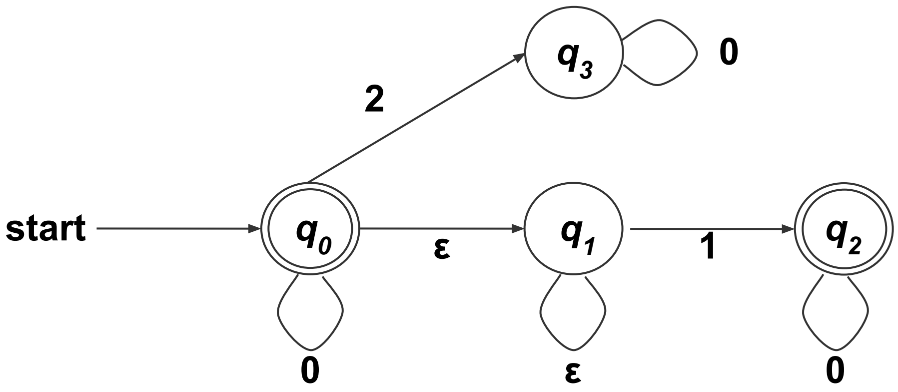
Task 6 – NFAs

a) What language does the following NFA accept?



Task 6 – NFAs

a) What language does the following NFA accept?



All strings of only 0’s and 1’s, not containing more than one 1.

Task 6 – NFAs

b) Create an NFA for the language “all binary strings that have a 1 as one of the last three digits”.

Task 6 – NFAs

b) Create an NFA for the language “all binary strings that have a 1 as one of the last three digits”.

Consider generating regex for:

All binary strings that have a 1 at the end

All binary strings that have a 1 at the second position from the end

All binary strings that have a 1 at the third position from the end

Task 6 – NFAs

b) Create an NFA for the language “all binary strings that have a 1 as one of the last three digits”.

Consider generating regex for:

All binary strings that have a 1 at the end

(0 U 1)* **1**

All binary strings that have a 1 at the second position from the end

All binary strings that have a 1 at the third position from the end

Task 6 – NFAs

b) Create an NFA for the language "all binary strings that have a 1 as one of the last three digits".

Consider generating regex for:

All binary strings that have a 1 at the end

(0 U 1)* **1**

All binary strings that have a 1 at the second position from the end

(0 U 1)* **1** (0 U 1)

All binary strings that have a 1 at the third position from the end

Task 6 – NFAs

b) Create an NFA for the language “all binary strings that have a 1 as one of the last three digits”.

Consider generating regex for:

All binary strings that have a 1 at the end
(0 U 1)* **1**

All binary strings that have a 1 at the second position from the end
(0 U 1)* **1** (0 U 1)

All binary strings that have a 1 at the third position from the end
(0 U 1)* **1** (0 U 1) (0 U 1)

Task 6 – NFAs

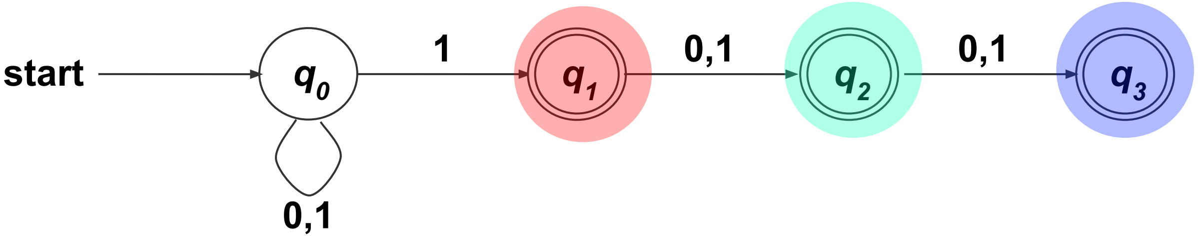
b) Create an NFA for the language “all binary strings that have a 1 as one of the last three digits”.

How can we use this to construct the NFA?

Task 6 – NFAs

b) Create an NFA for the language "all binary strings that have a 1 as one of the last three digits".

How can we use this to construct the NFA?



