Welcome to CS103!

- ***Handouts***
  - Course Information
  - Syllabus
- ***Today:***
  - Course Overview
  - Introduction to Set Theory
  - The Limits of Computation
  - *Problem Set 0 will be posted later today*

Are there “laws of physics”
in computer science?

Key Questions in CS103

- What problems can you solve with a computer?
  - ***Computability Theory***
- Why are some problems harder to solve than others?
  - ***Complexity Theory***
- How can we be certain in our answers to these questions?
  - ***Discrete Mathematics***

Instructors

Cynthia Lee (cbl@cs.stanford.edu)

TAs

Joshua Kravitz
David Liang
John Melloni
Teresa Noyola
Divya Saini
Magdy Saleh
Ryan Smith
Pranav Sriram
Hugo Valdivia

Course Website

https://cs103.stanford.edu

Prerequisite / Corequisite

CS106B

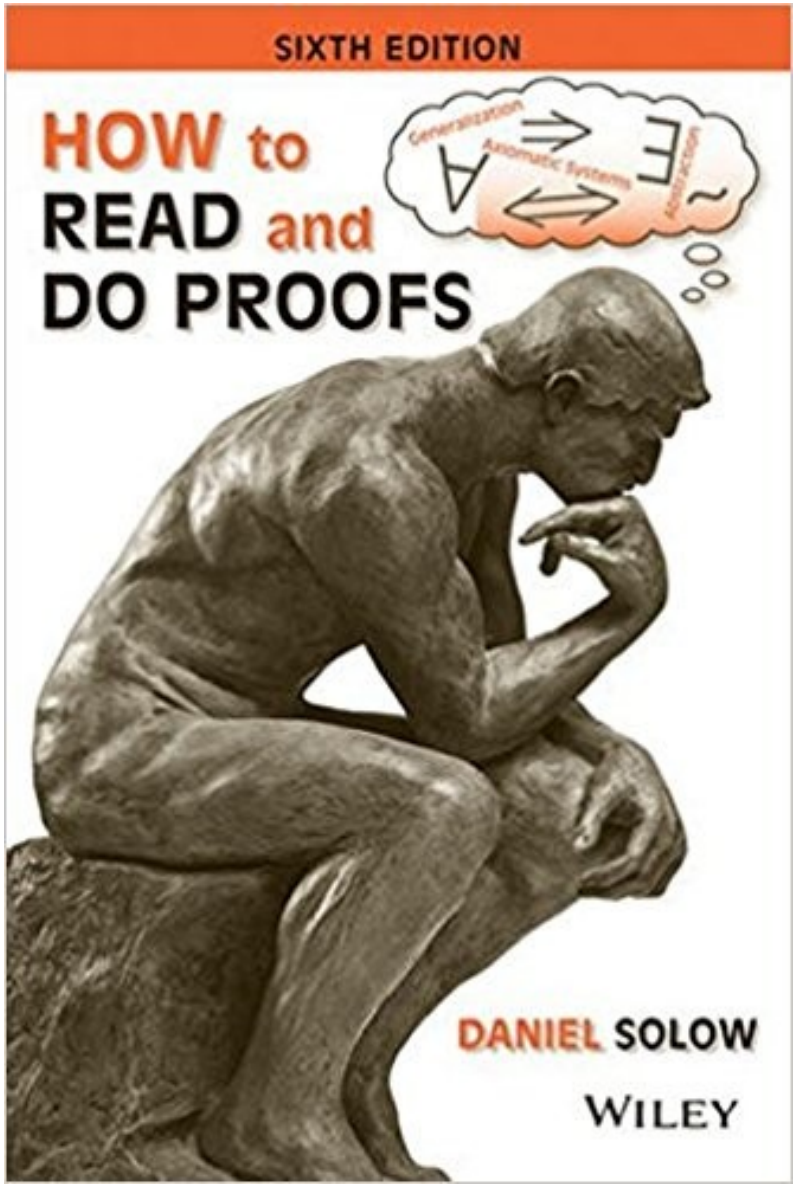
We have some programming assignments and will be referencing some concepts from CS106B/X throughout the quarter.

