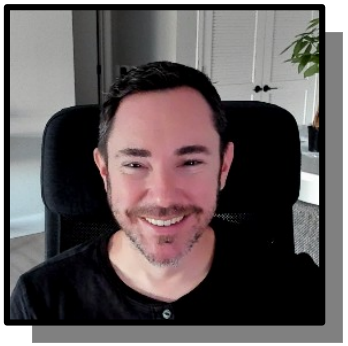
CS103
FALL 2025

Lecture 00:

Intro to Set Theory

Are there “laws of physics”
in computer science?



Sean Szumlanski
(Instructor)

Keith Schwarz
(Instructor)

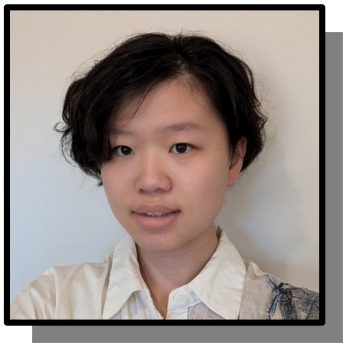
Evelyn Yee
(ACE Instructor)

Ari Barbella-Blaha
(TA)

Ethan Bliss
(TA)

Tushar Dalmia
(TA)

Andrew Li
(TA)



Kaia Li
(TA)

Kenneth Ma
(TA)

Usman Tariq
(TA)

Course Website

https://cs103.stanford.edu

All course content will be hosted here, except for lecture videos.

Prerequisite / Corequisite

CS106B

Some problem sets will have small coding components. We'll also reference some concepts from CS106B, particularly recursion, throughout the quarter.

There aren't any math prerequisites for this course. High-school algebra should be enough!

Another Option

CS154

CS154 is more appropriate if you have a background in the topics from the first half of this quarter (set theory, proofwriting, discrete math, formal logic, graphs, etc.) Come talk to me after class if you're curious about this!

CS103 ACE

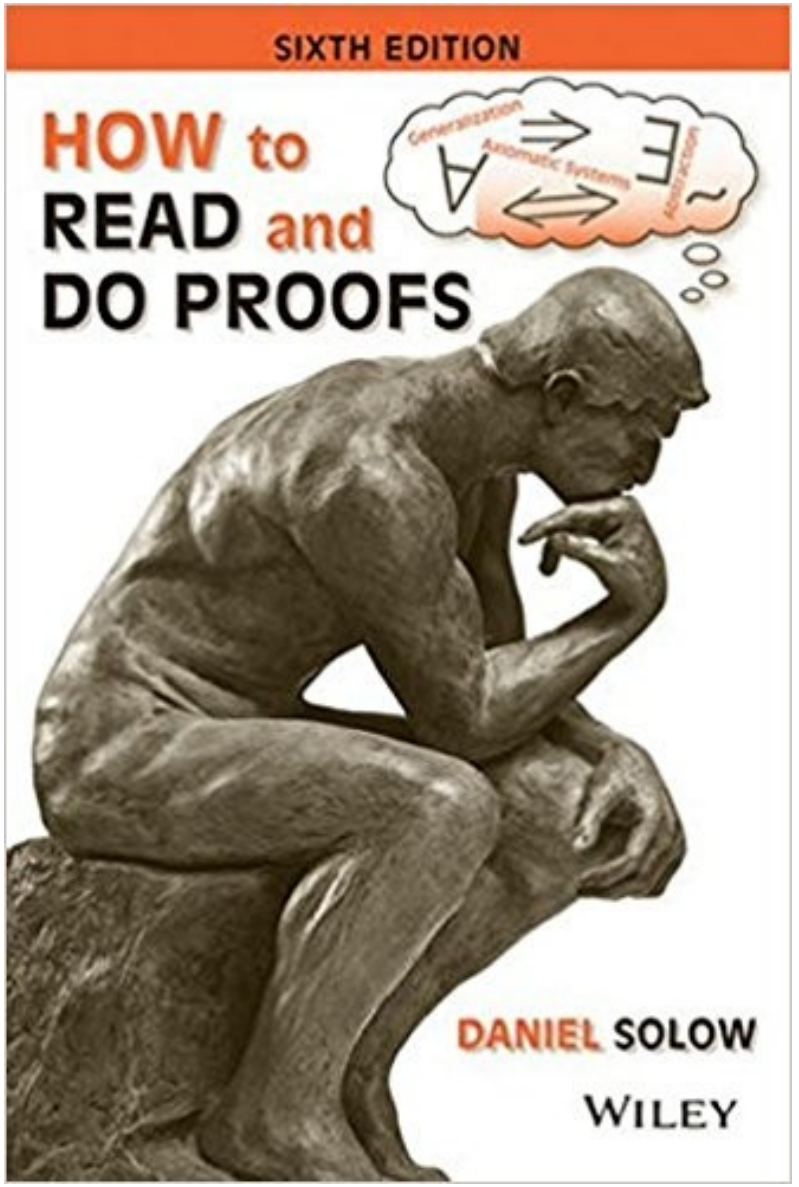
- ***CS103 ACE*** is an optional, one-unit companion course to CS103.
- CS103 ACE meets Thursdays, 1:30PM – 3:20PM and provides additional practice with the course material in a small group setting.
- This Thursday's class meeting is an "office hours"-style session where you can stop by to meet Evelyn and learn more about the course.
- Interested? Apply online at ***this link***.

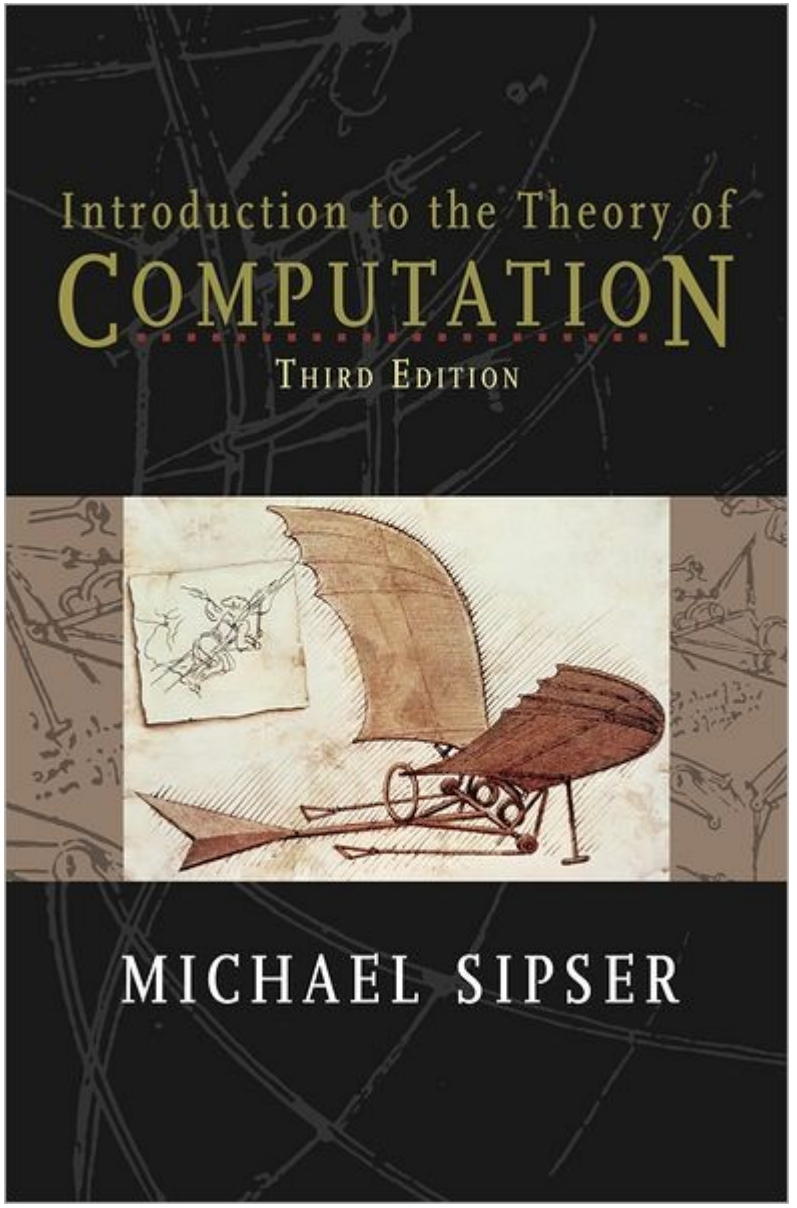
Evelyn Yee
(ACE Instructor)

Problem Set 0

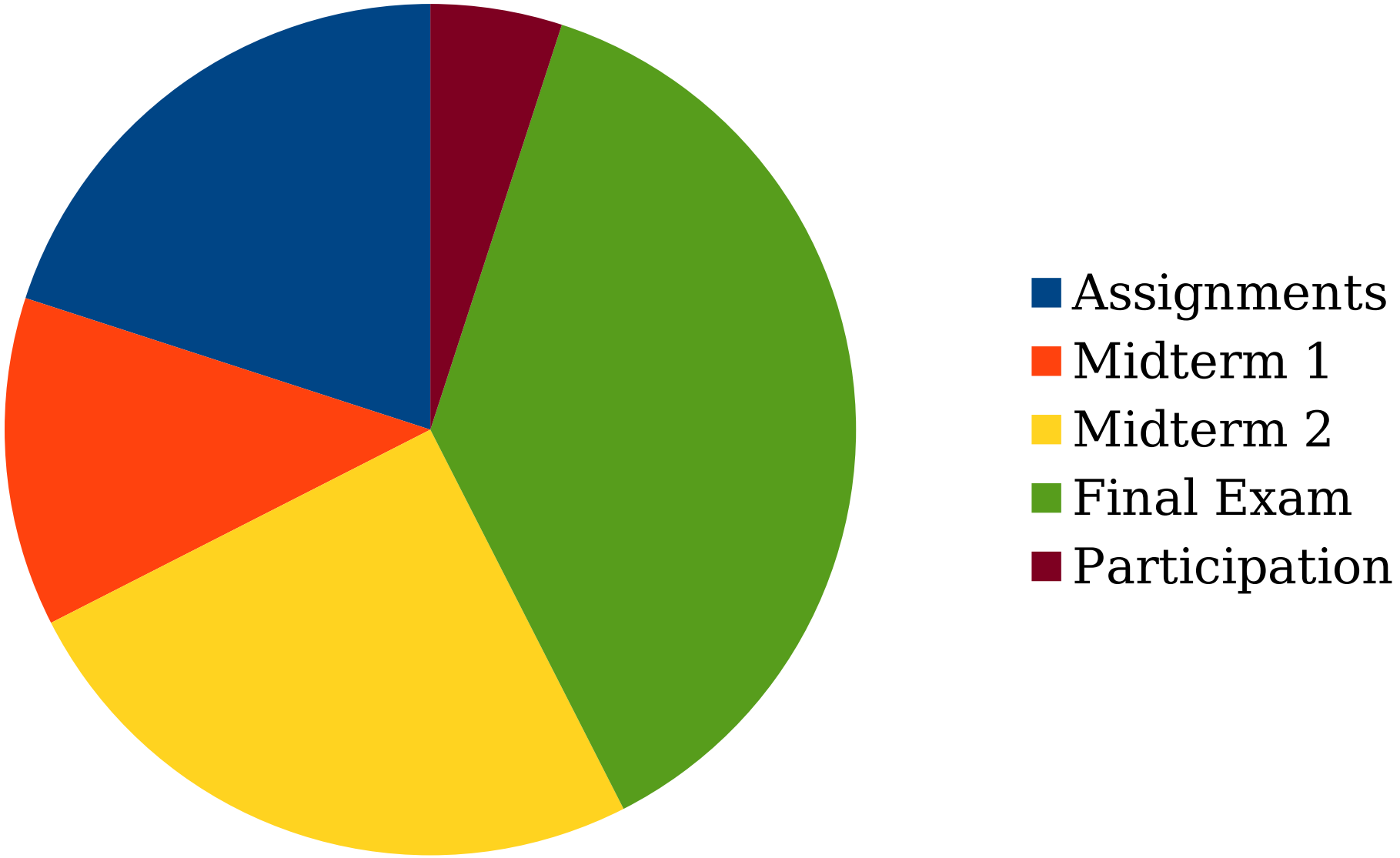
- Problem Set 0, goes out today. It's due ***Friday*** at ***1:00PM Pacific***.
  - Set up your development environment.
  - Familiarize yourself with GradeScope and EdStem.
  - There's no coding or math involved.
- ***Start early*** in case you encounter any technical issues.
  - There will be a Qt Creator install help session this Thursday, 7PM – 9PM in CoDa B45.

Recommended Reading





Grading



Approaching this Course

- This course will teach you to think and reason in an entirely new way. This takes time and practice.
- Our recommendations:
  - ***Be present***. Come to class each day instead of batching lectures up. Take notes by hand: lots of what we cover in lecture isn't in the slides.
  - ***Be proactive***. Start problem sets early; this is the #1 piece of advice CS103 veterans have for future students!
  - ***Be curious***. Don't accept a shaky understanding of a concept. Ask questions when something doesn't click.
  - ***Be humble***. Some of your intuitions will be spot-on. Some will be entirely off. Some things that seem trivial are incredibly important, and vice-versa. Be open to feedback.
- We are here to help you learn this material. Let us know how we can help you!

We've got a big journey ahead of us.

Let's get started!