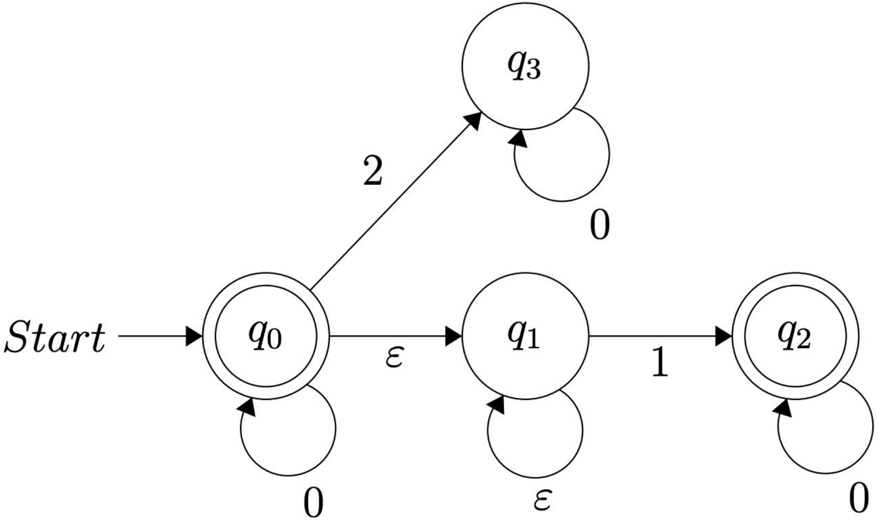
NFA to DFA

DFAs and NFAs

- Any language that can be accepted by a DFA can be accepted by an NFA
- A DFA is an NFA!
- Every language that can be accepted by an NFA can be accepted by a DFA also!
- Using the following algorithm:
    - Start at the set of states which can be reached from the empty state
    - Construct a table with all symbols of the Σ, and what each symbol takes the set of states to
    - Keep going until there are no more unique states

Task 7

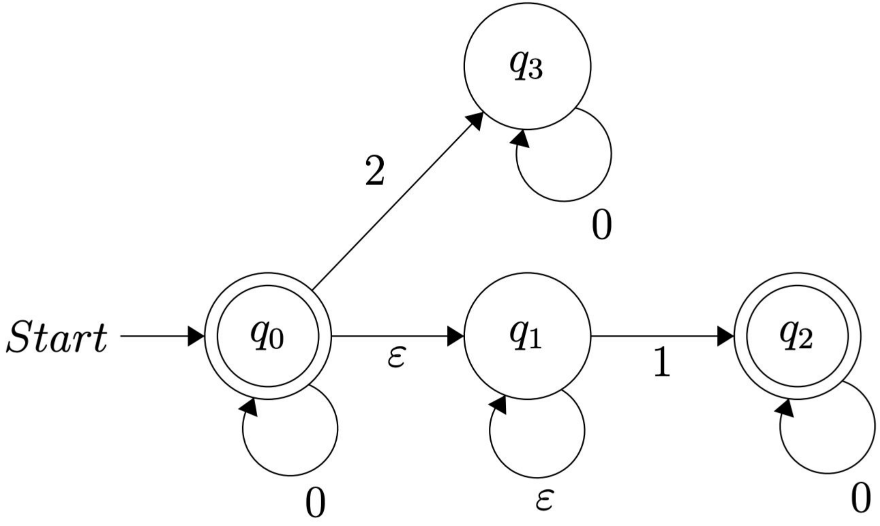
Convert the following NFA to a DFA of the same language



Task 7 - Documenting states in a table

Where can we get from the start state (q0) without reading input?

(DFA) state	0-transition	1-transition	2-transition

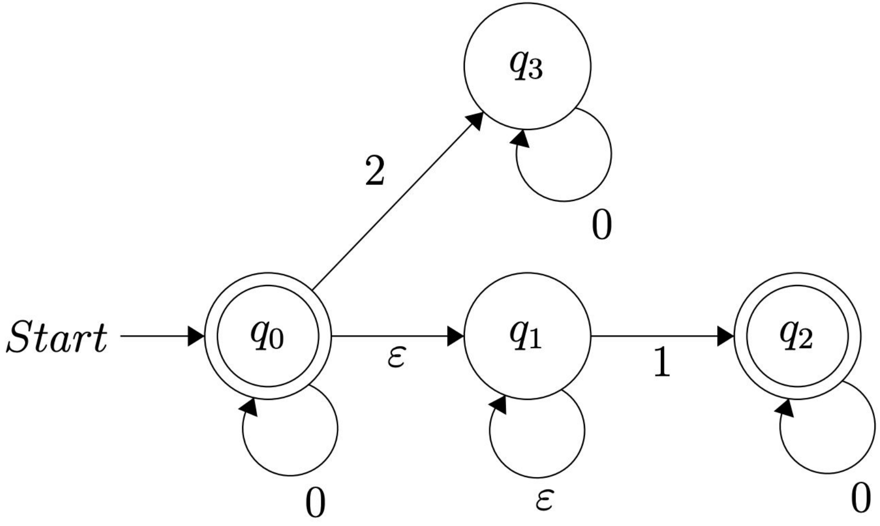


Task 7 - Documenting states in a table

Where can we get from the start state (q0) without reading input?

