CSE 311 Section 08

Regular Expressions, CFGs, Relations

Administrivia

Announcements & Reminders

- Midterm grades will be released shortly
- Homework 6 was due Wednesday (2/26)
- Homework 7 will be due Wednesday (3/5)
- Check your section participation grade on gradescope
    - If it different than what you expect, let your TA know

Regular Expressions

Regular Expressions

ε matches only the **empty string**

a matches only the one-character string *a*

A ∪ B matches all strings that either **A** matches or **B** matches (or both)

AB matches all strings that have a first part that **A** matches followed by a second part that **B** matches

A* matches all strings that have any number of strings (even 0) that **A** matches, one after another (**ε ∪ A ∪ AA ∪ AAA ∪ …**)

Definition of the *language* matched by a regular expression

Task 1 – Regular Expressions

a) Write a regular expression that matches base 10 numbers (e.g., there should be no leading zeroes).

b) Write a regular expression that matches all base-3 numbers that are divisible by 3.

c) Write a regular expression that matches all binary strings that contain the substring "111", but not the substring "000".

We will do (b) together, then work on c

Task 1 – Regular Expressions

a) Write a regular expression that matches base 10 numbers (e.g., there should be no leading zeroes).

base-10 numbers:

Our everyday numbers! Notice we have 10 symbols (0-9) to represent numbers.

256: $(\mathbf{2} * 10^2) + (\mathbf{5} * 10^1) + (\mathbf{6} * 10^0)$

base-2 numbers: (binary)

10: $(\mathbf{1} * 2^1) + (\mathbf{0} * 2^0)$

Task 1 – Regular Expressions

a) Write a regular expression that matches base 10 numbers (e.g., there should be no leading zeroes).

Representing numbers all possible *strings* using numbers 0-9:

Task 1 – Regular Expressions

a) Write a regular expression that matches base 10 numbers (e.g., there should be no leading zeroes).

Representing numbers all possible *strings* using numbers 0-9:

$$(0 \cup 1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)^*$$

Task 1 – Regular Expressions

a) Write a regular expression that matches base 10 numbers (e.g., there should be no leading zeroes).

Representing numbers all possible *strings* using numbers 0-9:

$(0 \cup 1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)^*$

⚠️ "<u>0</u>101" or "<u>0</u>91" **are not** Base-10 numbers

Task 1 – Regular Expressions

a) Write a regular expression that matches base 10 numbers (e.g., there should be no leading zeroes).

Representing numbers all possible *strings* using numbers 0-9:

$(0 \cup 1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)^*$

⚠️ "0101" or "091" **are not** Base-10 numbers

All possible *strings* using numbers 0-9 that never start with 0

Task 1 – Regular Expressions

a) Write a regular expression that matches base 10 numbers (e.g., there should be no leading zeroes).

Representing numbers all possible *strings* using numbers 0-9:

$(0 \cup 1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)^*$

⚠️ "<u>0</u>101" or "<u>0</u>91" **are not** Base-10 numbers

All possible *strings* using numbers 0-9 that never start with 0

$(1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)(0 \cup 1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)^*$

Task 1 – Regular Expressions

a) Write a regular expression that matches base 10 numbers (e.g., there should be no leading zeroes).

Representing numbers all possible *strings* using numbers 0-9:

$$(0 \cup 1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)^*$$

⚠️ “0101” or “091” **are not** Base-10 numbers

All possible *strings* using numbers 0-9 that never start with 0

$$(1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)(0 \cup 1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)^*$$

⚠️ “0” **is** a Base-10 number not considered

Task 1 – Regular Expressions

a) Write a regular expression that matches base 10 numbers (e.g., there should be no leading zeroes).

Representing numbers all possible *strings* using numbers 0-9:

$(0 \cup 1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)^*$

⚠️ "<u>0</u>101" or "<u>0</u>91" **are not** Base-10 numbers

All possible *strings* using numbers 0-9 that never start with 0

$(1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)(0 \cup 1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)^*$

⚠️ "<u>0</u>" **is** a Base-10 number not considered

All possible *strings* using numbers 0-9 that never start with 0 *or* is 0

Task 1 – Regular Expressions

a) Write a regular expression that matches base 10 numbers (e.g., there should be no leading zeroes).