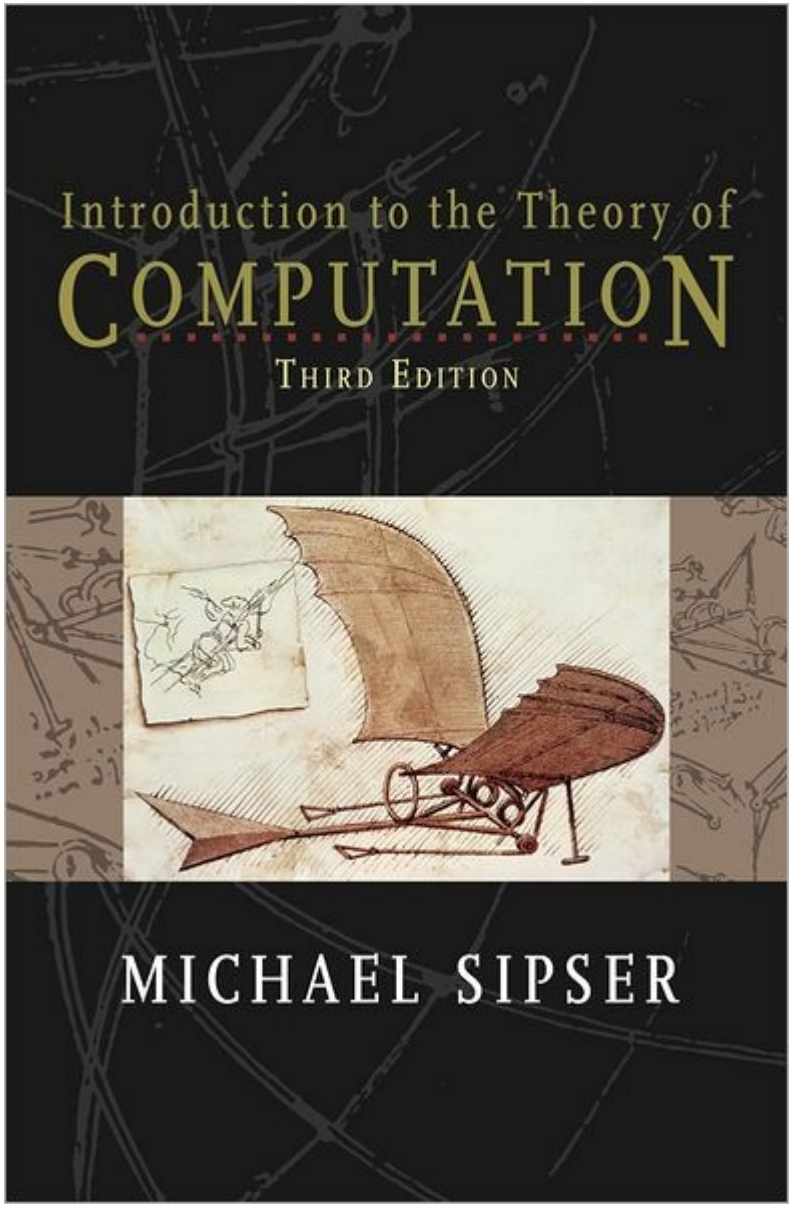
There aren't any math prerequisites for this course – high-school algebra should be enough!

Problem Set 0

- Your first assignment, Problem Set 0, goes out today. Its' due Friday at 2:30PM.
  - There's no actual coding involved. This is just to ensure that you get your coding environment set up and know how to submit your work.

Recommended Reading





Grading

Grading

Problem Sets

Ten Problem Sets

Problem sets (other than PS0) may be done individually or in pairs.