- q0
- take ε to q1

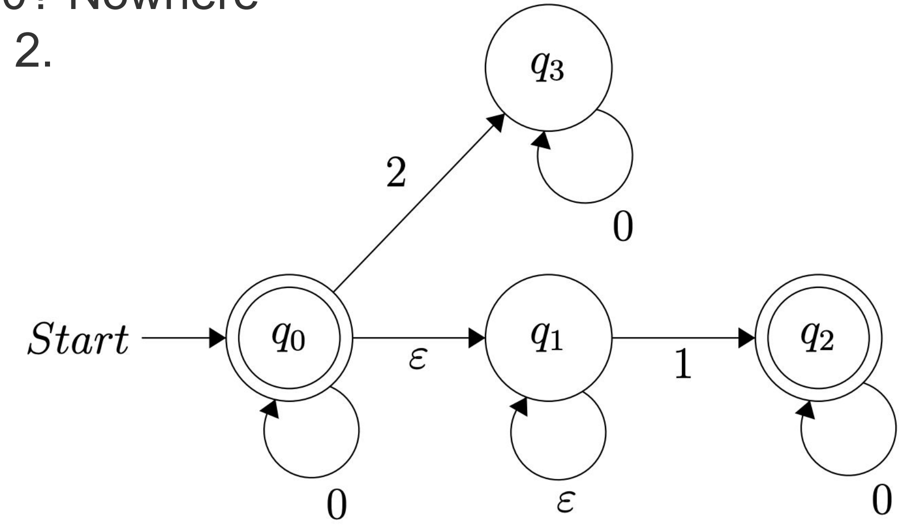
(DFA) state	0-transition	1-transition	2-transition
q0, q1			



Task 7 - Documenting states in a table

From q0, where can we get reading in 0? q0 and q1
From q1, where can we get reading in 0? Nowhere
Repeat for reading in 1, and reading in 2.
Then fill in the state table.

(DFA) state	0-transition	1-transition	2-transition
q0, q1	q0, q1	q2	q3

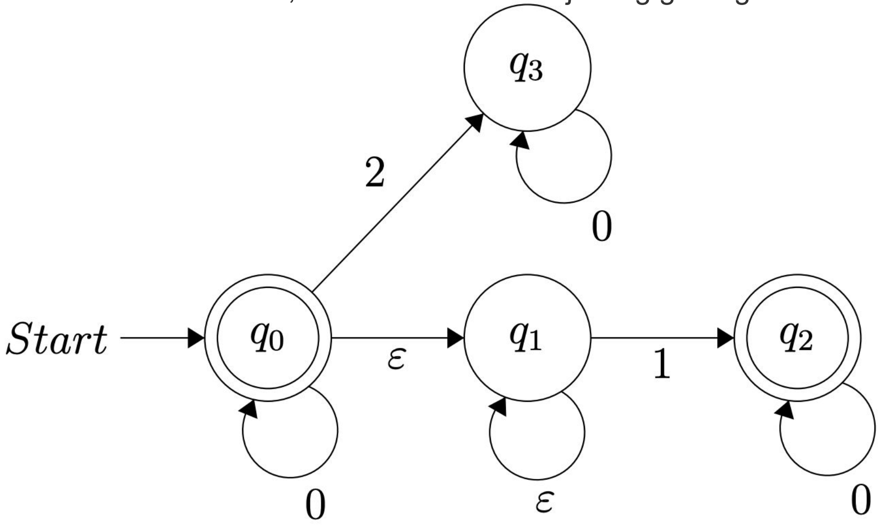


Task 7 - Documenting states in a table

We need to define the other states (q2, q3) in the table now.

- When there's no defined transition for an input, the computation dies in the NFA, so we send it to a rejecting garbage state ∅.