Representing numbers all possible *strings* using numbers 0-9:

(0 U 1 U 2 U 3 U 4 U 5 U 6 U 7 U 8 U 9)*

⚠️ “0101” or “091” **are not** Base-10 numbers

All possible *strings* using numbers 0-9 that never start with 0

(1 U 2 U 3 U 4 U 5 U 6 U 7 U 8 U 9)(0 U 1 U 2 U 3 U 4 U 5 U 6 U 7 U 8 U 9)*

⚠️ “0” **is** a Base-10 number not considered

All possible *strings* using numbers 0-9 that never start with 0 *or* is 0

0 U ((1 U 2 U 3 U 4 U 5 U 6 U 7 U 8 U 9)(0 U 1 U 2 U 3 U 4 U 5 U 6 U 7 U 8 U 9)*)

Task 1 – Regular Expressions

a) Write a regular expression that matches base 10 numbers (e.g., there should be no leading zeroes).

Representing numbers all possible *strings* using numbers 0-9:

(0 U 1 U 2 U 3 U 4 U 5 U 6 U 7 U 8 U 9)*

⚠️ "0101" or "091" **are not** Base-10 numbers

All possible *strings* using numbers 0-9 that never start with 0

(1 U 2 U 3 U 4 U 5 U 6 U 7 U 8 U 9)(0 U 1 U 2 U 3 U 4 U 5 U 6 U 7 U 8 U 9)*

⚠️ "0" **is** a Base-10 number not considered

All possible *strings* using numbers 0-9 that never start with 0 *or* is 0

0 U ((1 U 2 U 3 U 4 U 5 U 6 U 7 U 8 U 9)(0 U 1 U 2 U 3 U 4 U 5 U 6 U 7 U 8 U 9)*)

✅Generates only all possible Base-10 numbers

Task 1 – Regular Expressions

b) Write a regular expression that matches all base-3 numbers that are divisible by 3.

Task 1 – Regular Expressions

b) Write a regular expression that matches all base-3 numbers that are divisible by 3.

Write a regular expression that matches all base-3 numbers

Task 1 – Regular Expressions

b) Write a regular expression that matches all base-3 numbers that are divisible by 3.

Write a regular expression that matches all base-3 numbers

$$0 \cup ((1 \cup 2)(0 \cup 1 \cup 2)*)$$

✅Generates only all possible Base-3 numbers

Task 1 – Regular Expressions

b) Write a regular expression that matches all base-3 numbers that are divisible by 3.

Write a regular expression that matches all base-3 numbers

$$0 \cup ((1 \cup 2)(0 \cup 1 \cup 2)*)$$

Generates only all possible Base-3 numbers

…divisible by 3

Task 1 – Regular Expressions

b) Write a regular expression that matches all base-3 numbers that are divisible by 3.

Write a regular expression that matches all base-3 numbers

$0 \cup ((1 \cup 2)(0 \cup 1 \cup 2)*)$

Generates only all possible Base-3 numbers

…divisible by 3

Hint: you know that Base-<u>10</u> numbers are divisible by <u>10</u> when <u>they end in 0</u> (10, 20, 30, 40…)

Task 1 – Regular Expressions

b) Write a regular expression that matches all base-3 numbers that are divisible by 3.

Write a regular expression that matches all base-3 numbers

$0 \cup ((1 \cup 2)(0 \cup 1 \cup 2)*)$

Generates only all possible Base-3 numbers

…divisible by 3

Hint: you know that Base-10 numbers are divisible by 10 when <u>they end in 0</u> (10, 20, 30, 40…)

$0 \cup ((1 \cup 2)(0 \cup 1 \cup 2)*\mathbf{0})$

✅all possible Base-3 numbers divisible by 3

Task 1 – Regular Expressions

c) Write a regular expression that matches all binary strings that contain the substring “111”, but not the substring “000”.

Task 1 – Regular Expressions

c) Write a regular expression that matches all binary strings that contain the substring “111”, but not the substring “000”.

all binary strings that contain the substring “111”

Task 1 – Regular Expressions

c) Write a regular expression that matches all binary strings that contain the substring “111”, but not the substring “000”.

all binary strings that contain the substring “111”

$(0 \cup 1)^* \, 111 \, (0 \cup 1)^*$

⚠️ The Kleene-star has us generating any number of 0’s

Task 1 – Regular Expressions

c) Write a regular expression that matches all binary strings that contain the substring “111”, but not the substring “000”.

all binary strings that contain the substring “111”

$(0 \cup 1)^*\ 111\ (0 \cup 1)^*$

⚠️ The Kleene-star has us generating any number of 0’s

…without the substring “000”

Use careful case-work to modify this and produce only 0,1,or two 0’s

Task 1 – Regular Expressions

c) Write a regular expression that matches all binary strings that contain the substring "111", but not the substring "000".

all binary strings that contain the substring "111"