Grading

Midterm I

Monday, Feb 4
7:00PM – 10:00PM

Midterm II

Monday, Feb 25
7:00PM – 10:00PM

Grading

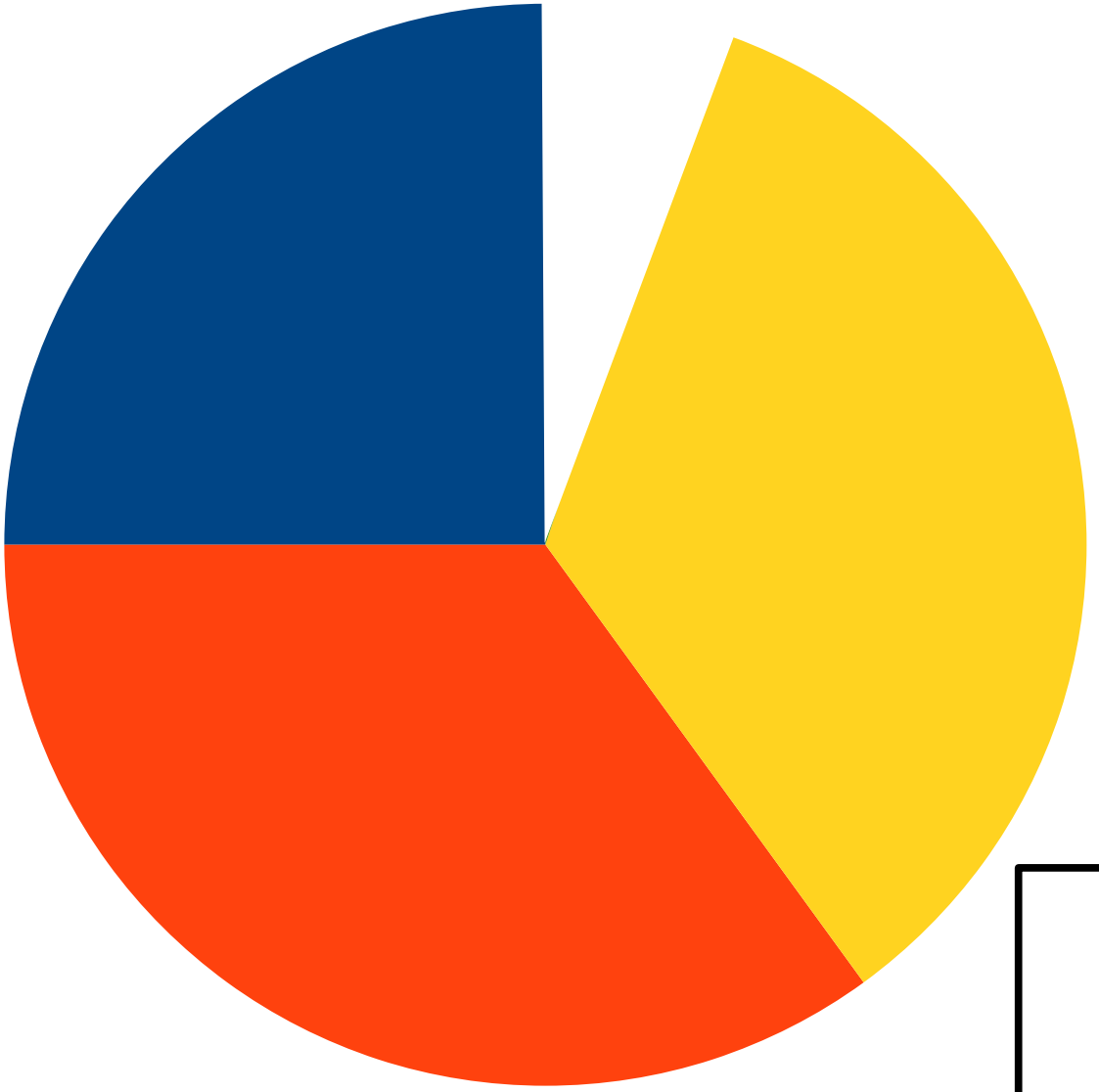
Problem Sets

Midterms

Midterm Score

$\frac{2}{3} \cdot \text{Higher} + \frac{1}{3} \cdot \text{Lower}$

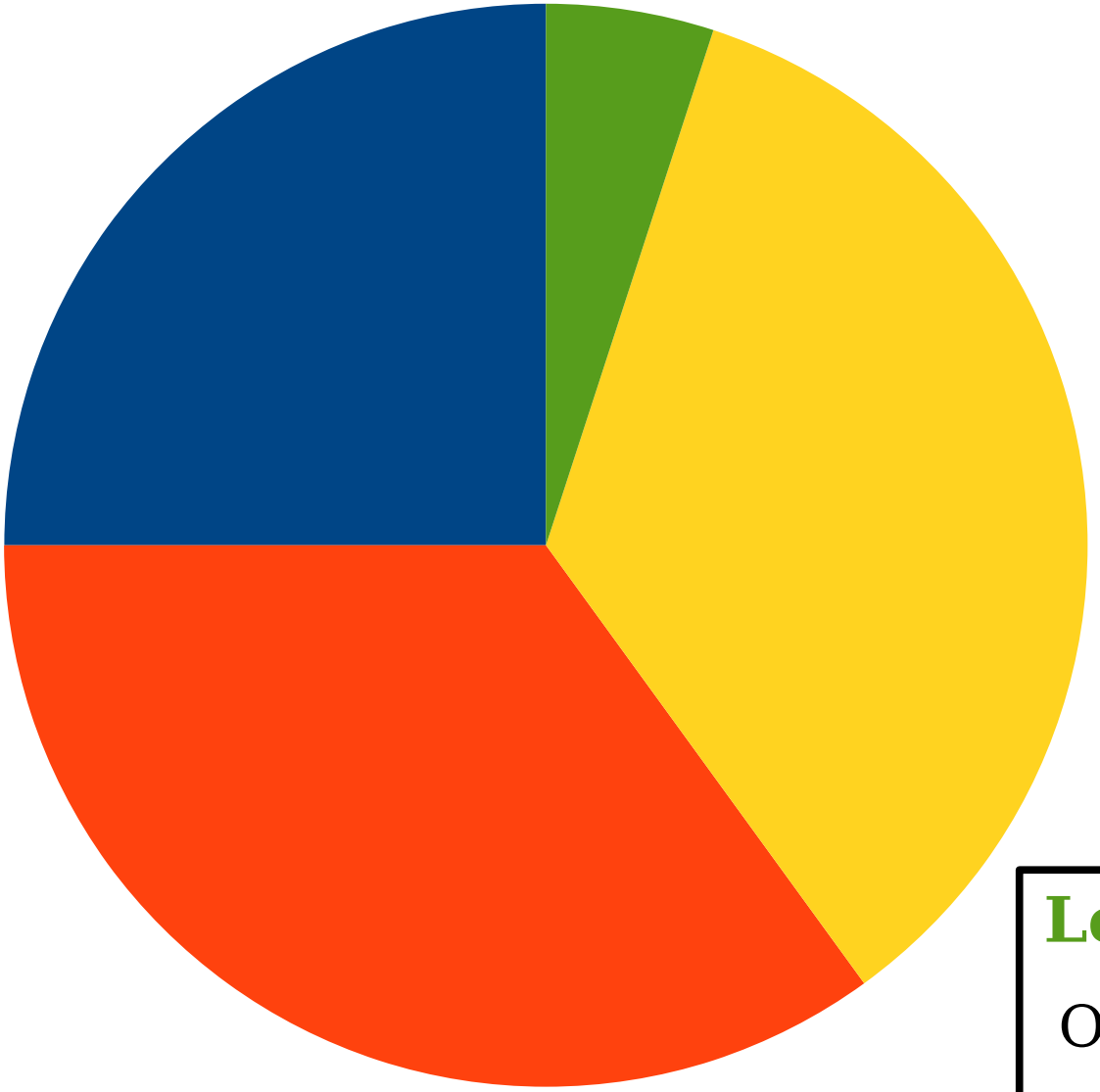
Grading



Final Exam

Monday, March 18
3:30PM – 6:30PM

Grading



- Problem Sets
- Midterms
- Final Exam
- Participation

Lecture Participation

Optional—otherwise this 5% splits between midterm and final.

CS103A

- ***CS103A*** is an optional, one-unit add-on course to CS103.