Introduction to Set Theory

“CS103 students”

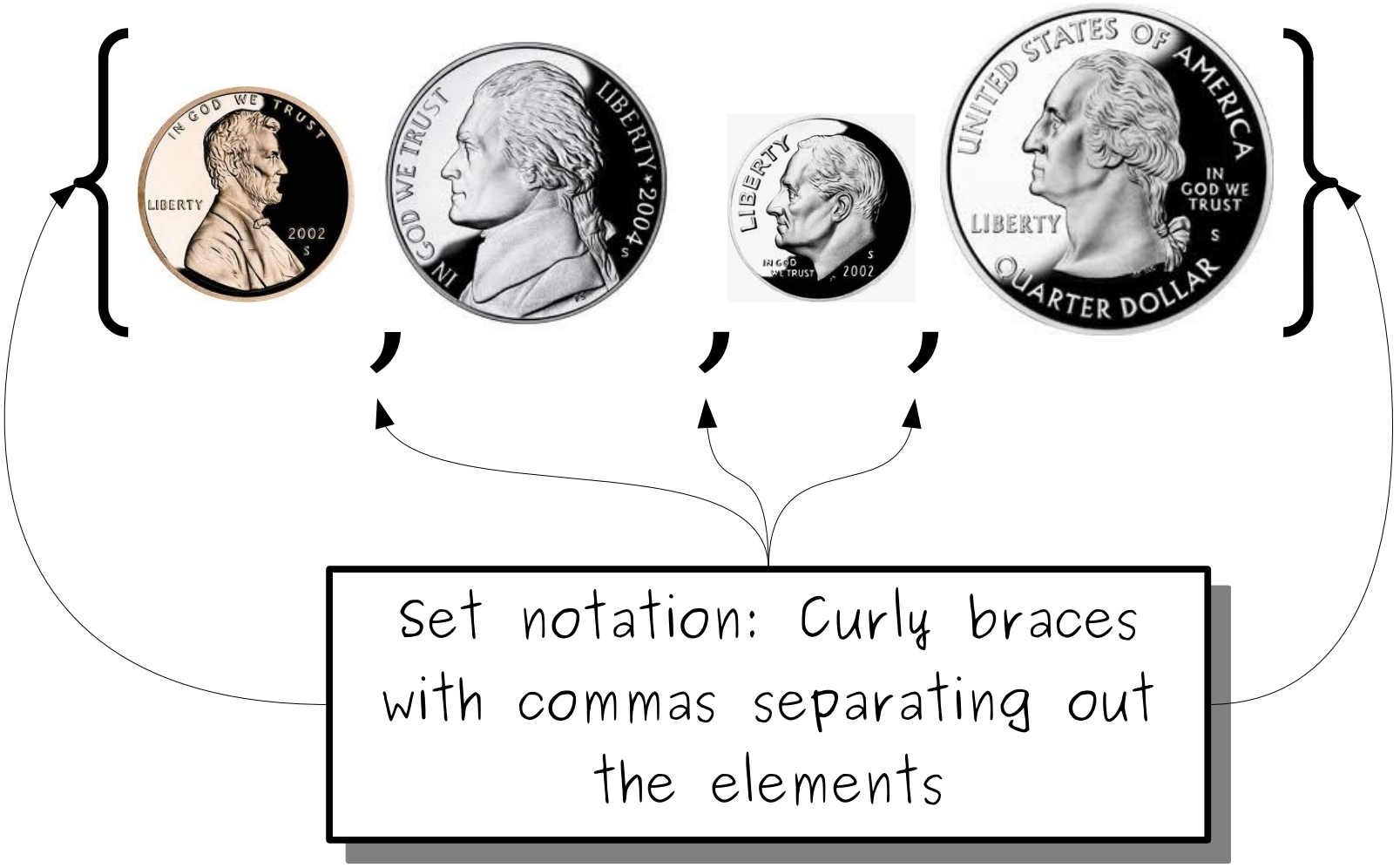
“Cool people”

“The chemical elements”

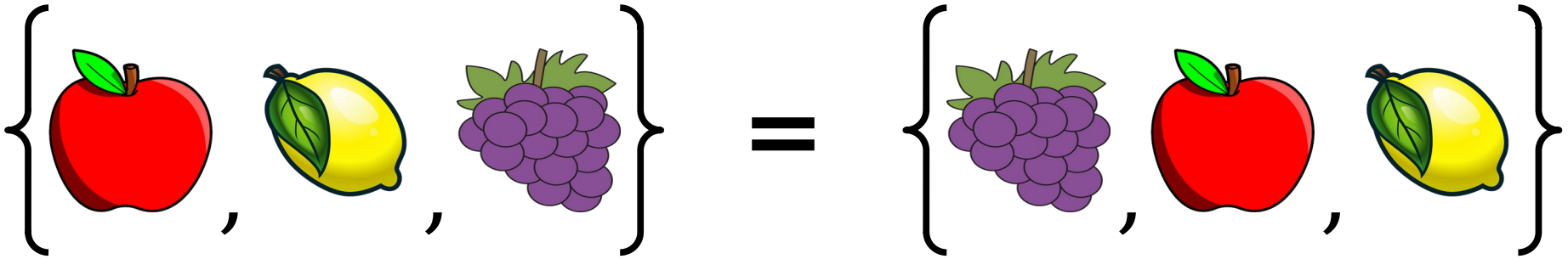
“Cute animals”

“US coins”

A ***set*** is an unordered collection of distinct objects, which may be anything, including other sets.

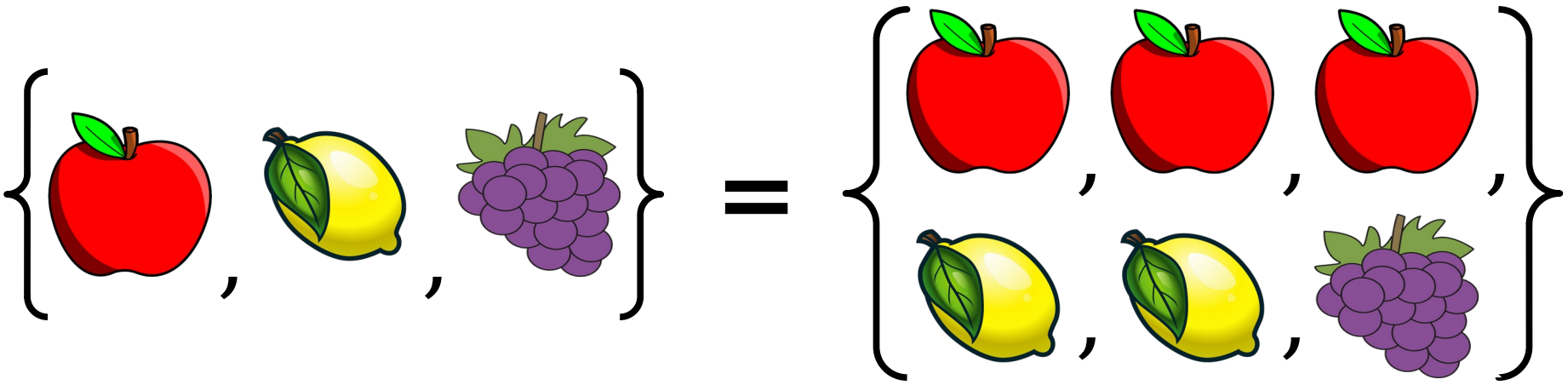


A ***set*** is an unordered collection of distinct objects, which may be anything, including other sets.



These are two descriptions of the same set.

Two sets are equal when they have the same contents, ignoring order.



These are two descriptions of the same set.

Sets cannot contain duplicate elements.
Any repeated elements are ignored.

The objects that make up a set are called the ***elements*** of that set.

$\notin \{$

,

,

,

$\}$

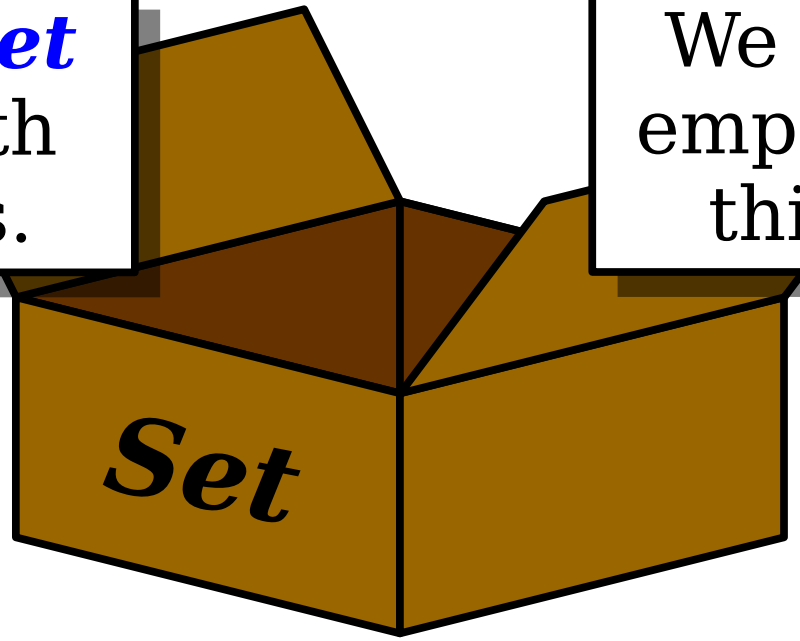
This symbol means "is not an element of."

The objects that make up a set are called the ***elements*** of that set.

$\{\ \} = \emptyset$

The ***empty set*** is the set with no elements.

We denote the empty set using this symbol.



Sets can contain any number of elements.

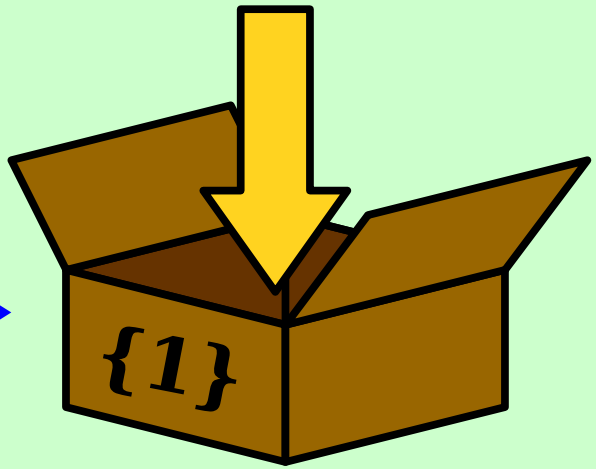
$$1 \neq \{1\}$$

This is a number.

1

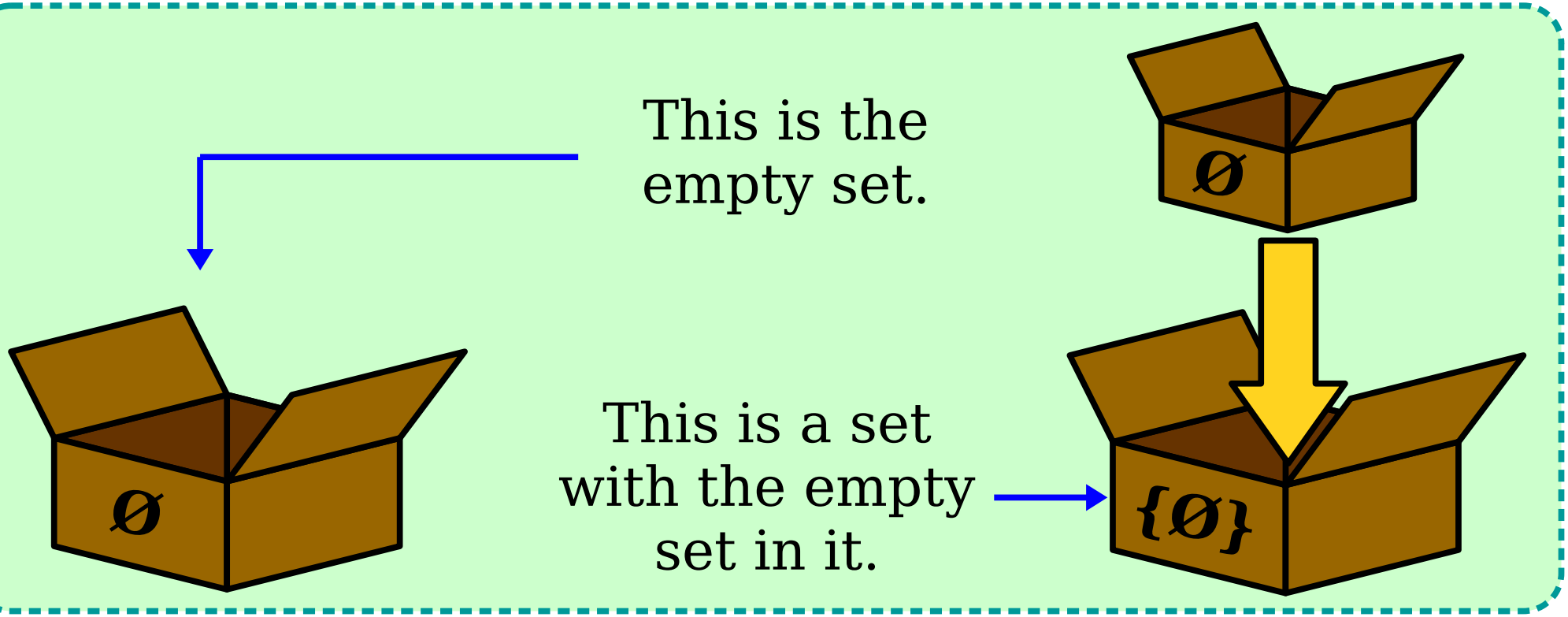
1

This is a set.
It contains a number.



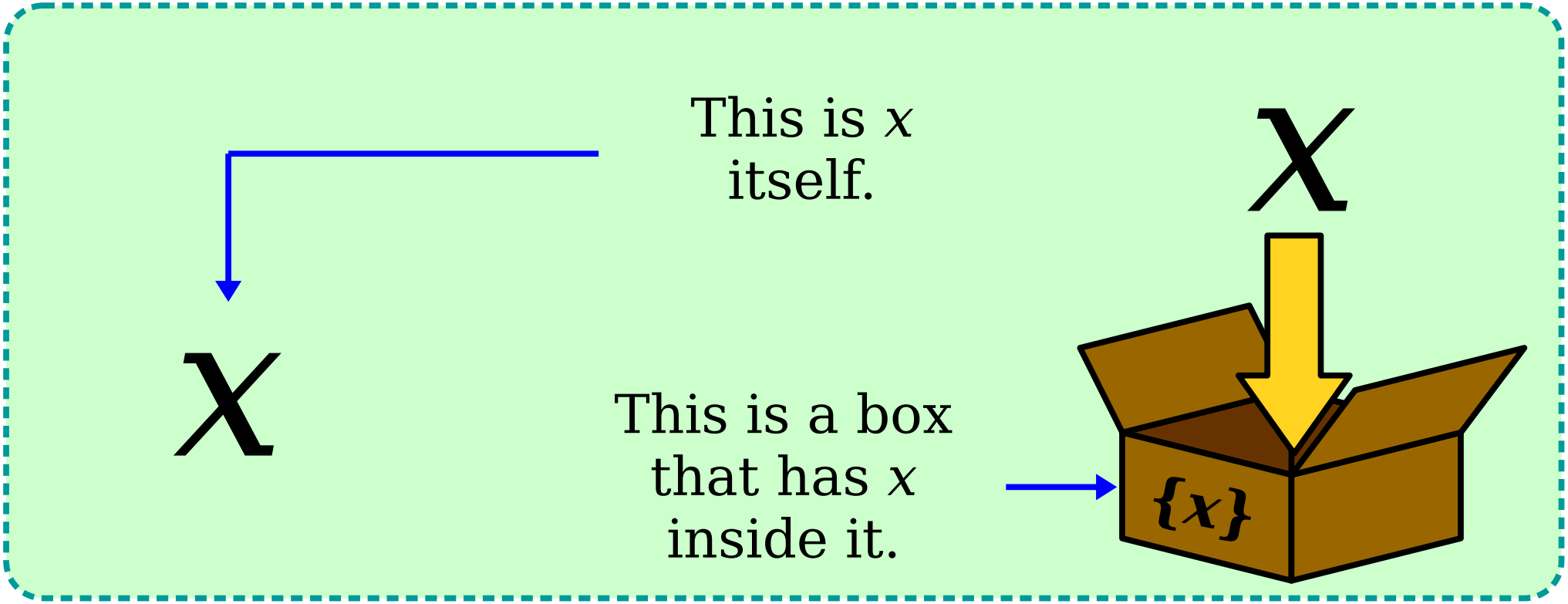
Question: Are these objects equal?

$$\emptyset \neq \{\emptyset\}$$



Question: Are these objects equal?

$$x \neq \{x\}$$



No object x is equal to the set containing x.