$(0 \cup 1)^* \, 111 \, (0 \cup 1)^*$

⚠️ The Kleene-star has us generating any number of 0's

…without the substring "000"

Use careful case-work to modify this and produce only 0,1,or two 0's

$(0 \cup 00 \cup \varepsilon) \, (1)^* \, 111 \, (0 \cup 00 \cup \varepsilon) \, (1)^*$

Task 1 – Regular Expressions

c) Write a regular expression that matches all binary strings that contain the substring "111", but not the substring "000".

all binary strings that contain the substring "111"

$(0 \cup 1)^* \, 111 \, (0 \cup 1)^*$

⚠️ The Kleene-star has us generating any number of 0's

…without the substring "000"

Use careful case-work to modify this and produce only 0,1,or two 0's

$(0 \cup 00 \cup \varepsilon)\,(1)^*\,111\,(0 \cup 00 \cup \varepsilon)\,(1)^*$

⚠️ Cannot produce 1's with "0" or "00" like "1011101"

Task 1 – Regular Expressions

c) Write a regular expression that matches all binary strings that contain the substring “111”, but not the substring “000”.

all binary strings that contain the substring “111”

(0 ∪ 1)* 111 (0 ∪ 1)*

⚠️ The Kleene-star has us generating any number of 0’s

…without the substring “000”

Use careful case-work to modify this and produce only 0,1,or two 0’s

(0 ∪ 00 ∪ ε) (1)* 111 **(0 ∪ 00 ∪ ε)** (1)*

⚠️ Cannot produce 1’s with “0” or “00” like “1011101”

(0 ∪ 00 ∪ ε) **(01 ∪ 001 ∪ 1)*** 111 (0 ∪ 00 ∪ ε) **(01 ∪ 001 ∪ 1)***

Task 1 – Regular Expressions

c) Write a regular expression that matches all binary strings that contain the substring "111", but not the substring "000".

all binary strings that contain the substring "111"

(0 ∪ 1)* 111 (0 ∪ 1)*

⚠️ The Kleene-star has us generating any number of 0's

…without the substring "000"

Use careful case-work to modify this and produce only 0,1,or two 0's

(0 ∪ 00 ∪ ε) (1)* 111 **(0 ∪ 00 ∪ ε)** (1)*

⚠️ Cannot produce 1's with "0" or "00" like "<u>1</u>011101<u>1</u>"

(0 ∪ 00 ∪ ε) **(01 ∪ 001 ∪ 1)*** 111 (0 ∪ 00 ∪ ε) **(01 ∪ 001 ∪ 1)***

⚠️ Generates "000" like "<u>00</u> <u>01</u> 111"

Task 1 – Regular Expressions

c) Write a regular expression that matches all binary strings that contain the substring “111”, but not the substring “000”.

all binary strings that contain the substring “111”

(0 ∪ 1)* 111 (0 ∪ 1)*

⚠️ The Kleene-star has us generating any number of 0’s

…without the substring “000”

Use careful case-work to modify this and produce only 0,1,or two 0’s

(0 ∪ 00 ∪ ε) (1)* 111 **(0 ∪ 00 ∪ ε)** (1)*

⚠️ Cannot produce 1’s with “0” or “00” like “<u>1</u>011101<u>1</u>”

(0 ∪ 00 ∪ ε) **(01 ∪ 001 ∪ 1)*** 111 (0 ∪ 00 ∪ ε) **(01 ∪ 001 ∪ 1)***

⚠️ Generates “000” like “<u>00</u> <u>01</u> 111”

(01 ∪ 001 ∪ 1)* (0 ∪ 00 ∪ ε) 111 (01 ∪ 001 ∪ 1)* (0 ∪ 00 ∪ ε)

✅ all binary strings with “111” and without “000”

Task 1 – Regular Expressions

c) Write a regular expression that matches all binary strings that contain the substring “111”, but not the substring “000”.

all binary strings that contain the substring “111”

(0 ∪ 1)* 111 (0 ∪ 1)*

⚠️ The Kleene-star has us generating any number of 0’s

…without the substring “000”

Use careful case-work to modify this and produce only 0,1,or two 0’s

(0 ∪ 00 ∪ ε) (1)* 111 **(0 ∪ 00 ∪ ε)** (1)*

⚠️ Cannot produce 1’s with “0” or “00” like “<u>1</u>011101”

(0 ∪ 00 ∪ ε) **(01 ∪ 001 ∪ 1)*** 111 (0 ∪ 00 ∪ ε) **(01 ∪ 001 ∪ 1)**

⚠️ Generates “000” like “<u>00</u> <u>01</u> 111”

(01 ∪ 001 ∪ 1)* (0 ∪ 00 ∪ ε) 111 (01 ∪ 001 ∪ 1)* (0 ∪ 00

✅ all binary strings with “111” and without “000”

(01 ∪ 001 ∪ 1)* (0 ∪ 00 ∪ ε) 111 (01 ∪ 001 ∪ 1)* (0 ∪ 00 ∪ ε)

Context-Free Grammars

Context-Free Grammars

A context free grammar (CFG) is a finite set of production rules over:

- An alphabet Σ of “terminal symbols”
- A finite set V of “nonterminal symbols”
- A start symbol (one of the elements of V) usually denoted S

Always think back to Regex!