- It provides extra review and practice with the material from CS103 and covers general problem-solving techniques useful in discrete math.
- Similar in spirit to Math 51A, Physics 41A, or Chem 31AC.
- Class meets Tuesdays 3-5pm, location TBA *(watch for announcement on course website)*.
- Tomorrow's class only: Gates 104 3:00-4:30pm.

We've got a big journey ahead of us.

Let's get started!

Introduction to Set Theory

“CS103 students”

“All the computers on the Stanford network”

“Cool people”

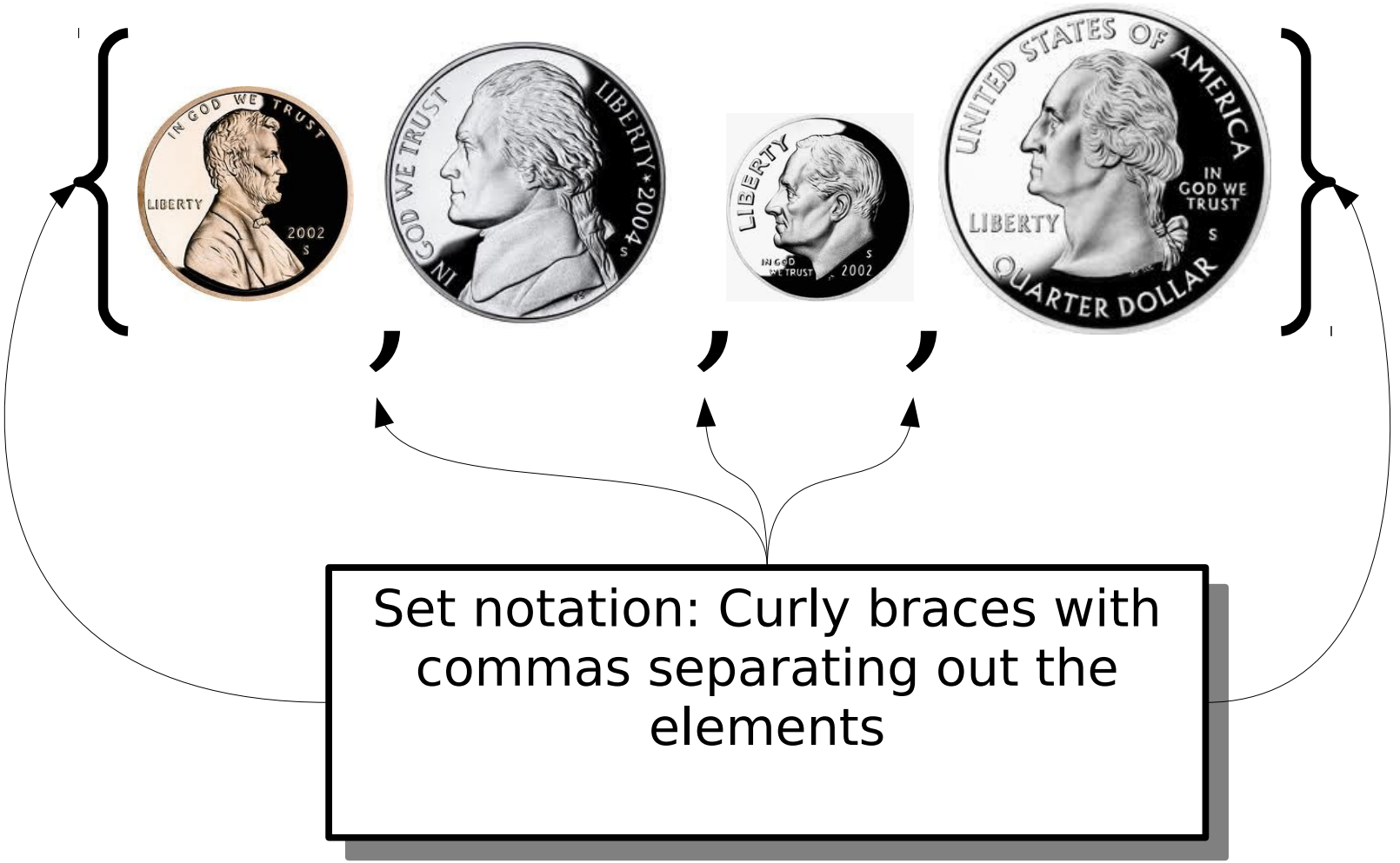
“The chemical elements”

“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).