Infinite Sets

- Some sets contain *infinitely many* elements!
- The set $\mathbb{N} = \{ 0, 1, 2, 3, \ldots \}$ is the set of all the ***natural numbers***.
  - Some mathematicians don't include zero; in this class, assume that 0 is a natural number.
- The set $\mathbb{Z} = \{ \ldots, -2, -1, 0, 1, 2, \ldots \}$ is the set of all the ***integers***.
  - Z is from German “Zahlen.”
- The set $\mathbb{R}$ is the set of all ***real numbers***.
  - $e \in \mathbb{R}$, and $4 \in \mathbb{R}$, and $-137 \in \mathbb{R}$,

Describing Complex Sets

- Here are some English descriptions of infinite sets:

  “The set of all even natural numbers.”

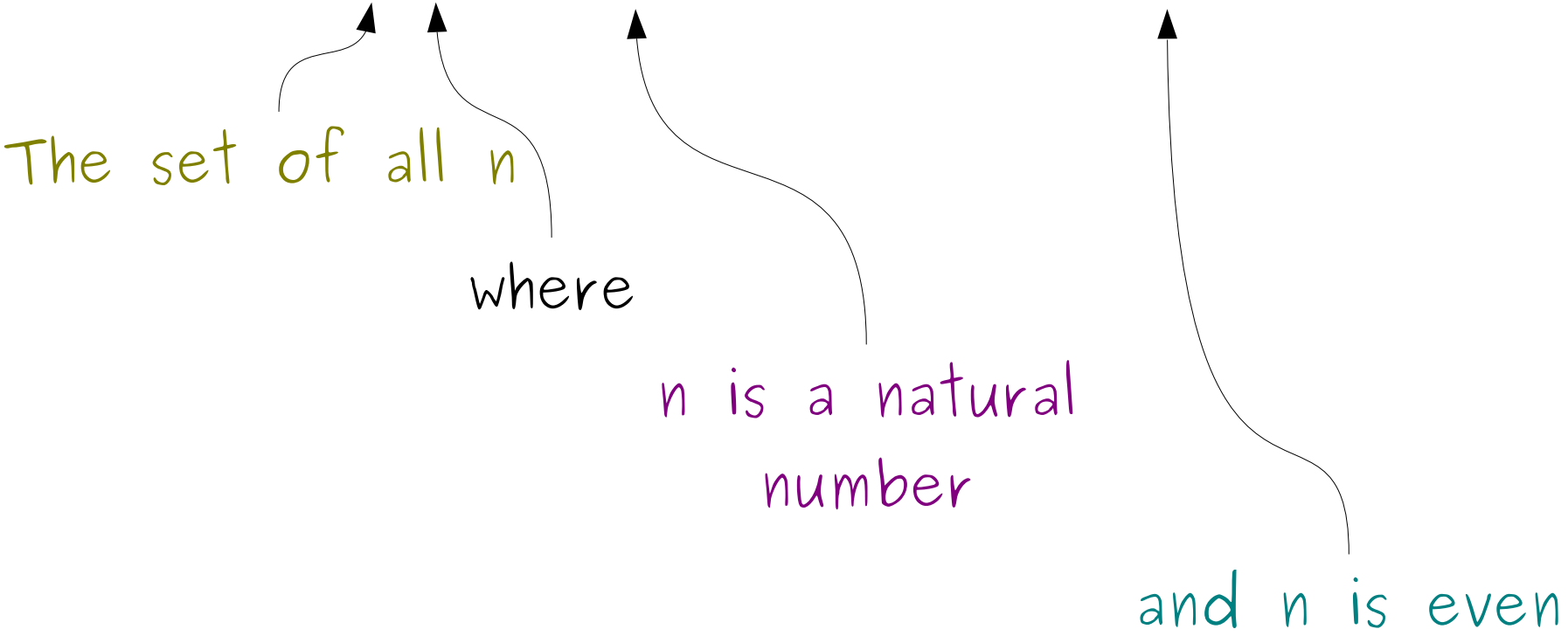
  “The set of all real numbers less than 137.”

  “The set of all Python programs.”

- To describe complex sets like these mathematically, we'll use ***set-builder notation***.

Even Natural Numbers

$$\{\ n \mid n \in \mathbb{N} \text{ and } n \text{ is even}\ \}$$



$$\{\ 0, 2, 4, 6, 8, 10, 12, 14, 16, \ldots\ \}$$

Set Builder Notation

- A set may be specified in ***set-builder notation***:

$$\{\ x \mid \textit{some property } x \textit{ satisfies}\ \}$$

$$\{\ x \in S \mid \textit{some property } x \textit{ satisfies}\ \}$$

- For example:

$\{\ n \mid n \in \mathbb{N} \text{ and } n \text{ is even}\ \}$

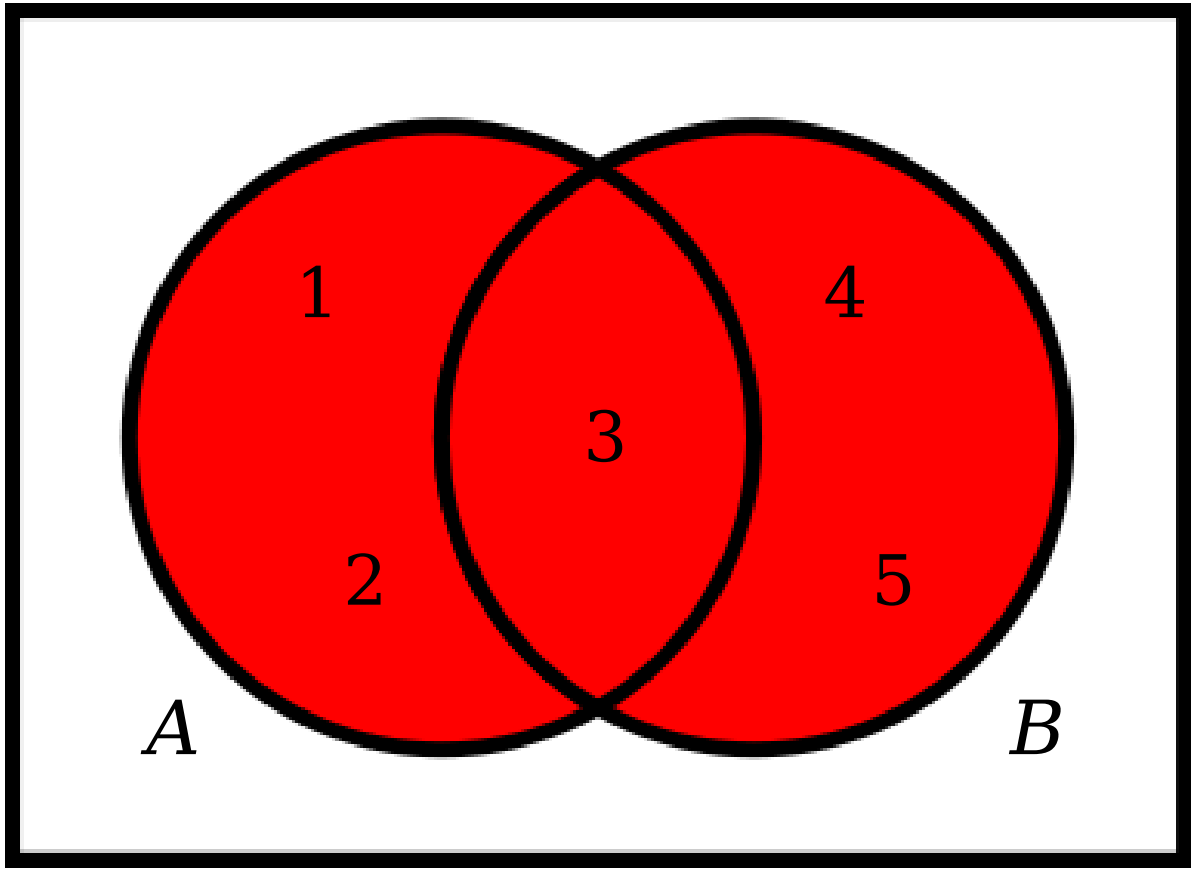
$\{\ C \mid C \text{ is a set of US coins}\ \}$