- CFG to match RE $\mathbf{A} \cup \mathbf{B}$

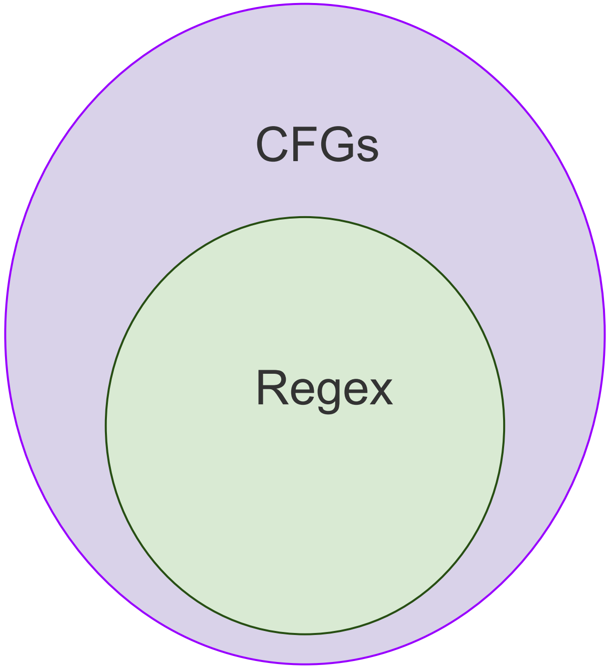
$\mathbf{S} \rightarrow \mathbf{S_1} \mid \mathbf{S_2}$ + rules from original CFGs

- CFG to match RE $\mathbf{AB}$

$\mathbf{S} \rightarrow \mathbf{S}_1 \mathbf{S}_2$ + rules from original CFGs

- CFG to match RE $\mathbf{A}^*$ $(= \varepsilon \cup \mathbf{A} \cup \mathbf{AA} \cup \mathbf{AAA} \cup \dots)$

$\mathbf{S} \rightarrow \mathbf{S_1} \mathbf{S} \mid \varepsilon$ + rules from CFG with $\mathbf{S_1}$



Always think back to Regex!

- CFG to match RE $\mathbf{A} \cup \mathbf{B}$

$\mathbf{S} \rightarrow \mathbf{S_1} \mid \mathbf{S_2}$ + rules from original CFGs

- CFG to match RE $\mathbf{AB}$

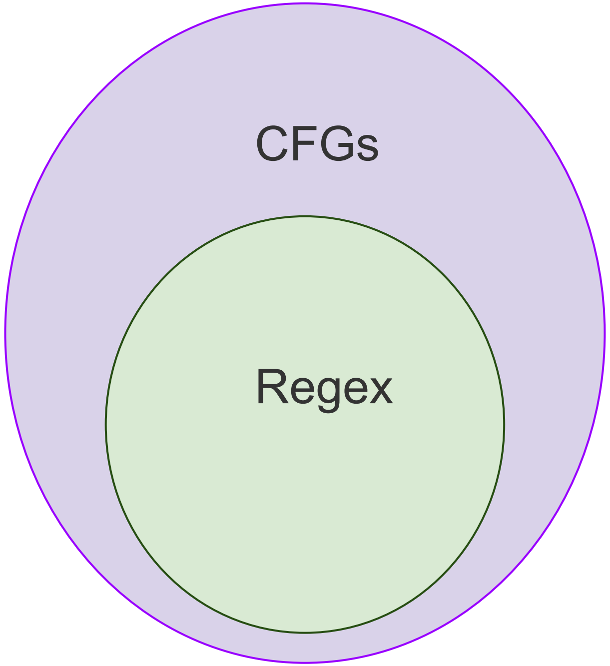
$\mathbf{S} \rightarrow \mathbf{S_1 S_2}$ + rules from original CFGs

- CFG to match RE $\mathbf{A}^*$ $(= \varepsilon \cup \mathbf{A} \cup \mathbf{AA} \cup \mathbf{AAA} \cup \ldots)$

$\mathbf{S} \rightarrow \mathbf{S_1 S} \mid \varepsilon$ + rules from CFG with $\mathbf{S_1}$

CFG or Regex?