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

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

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

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

Sets cannot contain the same object twice.
Repeated elements are ignored.

{

,

}

Sets cannot contain the same object twice.
Repeated elements are ignored.

Sets cannot contain the same object twice.
Repeated elements are ignored.

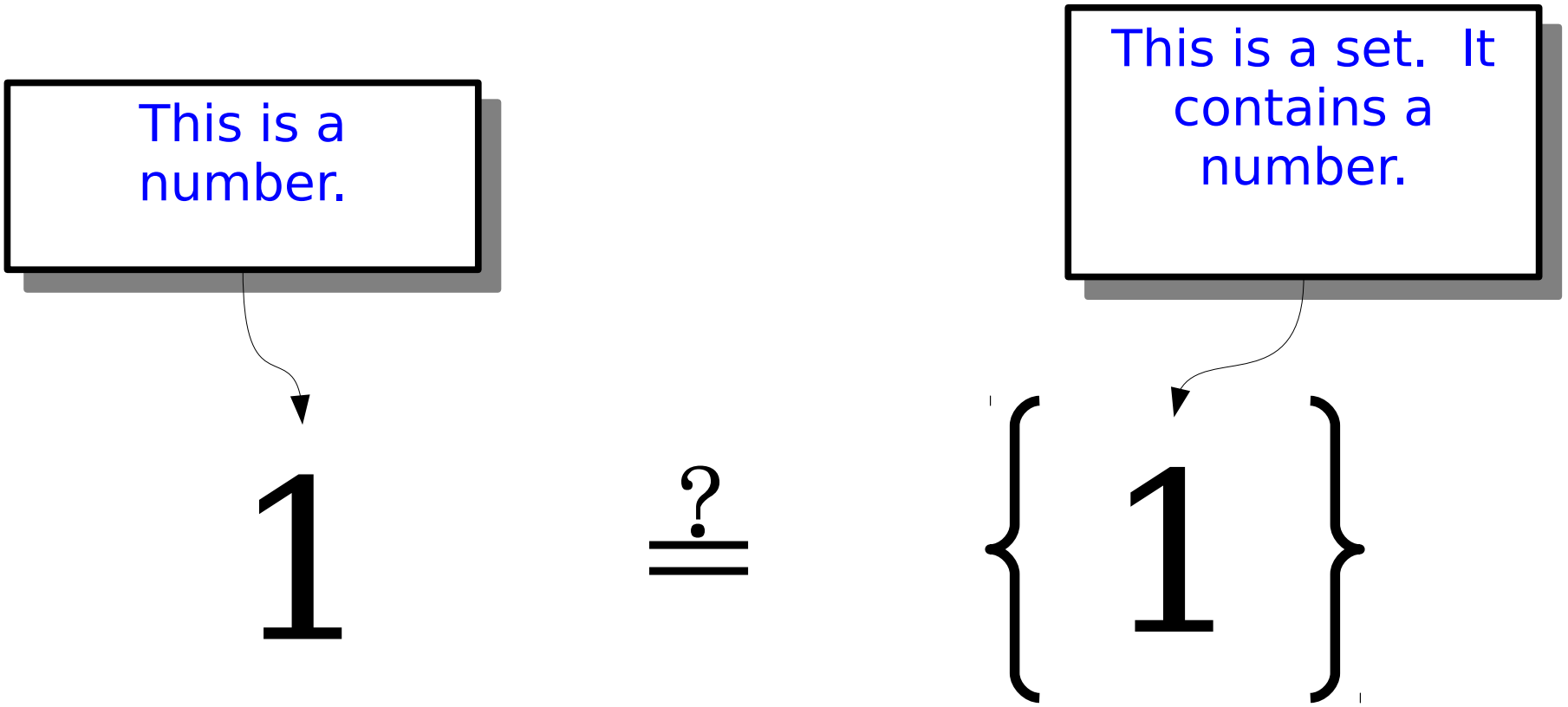
These are the same set!

Sets cannot contain the same object twice.
Repeated elements are ignored.