$\{\ r \in \mathbb{R} \mid r < 3\ \}$

$\{\ n \in \mathbb{N} \mid n < 3\ \}$ **(the set {0, 1, 2})**

Combining Sets

Venn Diagrams



Union

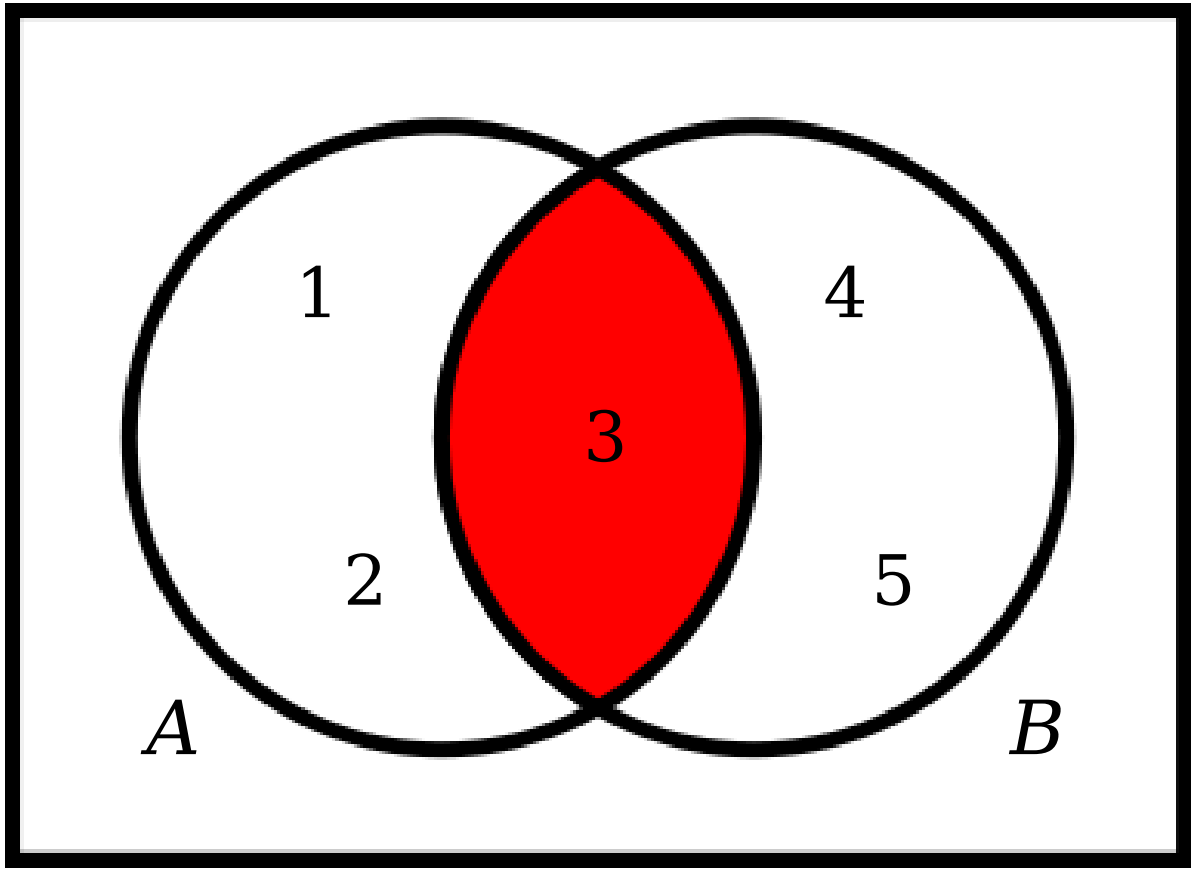
$A \cup B$

{ 1, 2, 3, 4, 5 }

$A = \{ 1, 2, 3 \}$

$B = \{ 3, 4, 5 \}$

Venn Diagrams



Intersection

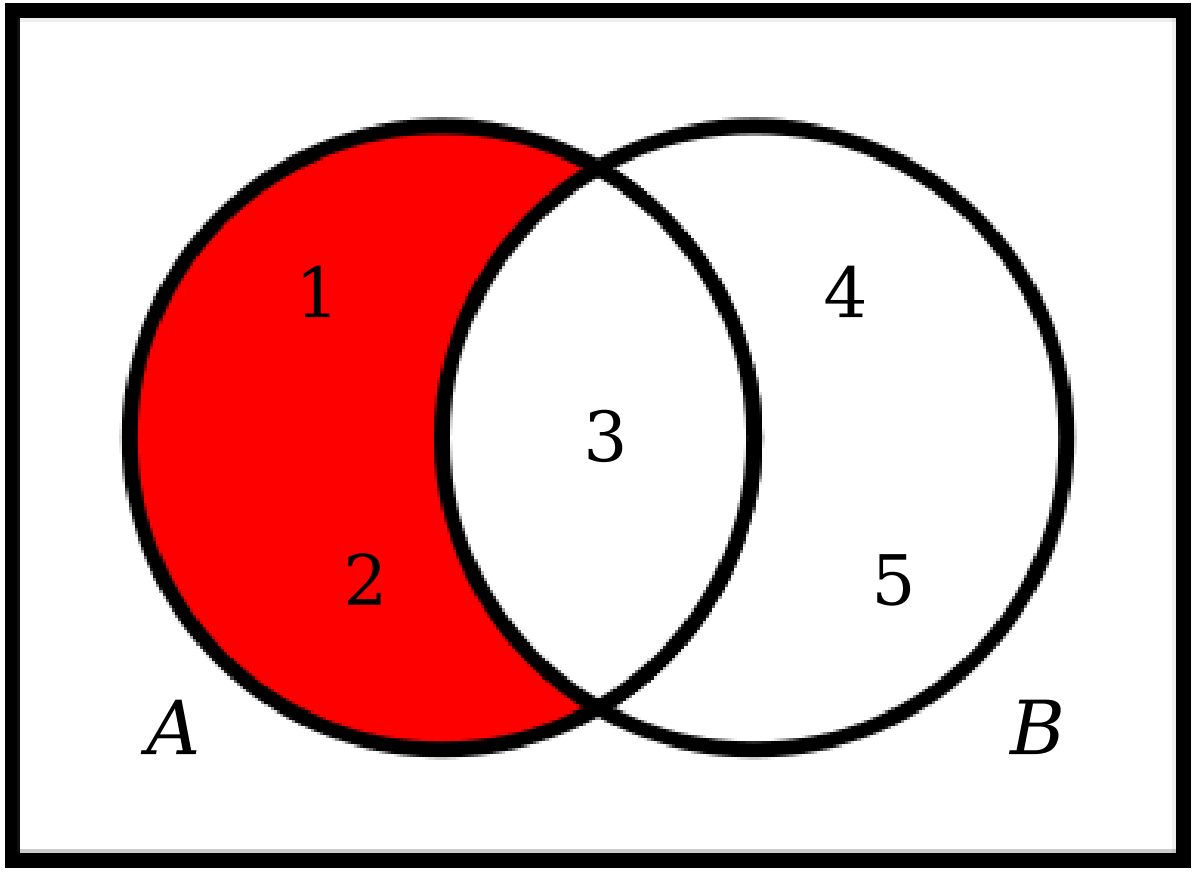
$A \cap B$

{ 3 }

$A = \{ 1, 2, 3 \}$

$B = \{ 3, 4, 5 \}$

Venn Diagrams



Difference

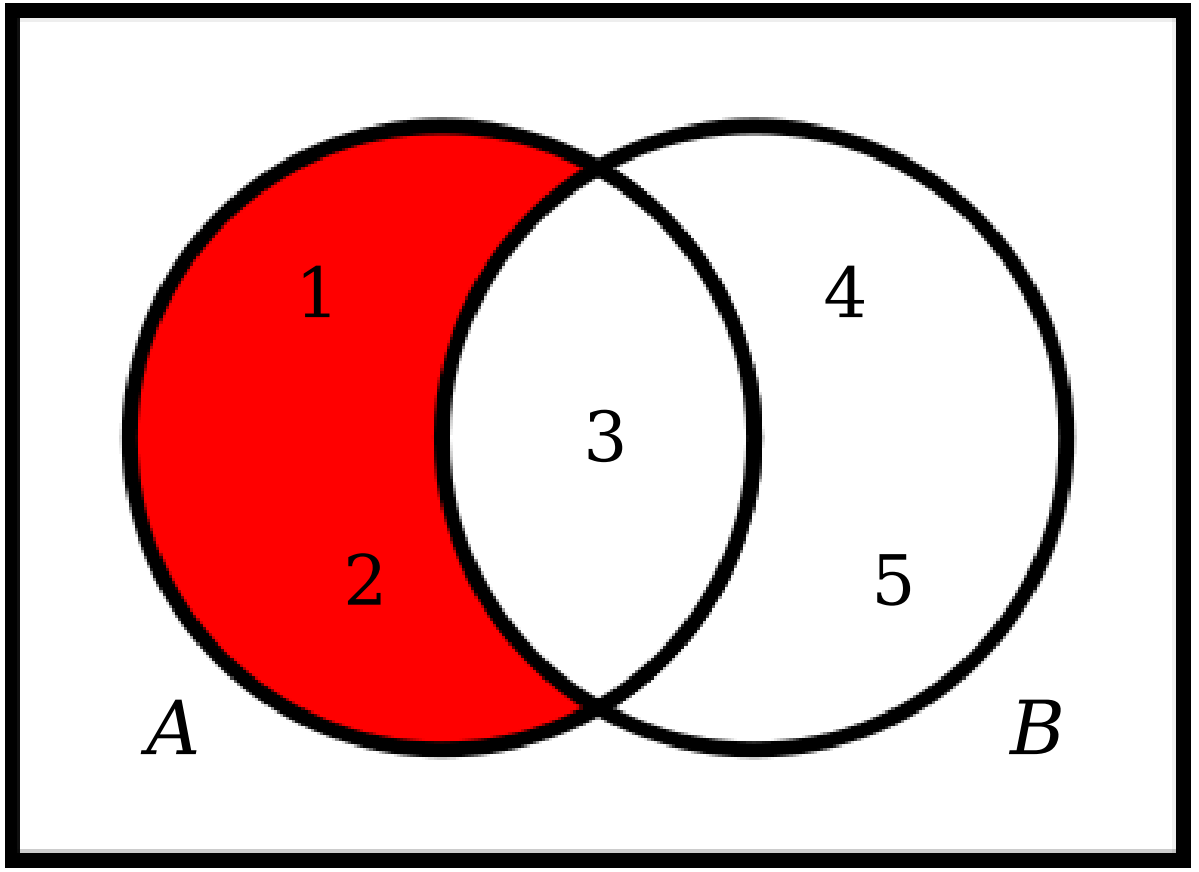
$A - B$

{ 1, 2 }

A = { 1, 2, 3 }

B = { 3, 4, 5 }

Venn Diagrams



Difference

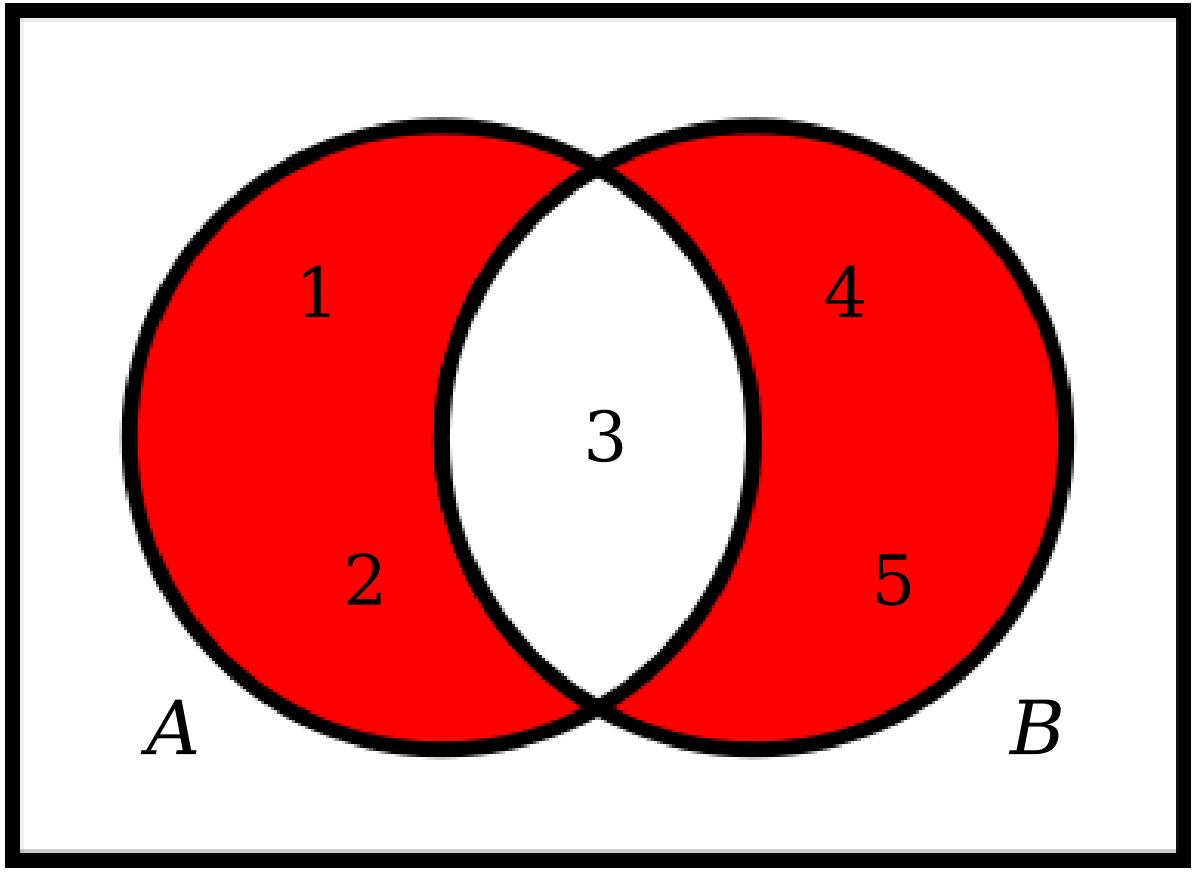
$A \setminus B$

{ 1, 2 }

$A = \{ 1, 2, 3 \}$

$B = \{ 3, 4, 5 \}$

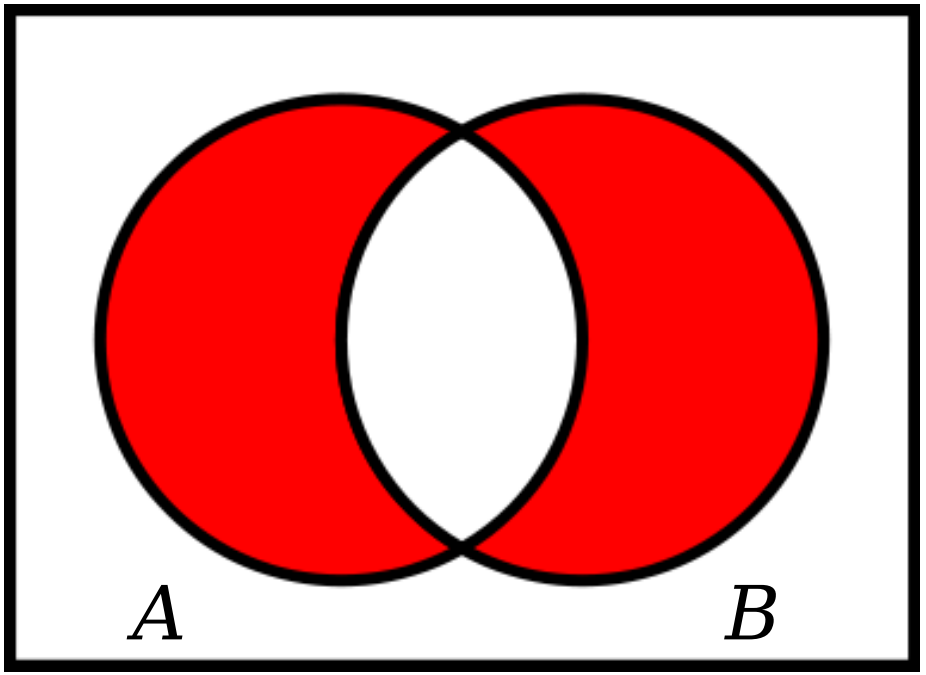
Venn Diagrams

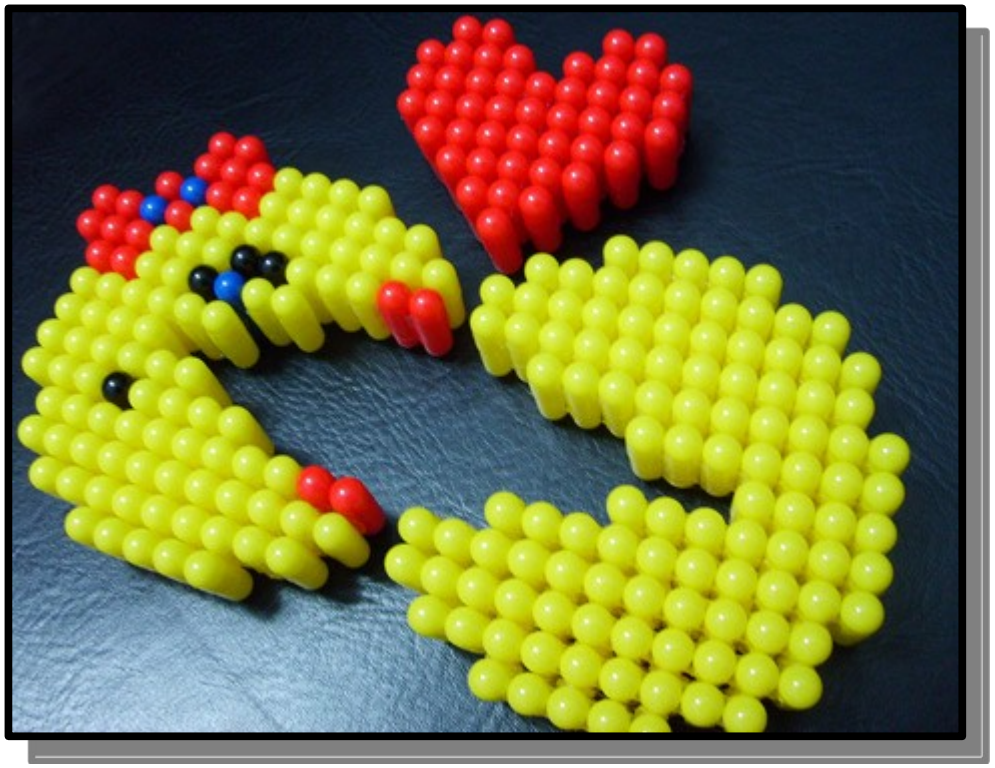


Symmetric Difference
$A \Delta B$
{ 1, 2, 4, 5 }

$A = \{ 1, 2, 3 \}$
$B = \{ 3, 4, 5 \}$

Venn Diagrams





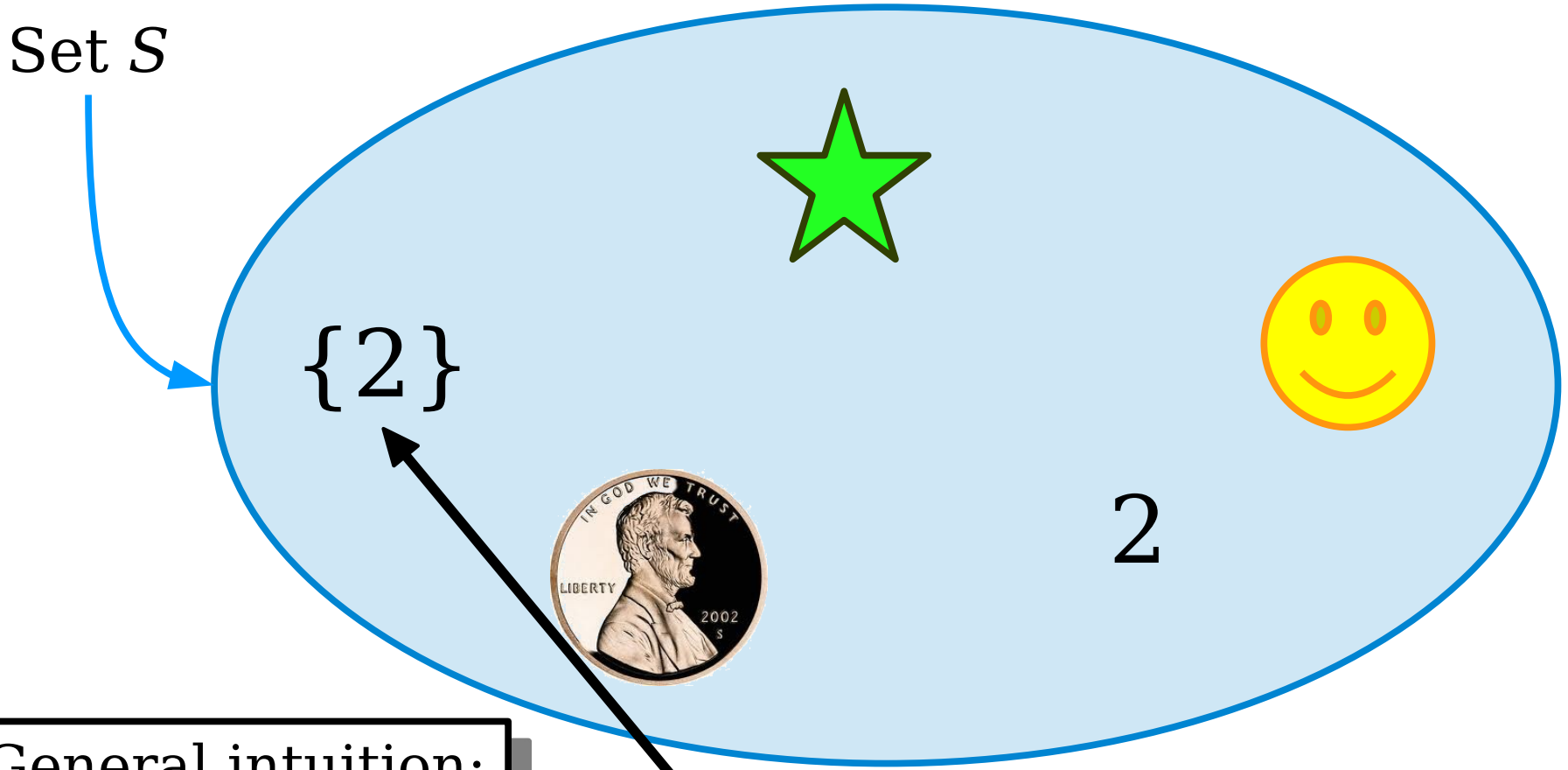
$A \Delta B$

Subsets and Power Sets

Subsets

- A set S is called a ***subset*** of a set T (denoted $\boldsymbol{S \subseteq T}$) when all elements of S are also elements of T.
- Examples:
  - $\{ 1, 2, 3 \} \subseteq \{ 1, 2, 3, 4 \}$
  - $\{ b, c \} \subseteq \{ a, b, c, d \}$
  - $\{ \mathrm{H}, \mathrm{He}, \mathrm{Li} \} \subseteq \{ \mathrm{H}, \mathrm{He}, \mathrm{Li} \}$
  - $\mathbb{N} \subseteq \mathbb{Z}$ (*every natural number is an integer*)
  - $\mathbb{Z} \subseteq \mathbb{R}$ (*every integer is a real number*)