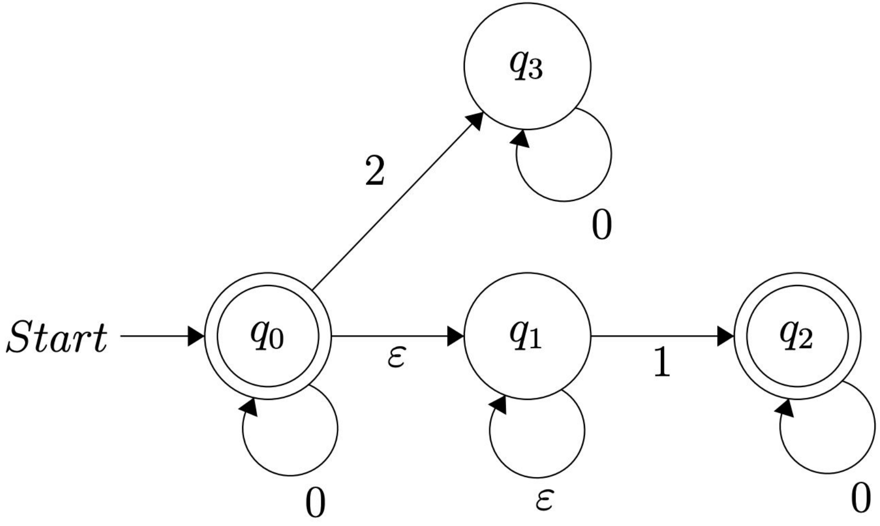
(DFA) state	0-transition	1-transition	2-transition
q0, q1	q0, q1	q2	q3
q2	q2	∅	∅
q3			



Task 7 - Documenting states in a table

We need to define the other states (q2 and q3) in the table now.

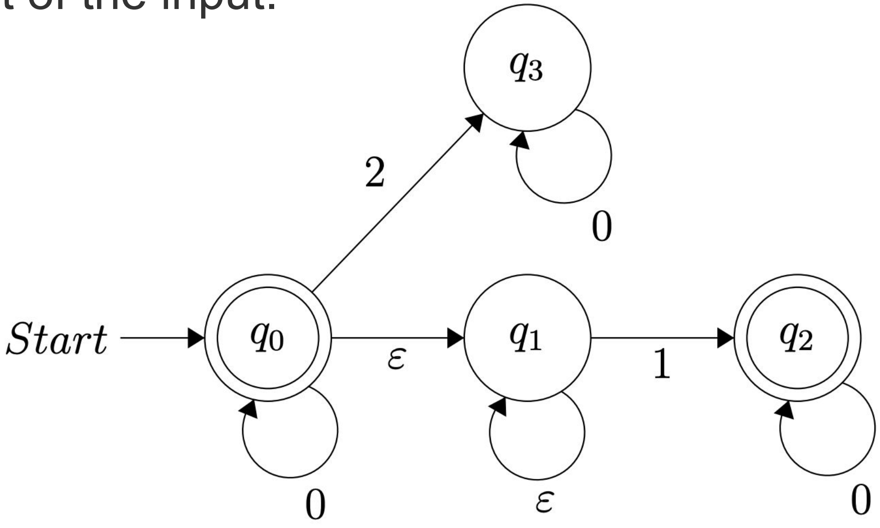
(DFA) state	0-transition	1-transition	2-transition
q0, q1	q0, q1	q2	q3
q2	q2	∅	∅
q3	q3	∅	∅



Task 7 - Documenting states in a table

The only state we have left to define is the “garbage” state. This one always is rejecting regardless of the rest of the input.

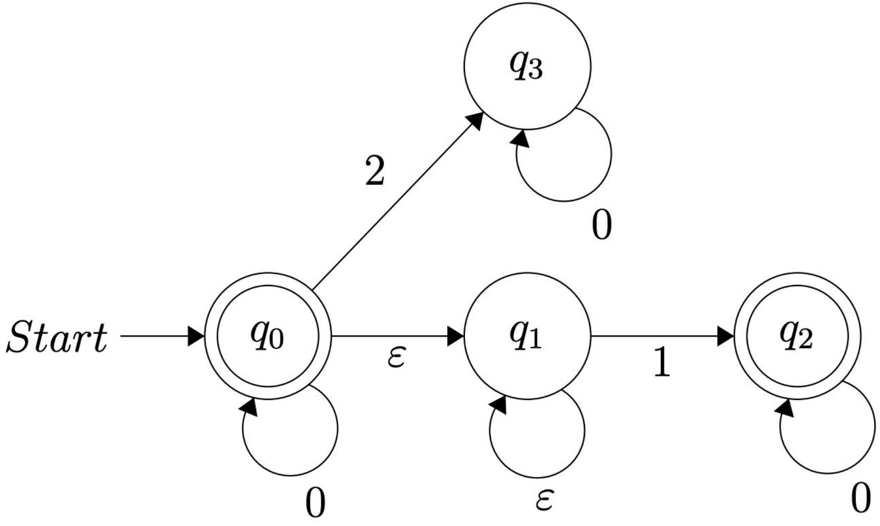
(DFA) state	0-transition	1-transition	2-transition
q0, q1	q0, q1	q2	q3
q2	q2	∅	∅
q3	q3	∅	∅
∅	∅	∅	∅