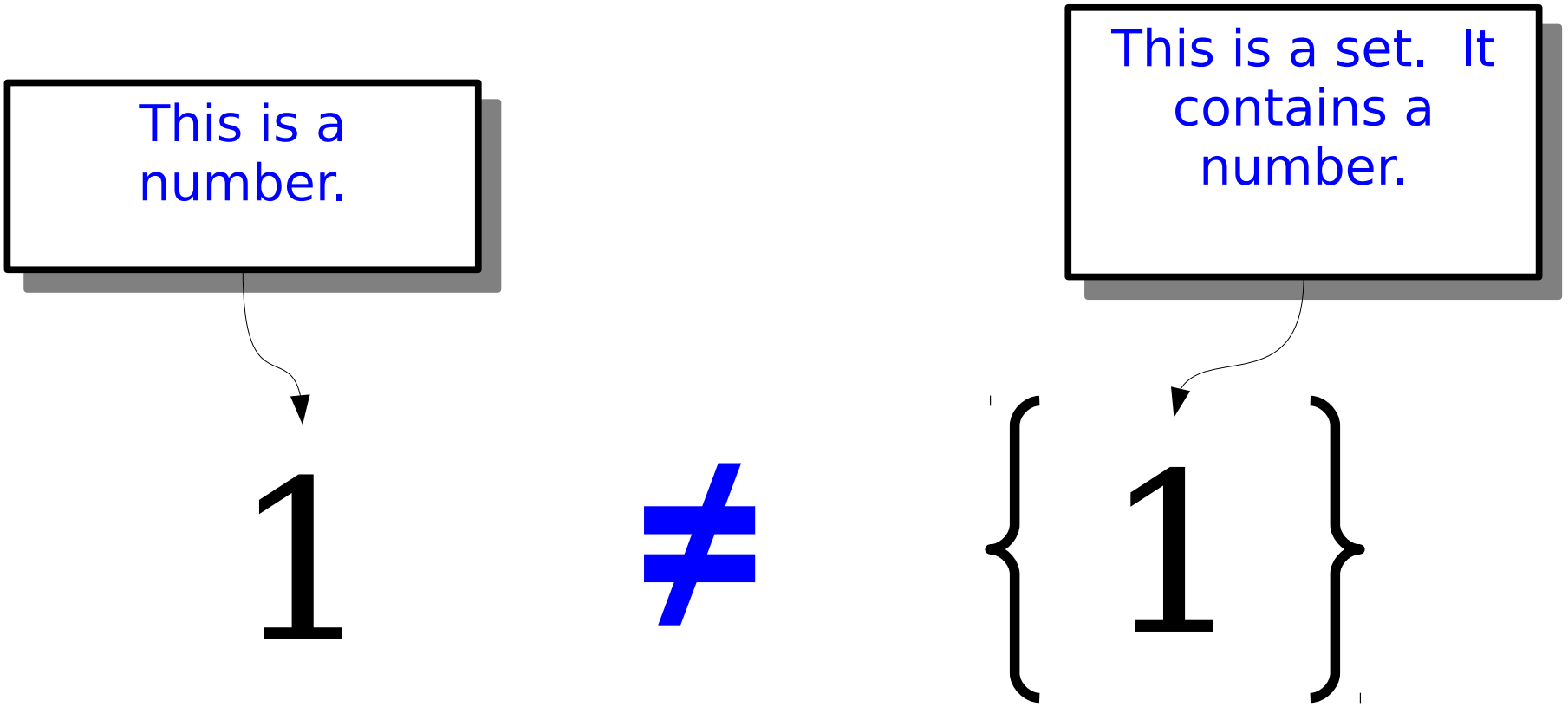
$\{\ \}$ = $\emptyset$

The **empty set** contains no elements.

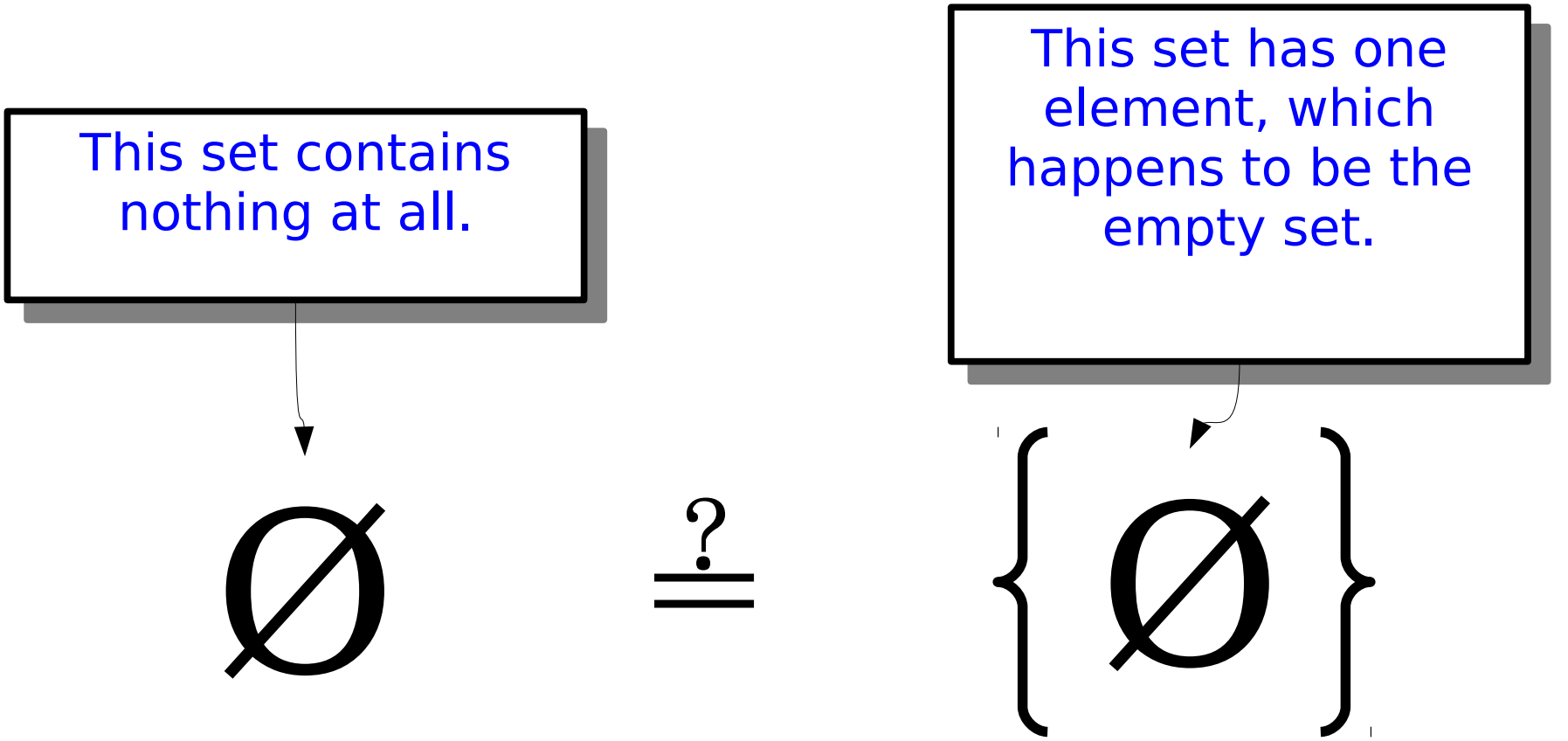
We use this symbol to denote the empty set.