"equal number of 0's and 1's" (ex. 011010)



Always think back to Regex!

- CFG to match RE **A ∪ B**

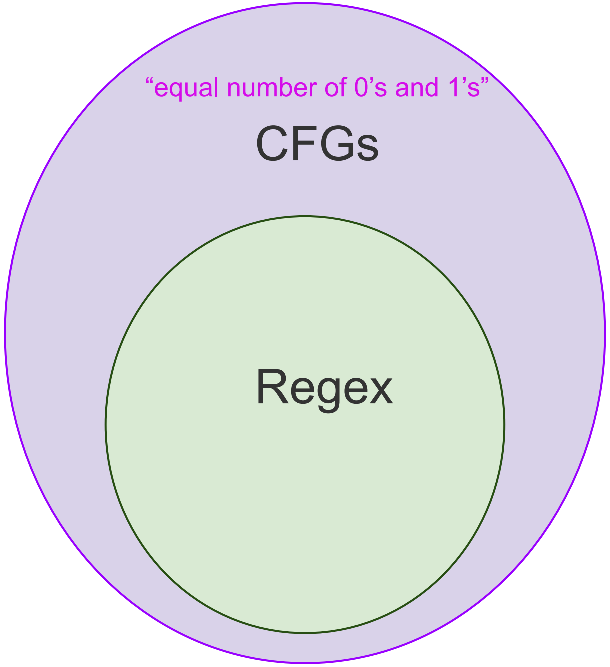
$\mathbf{S \rightarrow S_1 \mid S_2}$ + rules from original CFGs

- CFG to match RE **AB**

$\mathbf{S \rightarrow S_1 S_2}$ + rules from original CFGs

- CFG to match RE $\mathbf{A^*}$ (= $\varepsilon \cup \mathbf{A} \cup \mathbf{AA} \cup \mathbf{AAA} \cup \ldots$)

$\mathbf{S \rightarrow S_1 S} \mid \varepsilon$ + rules from CFG with $\mathbf{S_1}$



Task 2 – CFGs

Write a context-free grammar to match each of these languages.

a) All binary strings that start with 11.

b) All binary strings that contain at most one 1.

Work on this problem with the people around you.

Task 2 – CFGs

a) All binary strings that start with 11.

Task 2 – CFGs

a) All binary strings that start with 11.

Thinking back to regular expressions…

Task 2 – CFGs

a) All binary strings that start with 11.

Thinking back to regular expressions…

$11\ (0 \cup 1)^*$

Task 2 – CFGs

a) All binary strings that start with 11.

Thinking back to regular expressions…

$11\,(0 \cup 1)^*$

Now generate the CFG…

Task 2 – CFGs

a) All binary strings that start with 11.

Thinking back to regular expressions…

$11\ (0 \cup 1)^*$

Now generate the CFG…

$$
\begin{aligned}
\mathbf{S} &\rightarrow 11\mathbf{T} \\
\mathbf{T} &\rightarrow 1\mathbf{T} \mid 0\mathbf{T} \mid \varepsilon
\end{aligned}
$$

Task 2 – CFGs

b) All binary strings that contain at most one 1.

Task 2 – CFGs

b) All binary strings that contain at most one 1.

Thinking back to Regular expressions…

Task 2 – CFGs

b) All binary strings that contain at most one 1.

Thinking back to Regular expressions…

$0^* (1 \cup \varepsilon) 0^*$

Task 2 – CFGs

b) All binary strings that contain at most one 1.

Thinking back to Regular expressions…

$0^* \ (1 \cup \boldsymbol{\varepsilon}) \ 0^*$

Now generate the CFG…

Task 2 – CFGs

b) All binary strings that contain at most one 1.

Thinking back to Regular expressions…

$0^* \; (1 \cup \varepsilon) \; 0^*$

Now generate the CFG…

$$
\begin{aligned}
S &\rightarrow ABA \\
A &\rightarrow 0A \mid \varepsilon \\
B &\rightarrow 1 \mid \varepsilon
\end{aligned}
$$

Task 2 – CFGs

b) All binary strings that contain at most one 1.