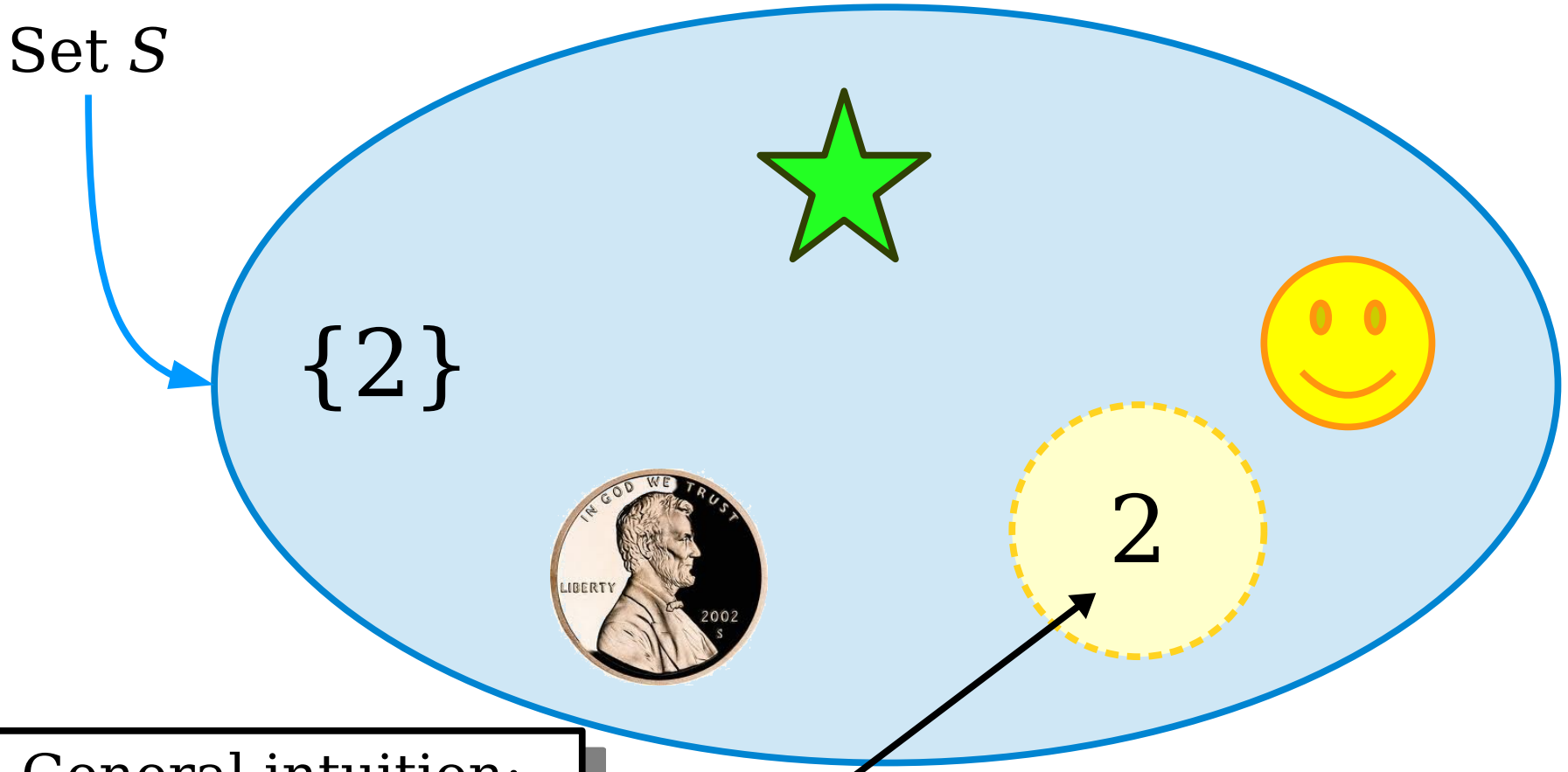
Subsets and Elements



$\{2\} \in S$

General intuition: $x \in S$ means you can ***point at x inside of S***.

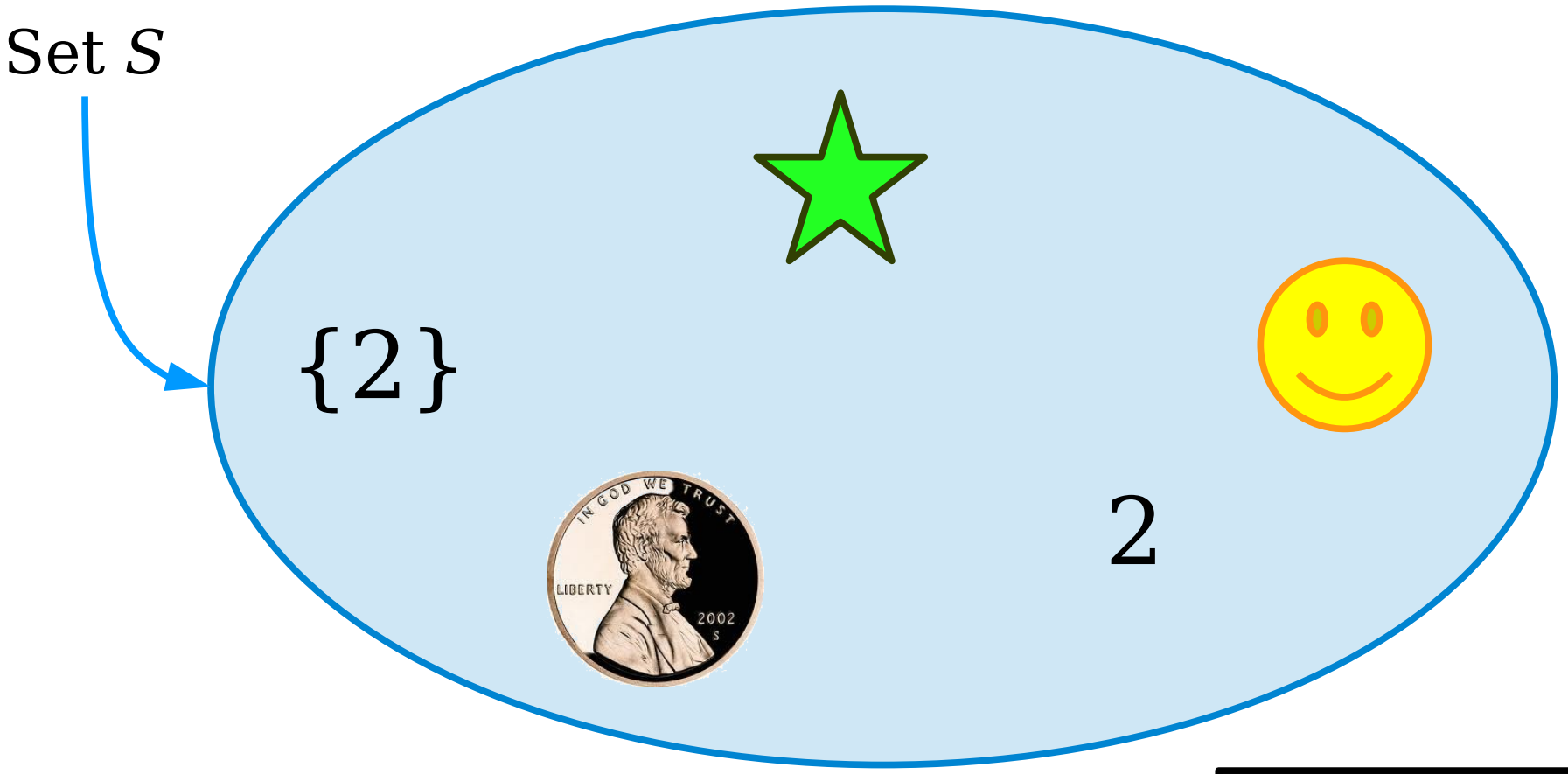
Subsets and Elements



General intuition:
$A \subseteq B$ if you can form A by ***circling elements of B***.

$\{2\} \subseteq S$

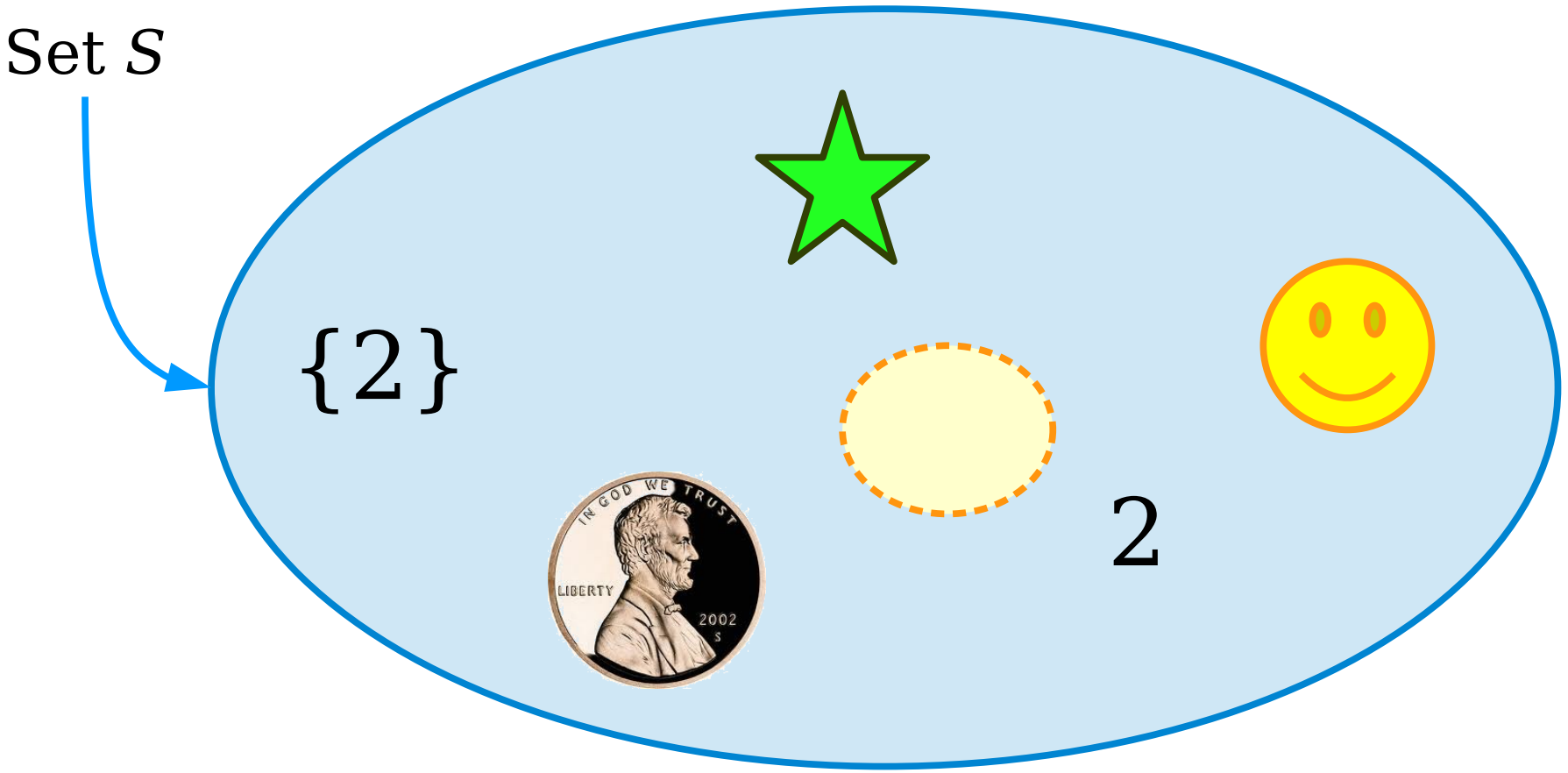
Subsets and Elements



$2 \notin S$

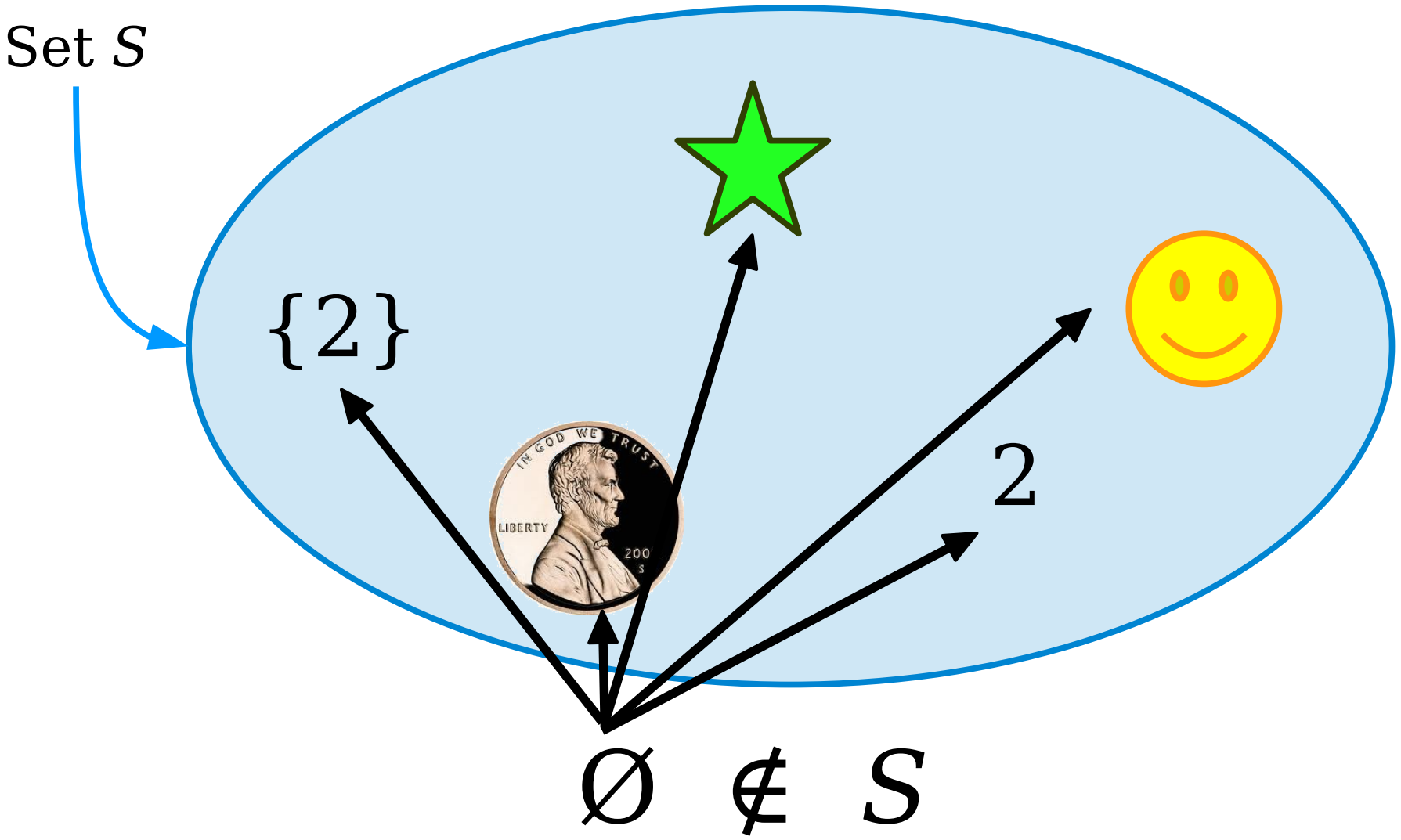
(Since 2 isn't a set.)