Task 7 - Documenting states in a table

Which states are accepting? (The rest are rejecting)

(DFA) state	0-transition	1-transition	2-transition
q0, q1	q0, q1	q2	q3
q2	q2	∅	∅
q3	q3	∅	∅
∅	∅	∅	∅

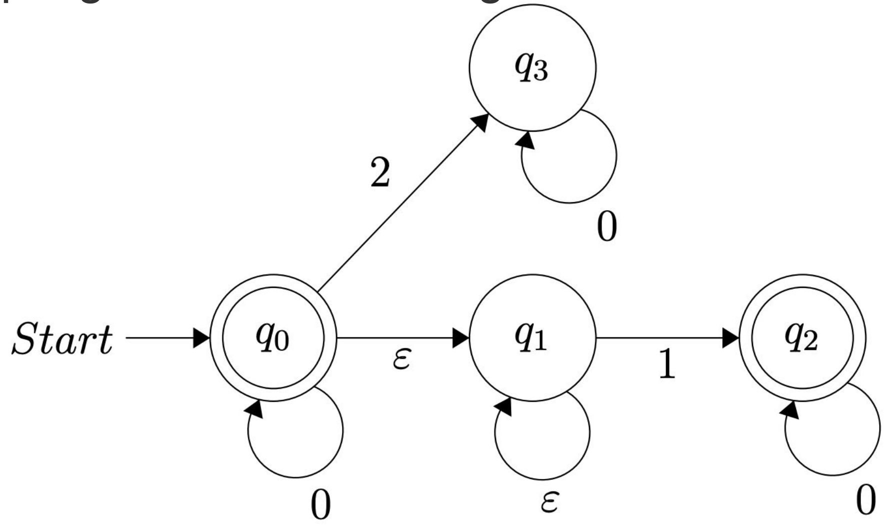


Task 7 - Documenting states in a table

Which states are accepting? (The rest are rejecting)

- All DFA states that include an accepting state from the original NFA.

(DFA) state	0-transition	1-transition	2-transition
q0, q1	q0, q1	q2	q3
q2	q2	∅	∅
q3	q3	∅	∅
∅	∅	∅	∅

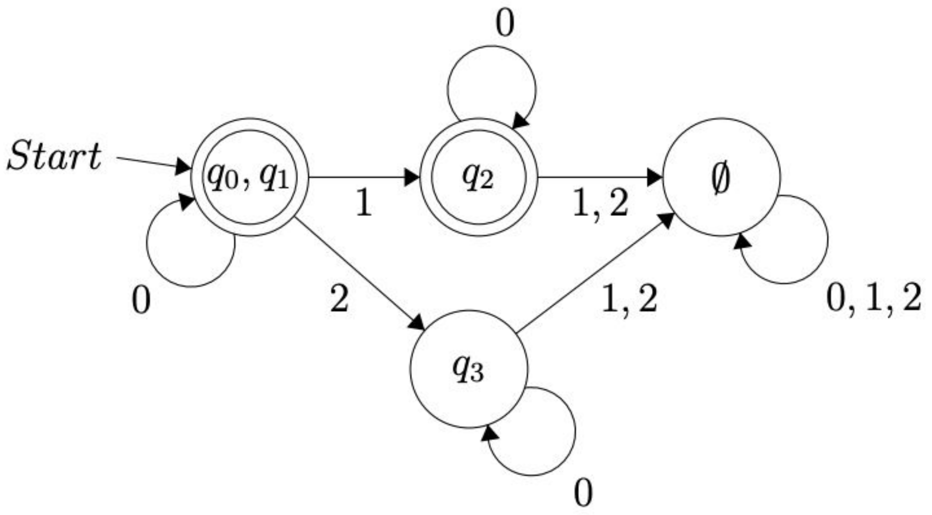


Now try drawing out the DFA using the state table.

- Don’t forget a start state!

Task 7

(DFA) state	0-transition	1-transition	2-transition
q0, q1	q0, q1	q2	q3
q2	q2	∅	∅
q3	q3	∅	∅
∅	∅	∅	∅



That's All, Folks!

Thanks for coming to section this week!
Any questions?

By: Aruna & Alysa