Thinking back to Regular expressions…

$0^* \, (1 \cup \varepsilon) \, 0^*$

Now generate the CFG…

$$
\begin{aligned}
S &\rightarrow ABA \\
A &\rightarrow 0A \mid \varepsilon \\
B &\rightarrow 1 \mid \varepsilon
\end{aligned}
$$

Alternative solution:

$$S \rightarrow 0S \mid S0 \mid 1 \mid 0 \mid \varepsilon$$

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 $(b,a) \notin R$

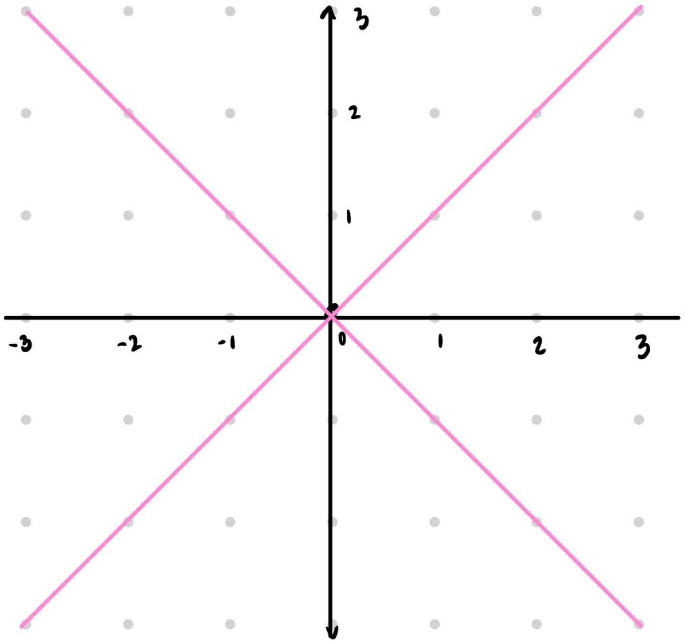
R is **transitive** iff $(a,b) \in R$ and $(b, c) \in R$ implies $(a, c) \in R$

Task 4b

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

Task 4b

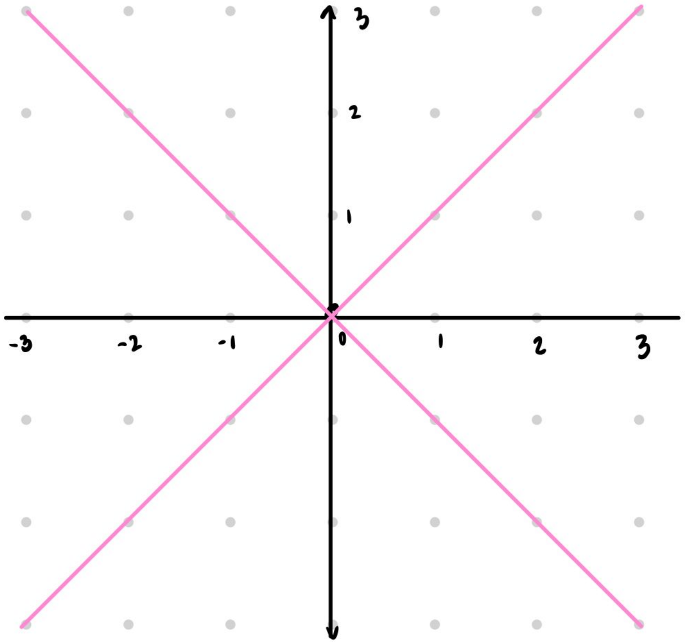
Let $R = \{(x, y) \ : \ x^2 = y^2\}$ on $\mathbb{R}$.



We can graph the points of R

Task 4b

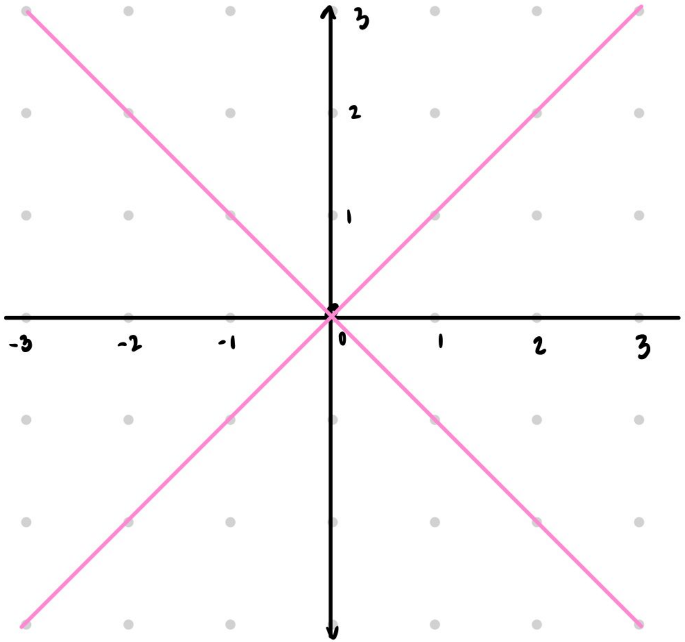
Let $R = \{(x, y) \ : \ x^2 = y^2\}$ on $\mathbb{R}$.