Subsets and Elements



$$\emptyset \subseteq S$$

Subsets and Elements



Subsets and Elements

- We say that $S \in T$ when, among the elements of T, one of them is *exactly* the object S.
- We say that $S \subseteq T$ when S is a set and every element of S is also an element of T. (S has to be a set for the statement $S \subseteq T$ to be true.)
- Although these concepts are similar, ***they are not the same!*** Not all elements of a set are subsets of that set and vice-versa.
- We have a resource on the course website, the Guide to Elements and Subsets, that explores this in more depth.

$$S = \left\{ \text{  }, \text{  } \right\}$$

$$\wp(S) = \left\{ \varnothing, \{\text{  }\}, \{\text{  }\}, \{\text{  }, \text{  }\} \right\}$$

This is the ***power set*** of S, the set of all subsets of S. We write the power set of S as $\wp(S)$.

Formally, $\wp(S) = \{ T \mid T \subseteq S \}$.

(Do you see why?)

What is $\wp(\emptyset)$?

Answer: $\{\emptyset\}$

Remember that $\emptyset \neq \{\emptyset\}$*!*

Cardinality

Cardinality

- The ***cardinality*** of a set is the number of elements it contains.
- If S is a set, we denote its cardinality as $\boldsymbol{|S|}$.
- Examples:
  - $|\{whimsy, mirth\}| = 2$
  - $|\{\{a, b\}, \{c, d, e, f, g\}, \{h\}\}| = 3$
  - $|\{1, 2, 3, 3, 3, 3, 3\}| = 3$
  - $|\{\ n \in \mathbb{N} \mid n < 4\ \}| = |\{0, 1, 2, 3\}| = 4$
  - $|\ \varnothing\ | = 0$
  - $|\ \{\varnothing\}\ | = 1$

The Cardinality of ℕ

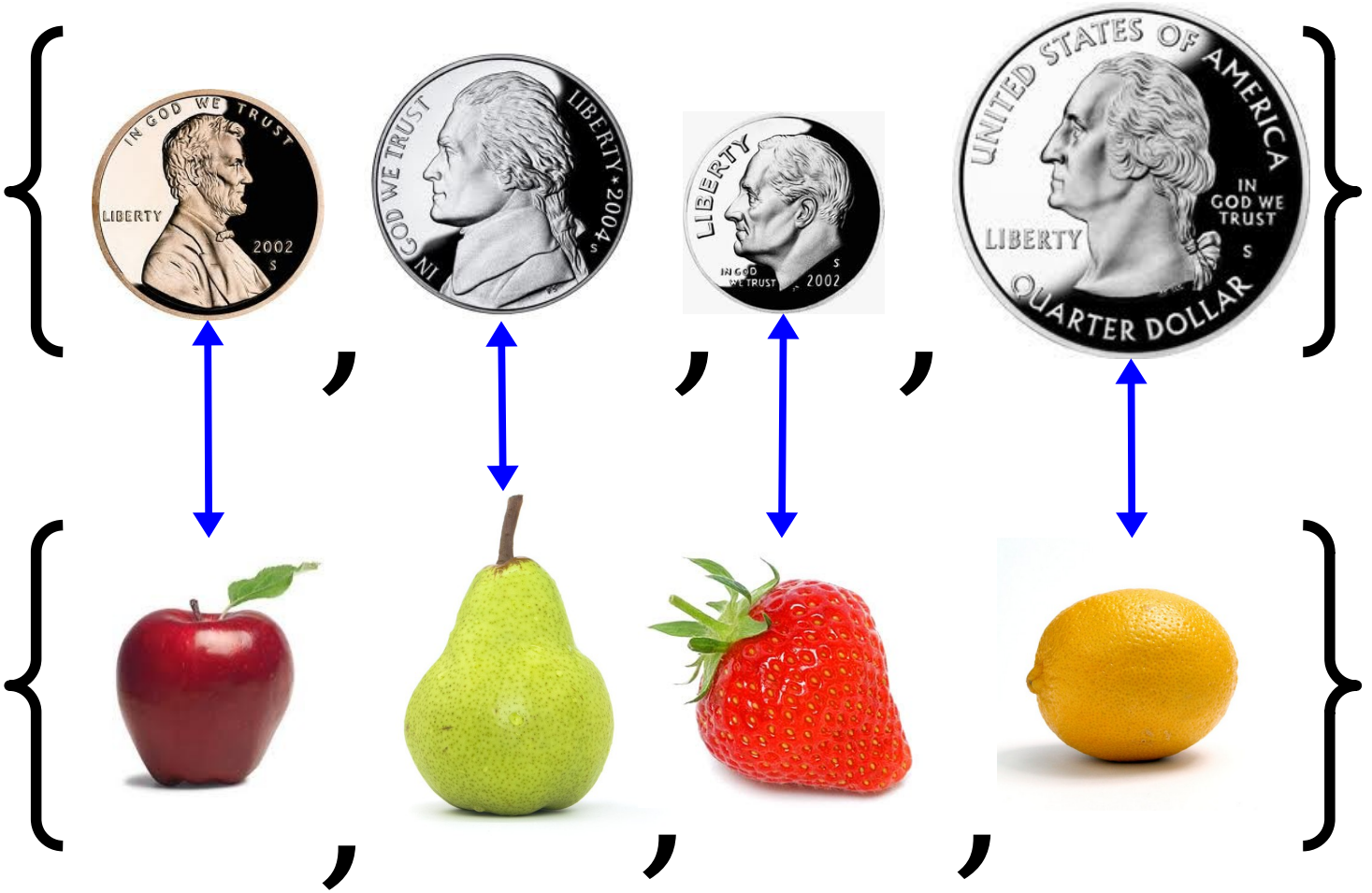
- What is |ℕ|?
  - There are infinitely many natural numbers.
  - |ℕ| can't be a natural number, since it's infinitely large.
- We need to introduce a new term.
- Let's define **ℵ₀** = |ℕ|.
  - ℵ₀ is pronounced “aleph-zero,” “aleph-nought,” or “aleph-null.”
- ***Question:*** Why don’t we say |ℕ| = ∞?

Astonishing Fact: Not all infinite sets have the same cardinality. Some infinite sets are bigger than others!

More Astonishing Fact: This has practical consequences!

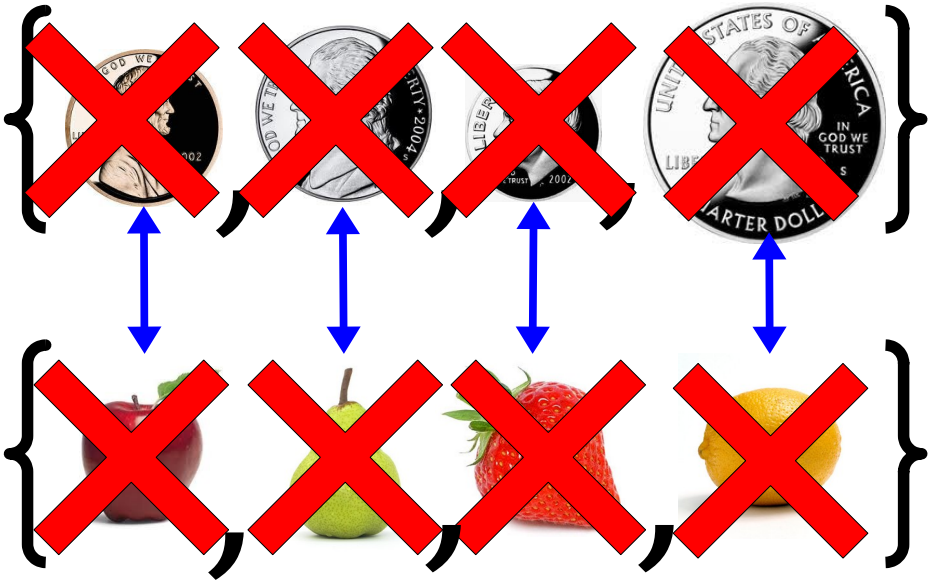
What does it mean for one set to be “bigger” than another?

How Big Are These Sets?



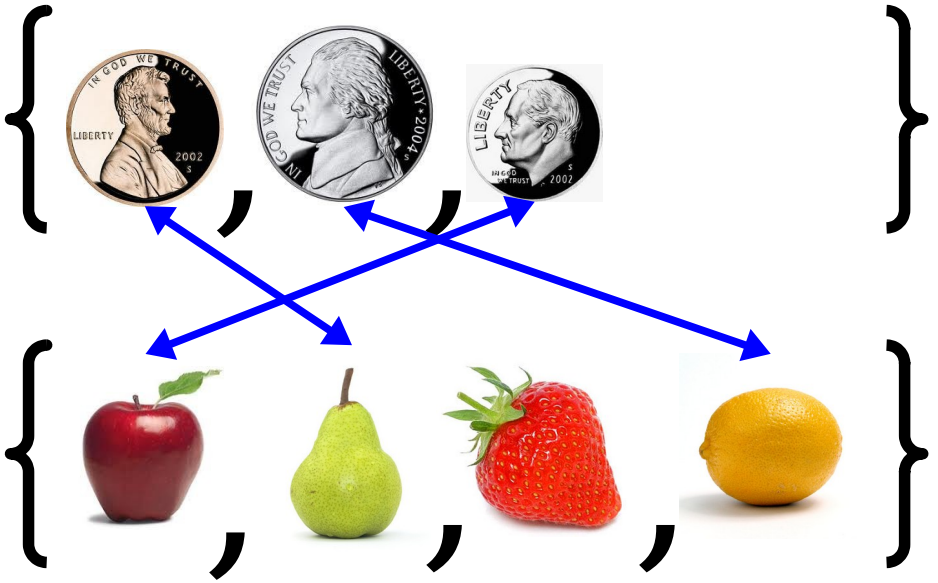
Comparing Cardinalities

- If S and T are sets, we say that $|S| = |T|$ when there is a way of pairing off the elements of S and T without leaving anything uncovered.

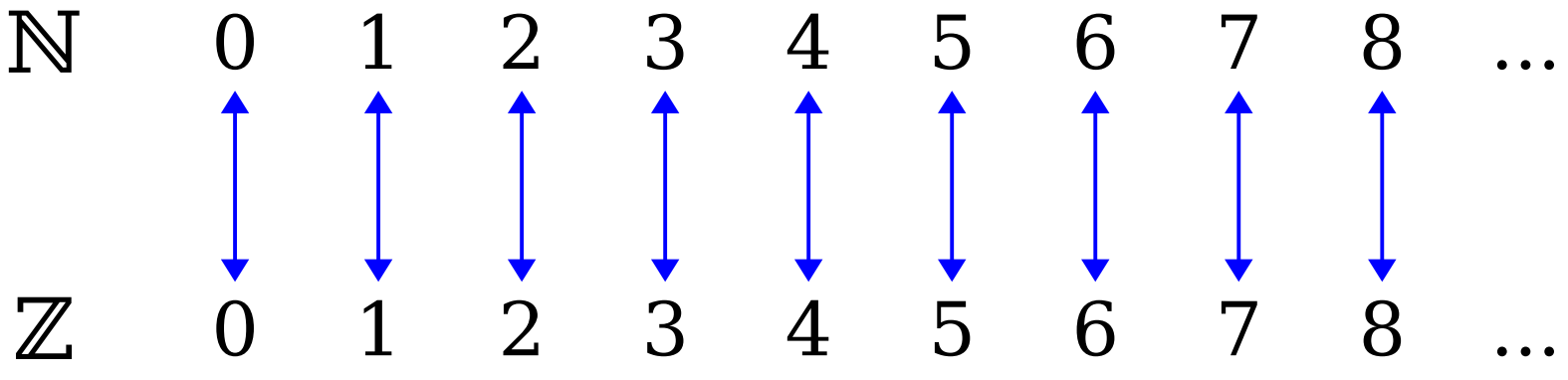


Comparing Cardinalities

- If S and T are sets, we say $|S| < |T|$ when, no matter how you pair off the elements of S and T, there's always at least one element of T left uncovered.



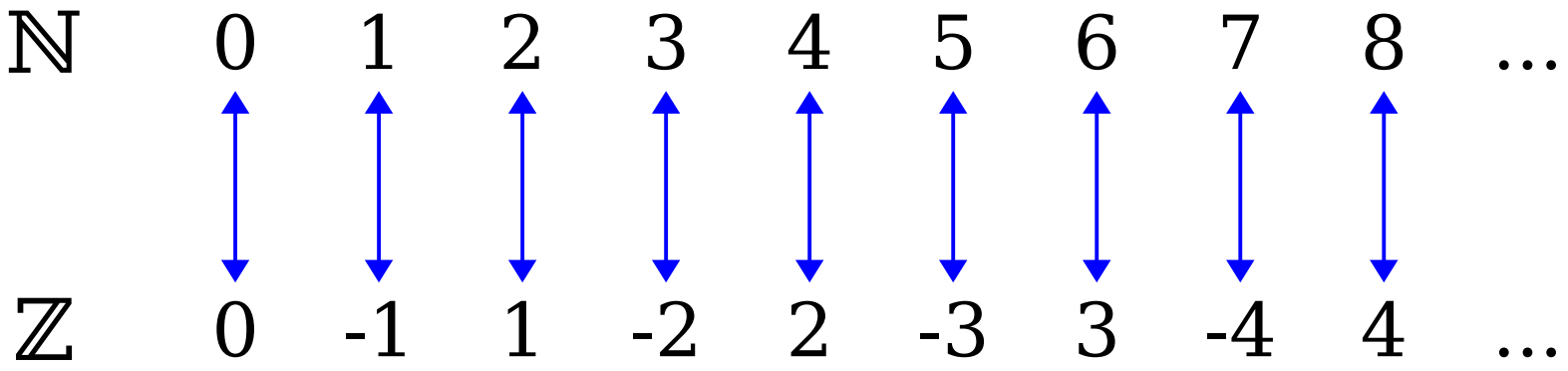
Infinite Cardinalities



One set is smaller than another when *every way* of pairing the sets off leaves an element of the second set uncovered.