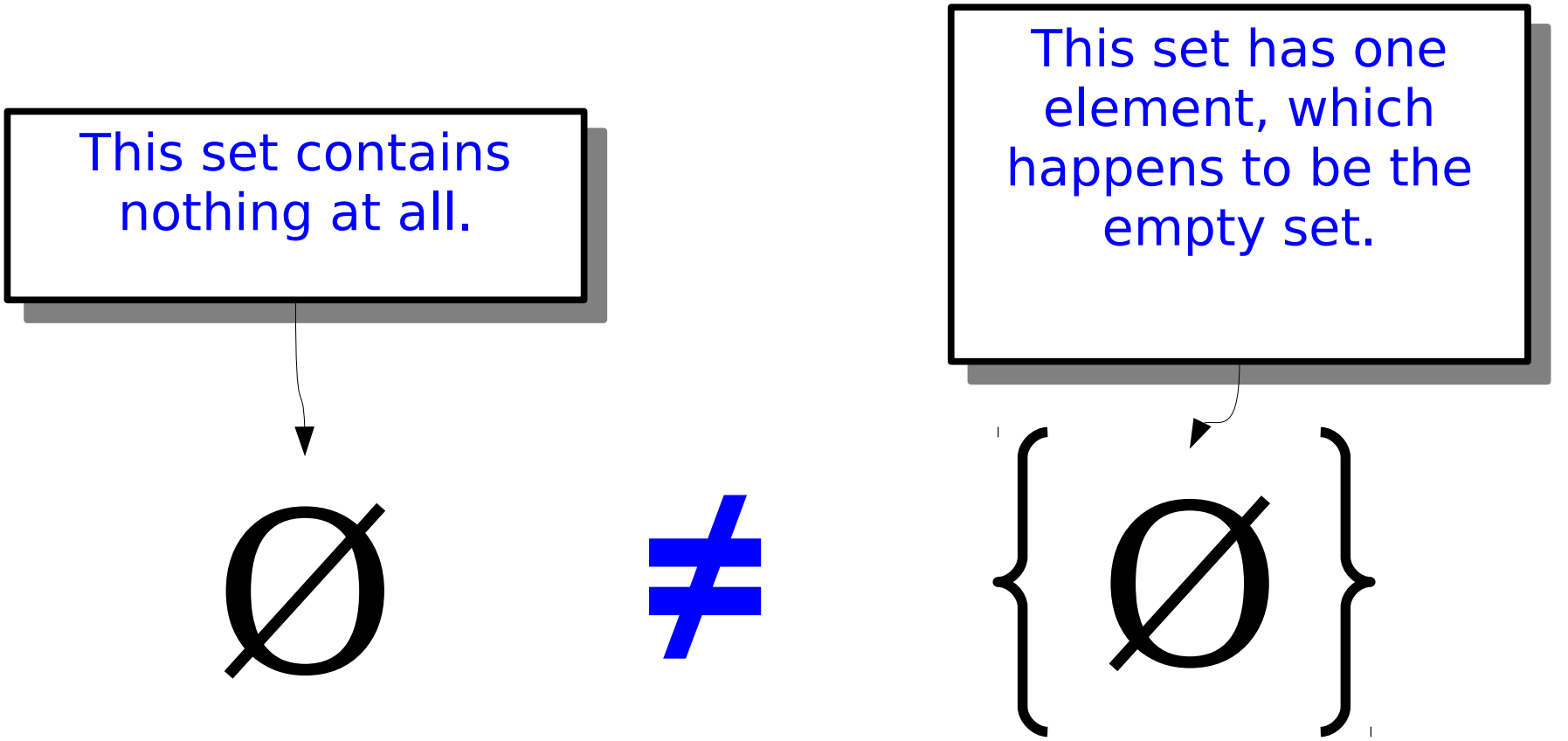
Are these objects equal to one another?