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

Task 4b

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

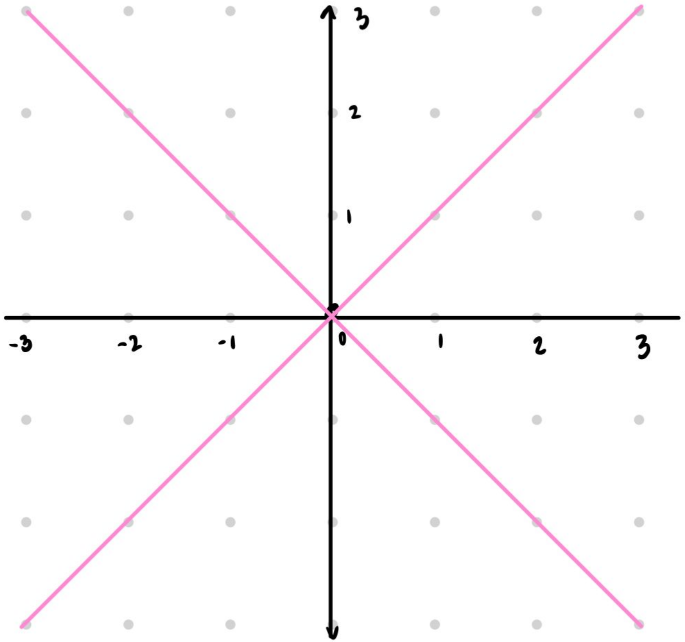


If all points on the line of y = x are in the relation then the relation is reflexive

The relation is reflexive!

Task 4b

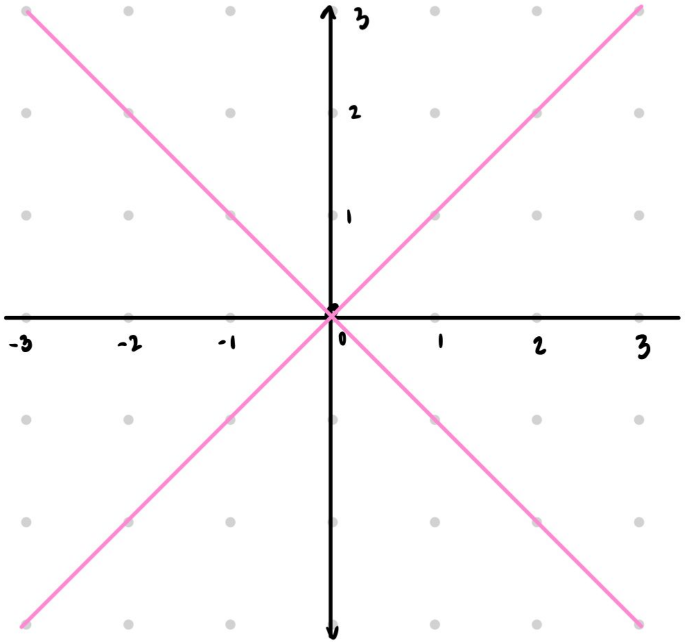
Let $R = \{(x, y) \ : \ x^2 = y^2\}$ on $\mathbb{R}$.



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

Task 4b

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

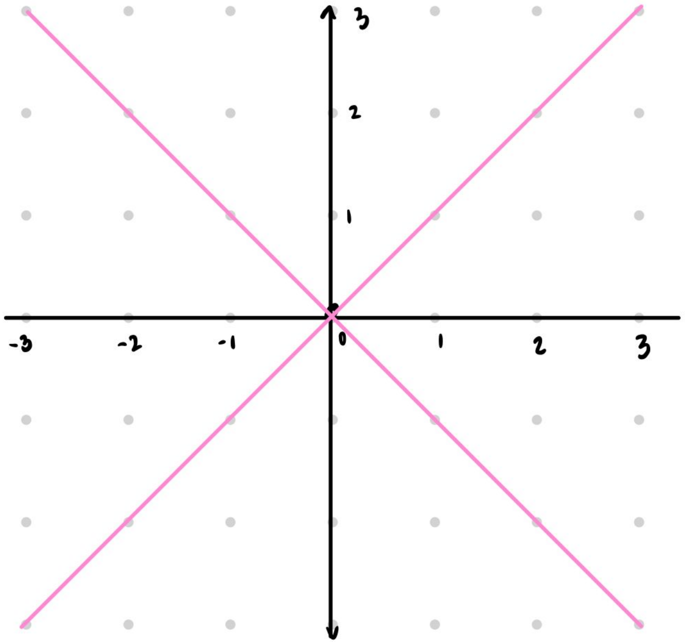


If all points that are reflected across y = x are also in the relation, then the relation is symmetric

The relation is symmetric!