Two sets have the same size when *there is a way* to pair their elements off without leaving any elements uncovered

Infinite Cardinalities



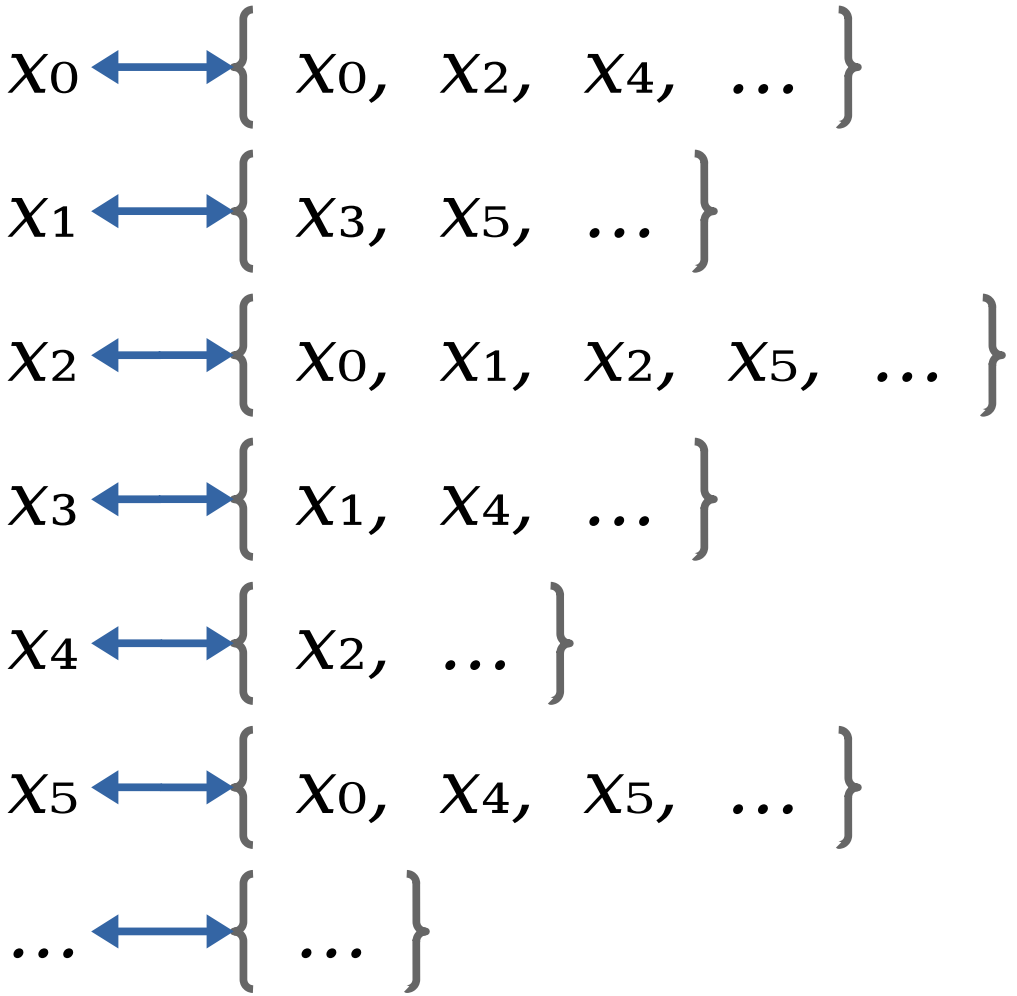
$$|\mathbb{N}| = |\mathbb{Z}| = \aleph_0$$

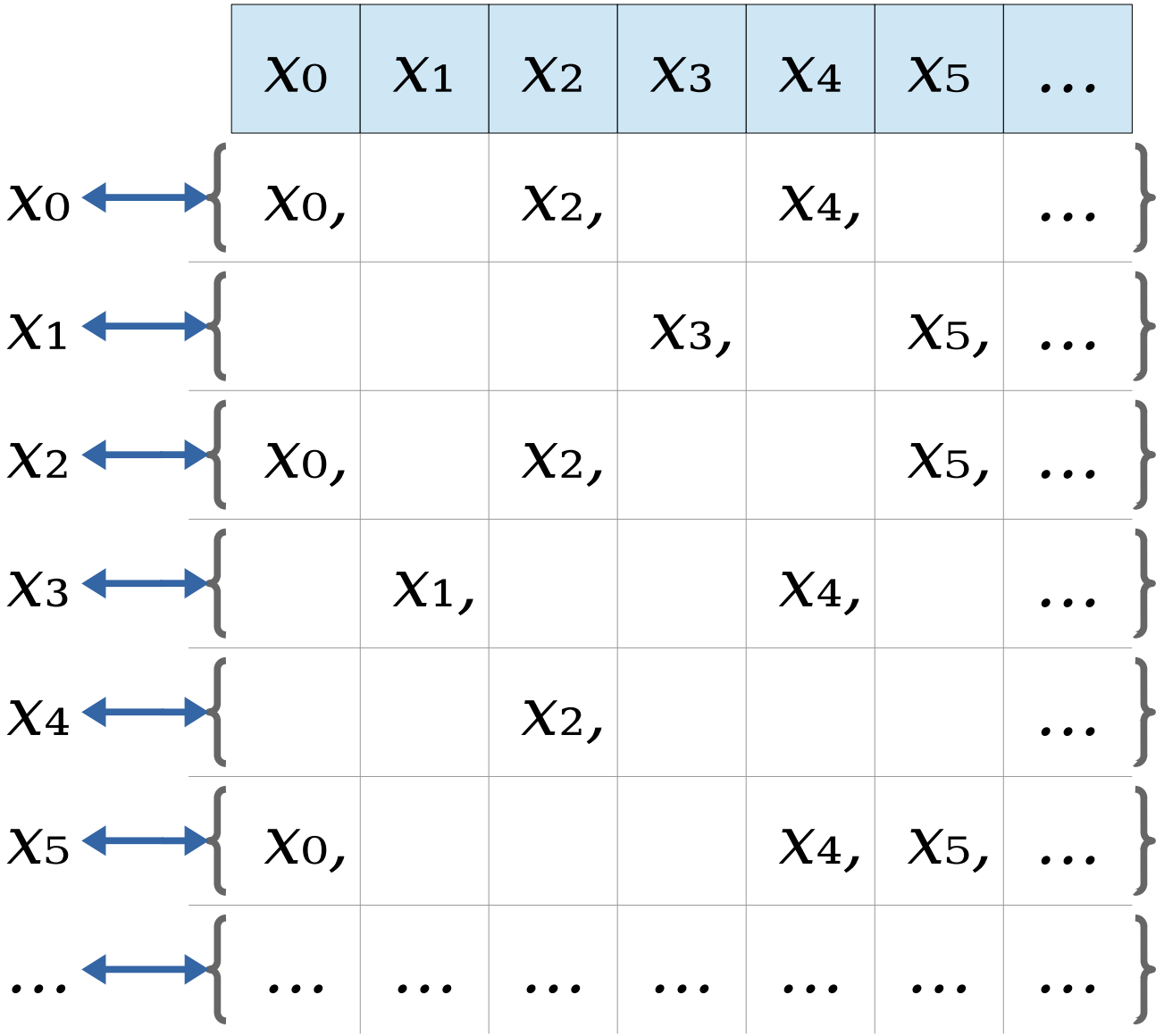
Pair nonnegative integers with even natural numbers.
Pair negative integers with odd natural numbers.

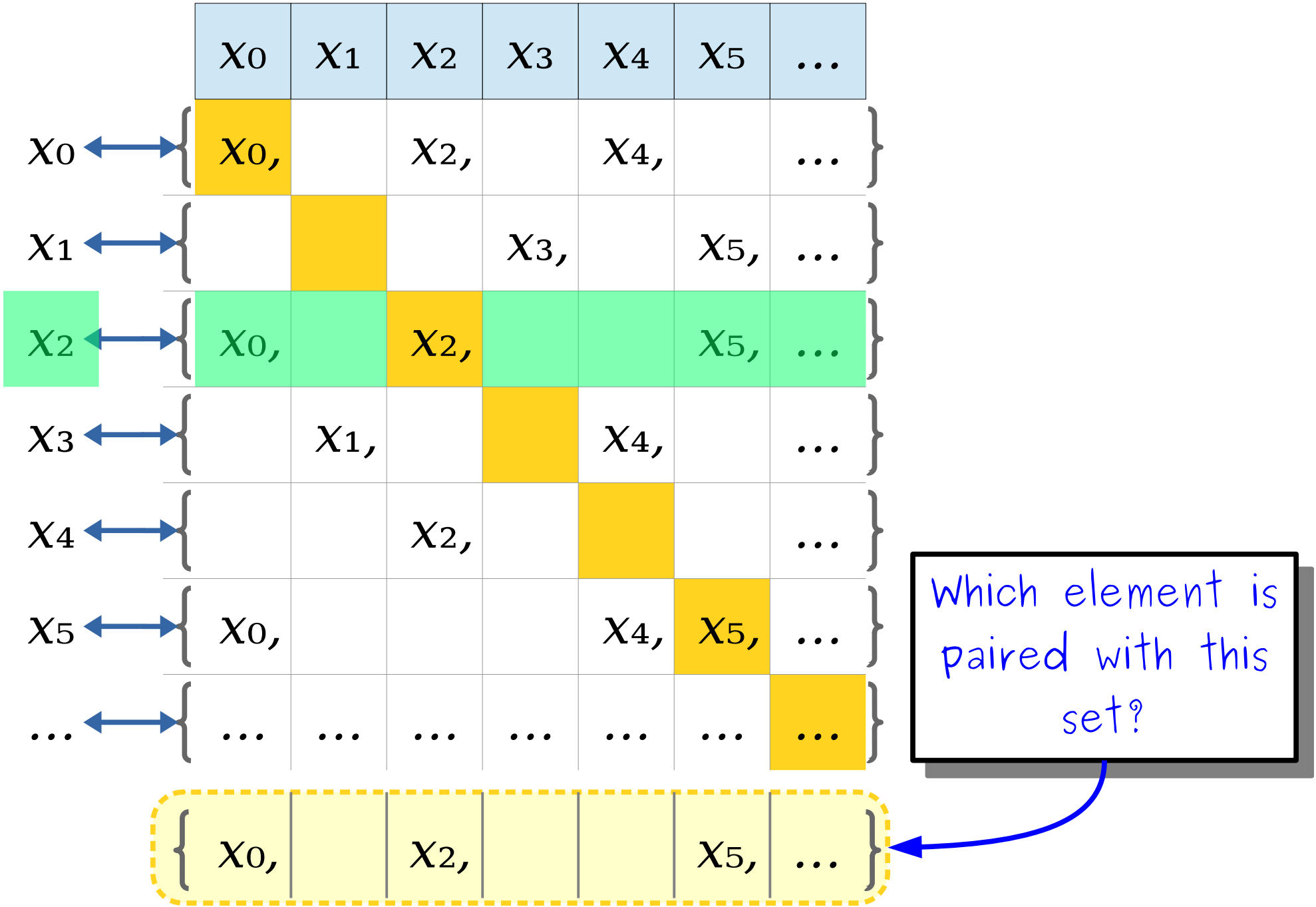
A Beautiful Result: ***Cantor's Theorem***

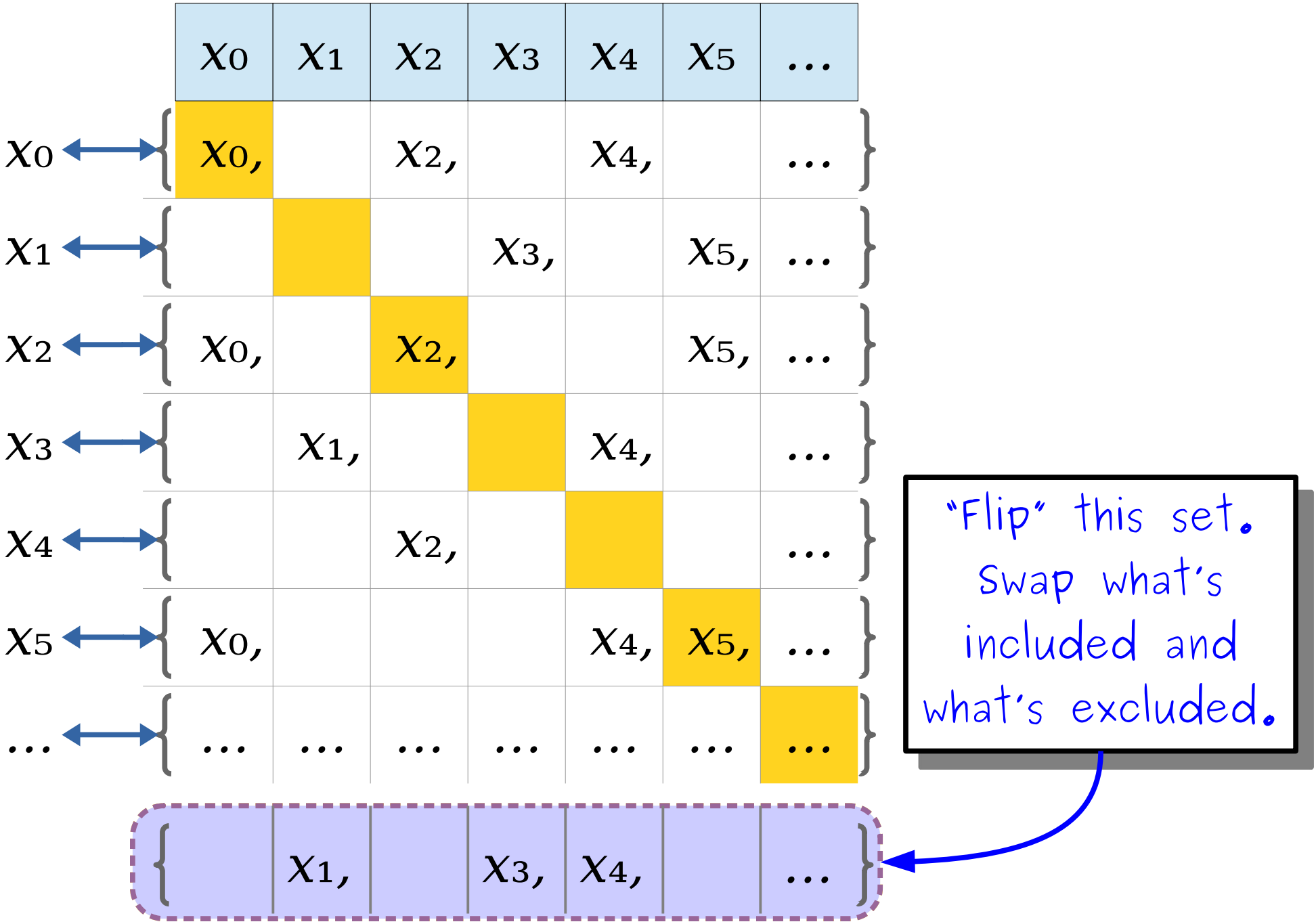
Cantor's Theorem: If S is a set, then $|S| < |\wp(S)|$.

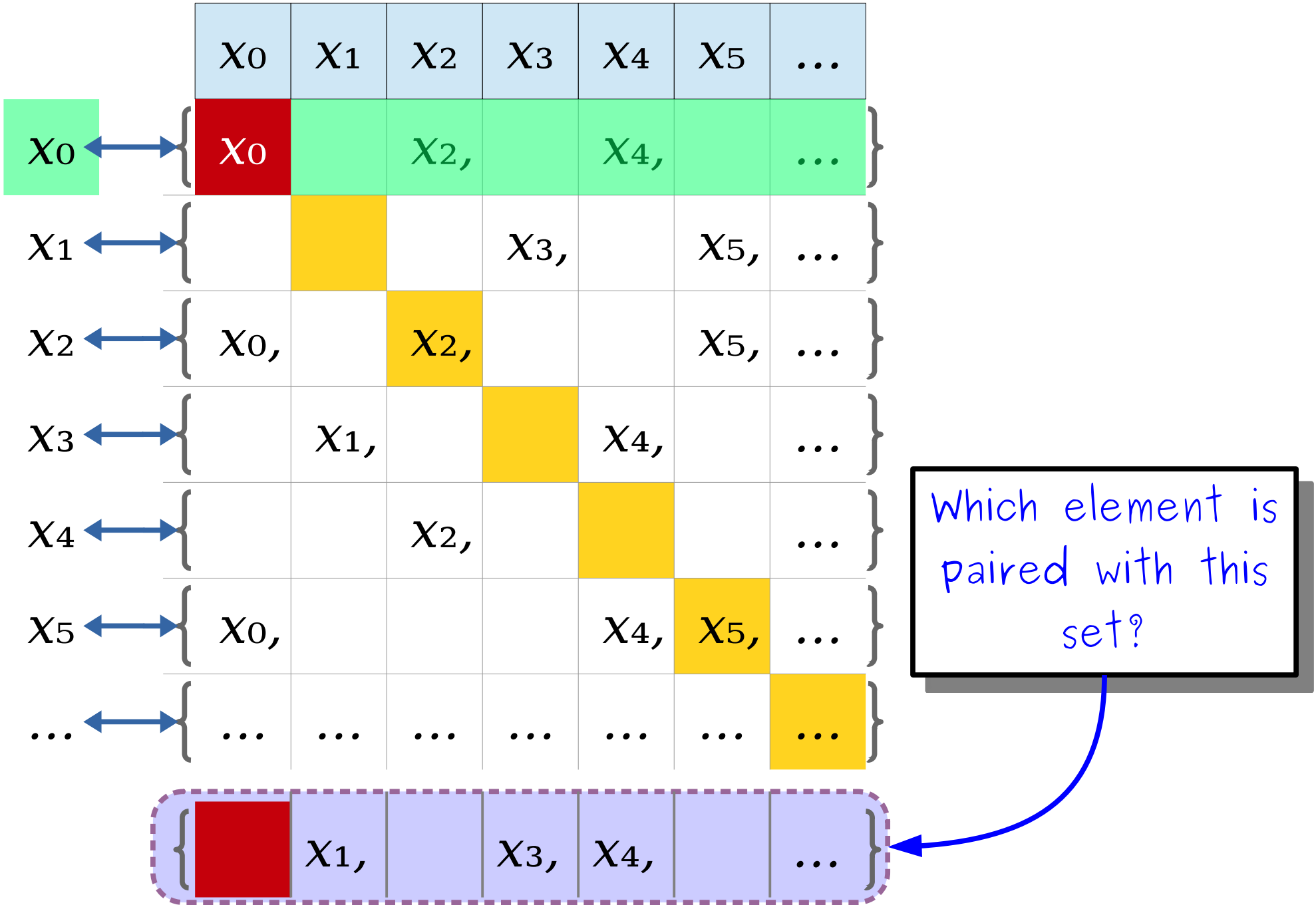
Stated differently: no matter how you pair off the elements of a set S with the subsets of S, there is always some subset of S left uncovered.


$x_0 \leftrightarrow \{ x_0, x_2, x_4, \ldots \}$
$x_1 \leftrightarrow \{ x_3, x_5, \ldots \}$
$x_2 \leftrightarrow \{ x_0, x_1, x_2, x_5, \ldots \}$
$x_3 \leftrightarrow \{ x_1, x_4, \ldots \}$
$x_4 \leftrightarrow \{ x_2, \ldots \}$
$x_5 \leftrightarrow \{ x_0, x_4, x_5, \ldots \}$
$\ldots \leftrightarrow \{ \ldots \}$


x_0 x_1 x_2 x_3 x_4 x_5 ...
x_0 x_0, x_2, x_4, ...
x_1 x_3, x_5, ...
x_2 x_0, x_2, x_5, ...
x_3 x_1, x_4, ...
x_4 x_2, ...
x_5 x_0, x_4, x_5, ...
... ...


x_0 x_1 x_2 x_3 x_4 x_5 …
x_0 x_0, x_2, x_4, …
x_1 x_3, x_5, …
x_2 x_0, x_2, x_5, …
x_3 x_1, x_4, …
x_4 x_2, …
x_5 x_0, x_4, x_5, …
… … … … … … … …
x_0, x_2, x_5, …
Which element is paired with this set?


x0 x1 x2 x3 x4 x5 ...
x0 x0, x2, x4, ...
x1 x1, x3, x5, ...
x2 x0, x2, x5, ...
x3 x1, x4, ...
x4 x2, ...
x5 x0, x4, x5, ...
... ...
{ x1, x3, x4, ... }
"Flip" this set. Swap what's included and what's excluded.


x0 x1 x2 x3 x4 x5 ...
x0
x0 x2, x4, ...
x1
x3, x5, ...
x2
x0, x2, x5, ...
x3
x1, x4, ...
x4
x2, ...
x5
x0, x4, x5, ...
...
... ...
x1, x3, x4, ...
Which element is paired with this set?

…And Beyond!

- By Cantor's Theorem:

$$|\mathbb{N}| < |\wp(\mathbb{N})|$$
$$|\wp(\mathbb{N})| < |\wp(\wp(\mathbb{N}))|$$
$$|\wp(\wp(\mathbb{N}))| < |\wp(\wp(\wp(\mathbb{N})))|$$
$$|\wp(\wp(\wp(\mathbb{N})))| < |\wp(\wp(\wp(\wp(\mathbb{N}))))|$$
$$\ldots$$

- ***Not all infinite sets have the same size!***
- ***There is no biggest infinity!***
- ***There are infinitely many infinities!***

How does this have any practical consequences?

What does this have to do with computation?

“The set of all computer programs”

“The set of all problems to solve”

Every computer program is a string.

So, the number of programs is at most the number of strings.

From Cantor's Theorem, we know that there are more sets of strings than strings.

There are at least as many problems as there are sets of strings *(see appendix!)*.

$$|\text{Programs}| \leq |\text{Strings}| < |\wp(\text{Strings})| \leq |\text{Problems}|$$

There are more problems to solve than there are programs to solve them.

|Programs| < |Problems|

It Gets Worse

- Using more advanced set theory, we can show that there are *infinitely more* problems than solutions.
- In fact, if you pick a totally random problem, the probability that you can solve it is *zero*.
- ***More troubling fact:*** We've just shown that *some* problems are impossible to solve with computers, but we don't know *which* problems those are!

We need to develop a more nuanced understanding of computation.

Where We're Going

- ***What makes a problem impossible to solve with computers?***
  - Is there a deep reason why certain problems can't be solved with computers, or is it completely arbitrary?
  - How do you know when you're looking at an impossible problem?
  - Are these real-world problems, or are they highly contrived?
- ***How do we know that we're right?***
  - How can we back up our pictures with rigorous proofs?
  - How do we build a mathematical framework for studying computation?

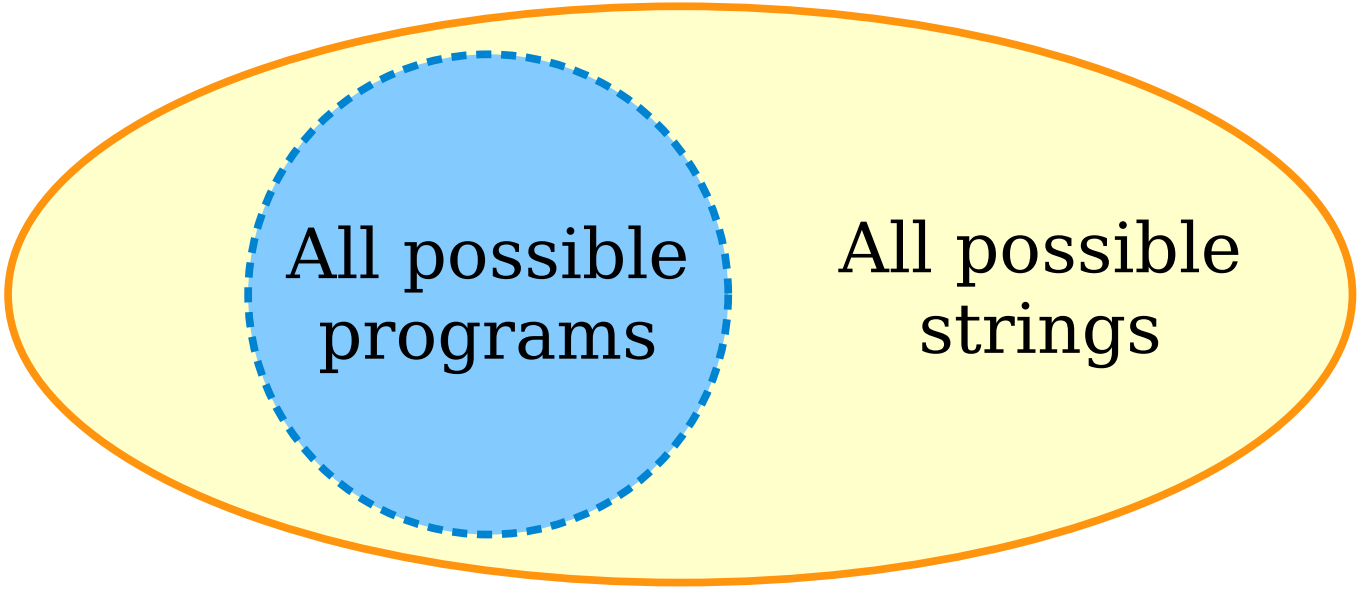
Next Time

- ***Mathematical Proof***
  - What is a mathematical proof?
  - How can we prove things with certainty?

Appendix: Stringy Thingies

Strings and Programs

- The source code of a computer program is just a (long, structured, well-commented) string of text.
- All programs are strings, but not all strings are necessarily programs.



|Programs| ≤ |Strings|

Strings and Problems

- There is a connection between the number of sets of strings and the number of problems to solve.
- Let S be any set of strings. This set S gives rise to a problem to solve:

Given a string w, determine whether $w \in S$.

Strings and Problems

Given a string *w*, determine whether *w* ∈ *S*.

- Suppose that S is the set

$$S = \{ \text{"}a\text{"}, \text{"}b\text{"}, \text{"}c\text{"}, \ldots, \text{"}z\text{"} \}$$

- From this set S, we get this problem:

Given a string *w*, determine whether *w* is a single lower-case English letter.

Strings and Problems

Given a string *w*, determine whether *w* ∈ *S*.

- Suppose that S is the set

  S = { "0", "1", "2", …, "9", "10", "11", … }

- From this set S, we get this problem:

Given a string *w*, determine whether *w* represents a natural number.

Strings and Problems

Given a string *w*, determine whether *w* ∈ *S*.

- Suppose that S is the set

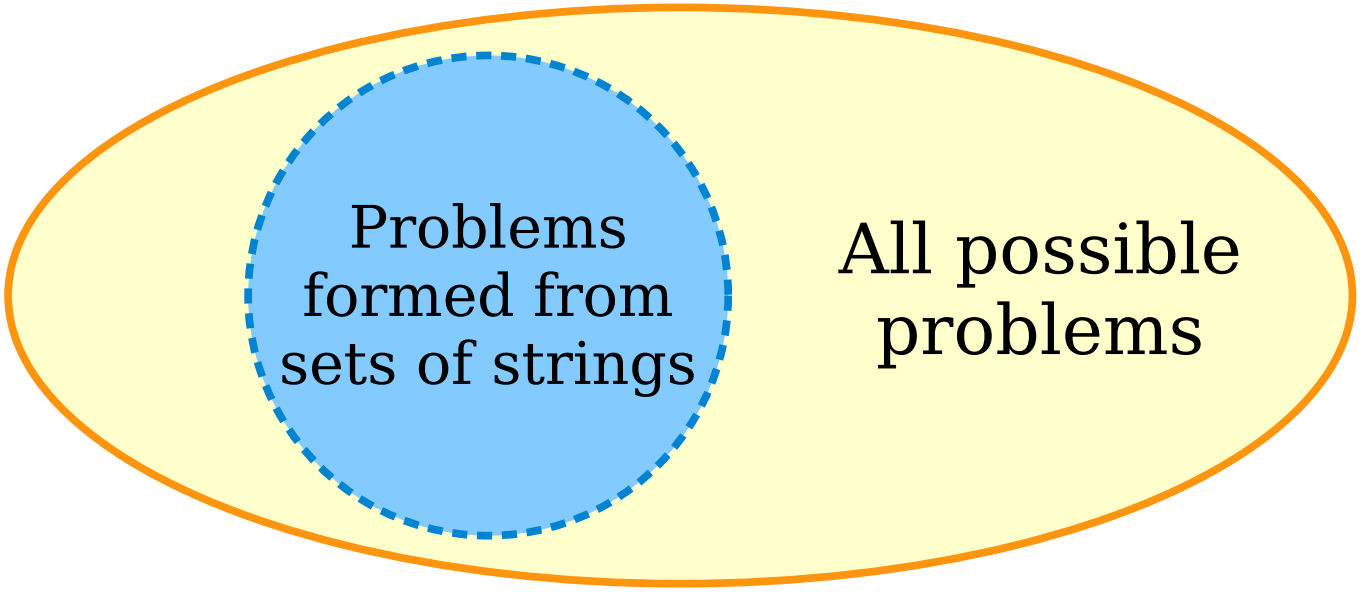
$$S = \{ p \mid p \text{ is a legal C++ program} \}$$

- From this set S, we get this problem:

Given a string *w*, determine whether *w* is a legal C++ program.

Strings and Problems

- Every set of strings gives rise to a unique problem to solve.
- Other problems exist as well.



$|\wp$(Strings)| ≤ |Problems|