Task 4b

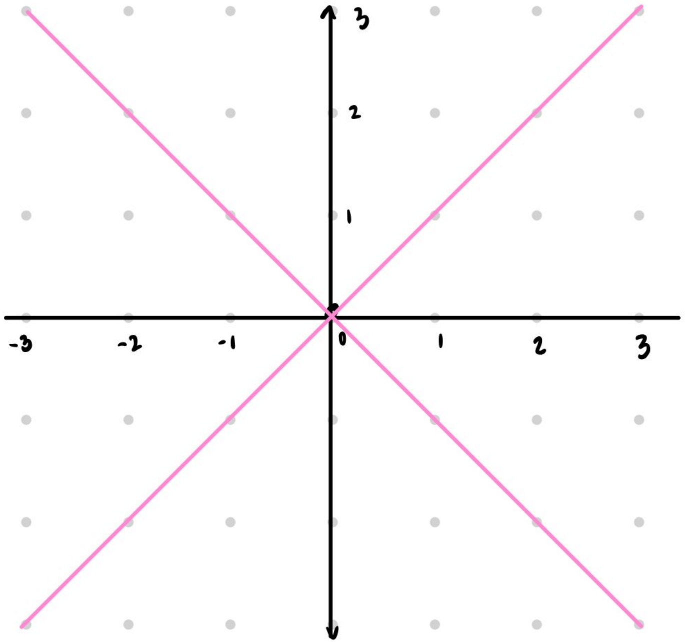
Let $R = \{(x, y) \ : \ x^2 = y^2\}$ on $\mathbb{R}$.



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

Task 4b

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!

Task 4b

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!

Task 6

Let R be a relation on a set A. Given that R is reflexive, it follows that $R \subseteq R \circ R$.

Task 6

Let R be a relation on a set A. Given that R is reflexive, it follows that $R \subseteq R \circ R$.

R is reflexive means $\forall a \in A, (a, a) \in R$

We are trying to prove $\forall x, x \in R \rightarrow x \in R \circ R$

It can be helpful to convert definitions to formal logic so you know precisely what you are proving and where to get started.

Task 6

Let R be a relation on a set A. Given that R is reflexive, it follows that $R \subseteq R \circ R$.

Let x be an arbitrary object.

Since x was arbitrary, we have proven that R is a subset of $R \circ R$.

Task 6

Let R be a relation on a set A. Given that R is reflexive, it follows that $R \subseteq R \circ R$.

Let x be an arbitrary object.

Suppose that $x \in R$.

Since x was arbitrary, we have proven that R is a subset of $R \circ R$.

Task 6

Let R be a relation on a set A. Given that R is reflexive, it follows that $R \subseteq R \circ R$.

Let x be an arbitrary object.

Suppose that $x \in R$.

Since $x \in R$, $x = (a, b)$ for some a and b in A.

Since x was arbitrary, we have proven that R is a subset of $R \circ R$.

Task 6

Let R be a relation on a set A. Given that R is reflexive, it follows that $R \subseteq R \circ R$.

Let x be an arbitrary object.

Suppose that $x \in R$.

Since $x \in R$, $x = (a, b)$ for some a and b in A.

We want to show that (a,b) ∈ R ○ R. So we need to show that there is some c such that (a,c) is in R and (c,b) in R

Thus, $x \in R \circ R$, by the definition of composition.

Since x was arbitrary, we have proven that R is a subset of $R \circ R$.

Task 6

Let R be a relation on a set A. Given that R is reflexive, it follows that $R \subseteq R \circ R$.

Let x be an arbitrary object.

Suppose that $x \in R$.

Since $x \in R$, $x = (a, b)$ for some a and b in A.

Since R is reflexive, we know that $(b, b) \in R$ as well. This shows that there is some $c \in A$ (namely, $c = b$) such that $(a, c) \in R$ and $(c, b) \in R$.

Thus, $x \in R \circ R$, by the definition of composition.

Since x was arbitrary, we have proven that R is a subset of $R \circ R$.

That's All, Folks!

Thanks for coming to section this week!
Any questions?