Are these objects equal to one another?



Are these objects equal to one another?



Are these objects equal to one another?

Membership

Membership

{
GOD WE TRUST
LIBERTY 2004
,
IN GOD WE TRUST
LIBERTY
2002
,
UNITED STATES OF AMERICA
IN GOD WE TRUST
LIBERTY
QUARTER DOLLAR
,
LIBERTY
2002
}

Membership

{

,

,

,

}

Is

in this set?

Membership

$\in \{$

,

,

,

$\}$

Is

in this set?

Membership

$\in$ $\{$

,

,

,

$\}$

Is

in this set?

Membership

$\notin$ $\{$

,

,

,

$\}$

Is

in this set?

Set Membership

- Given a set S and an object x, we write

  $$x \in S$$

  if x is contained in S, and

  $$x \notin S$$

  otherwise.
- If $x \in S$, we say that x is an ***element*** of S.
- Given any object x and any set S, either $x \in S$ or $x \notin S$.

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***.
  - $e \in \mathbb{R}$, $\pi \in \mathbb{R}$, $4 \in \mathbb{R}$, etc.

Describing Complex Sets

- Here are some English descriptions of infinite sets:

  “The set of all even numbers.”

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

  “The set of all negative integers.”

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

Even Natural Numbers

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

Even Natural Numbers

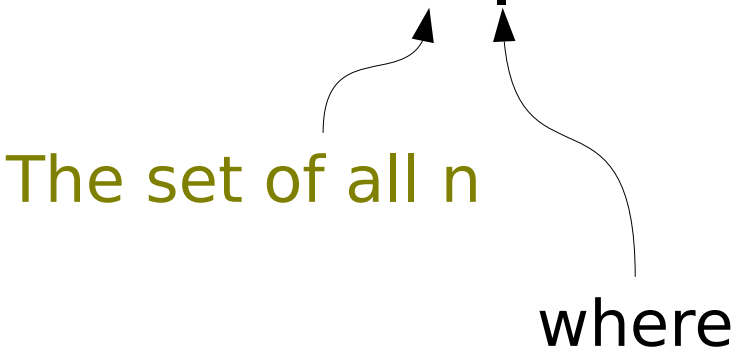
$$\{\, n \mid n \in \mathbb{N} \text{ and } n \text{ is even} \,\}$$

Even Natural Numbers

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

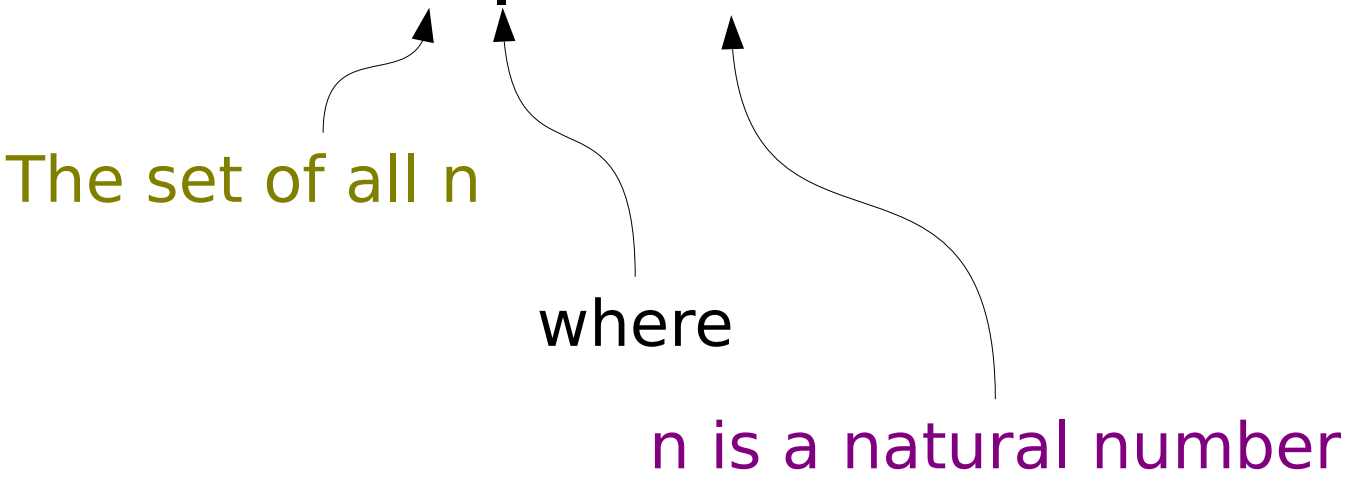
Even Natural Numbers

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

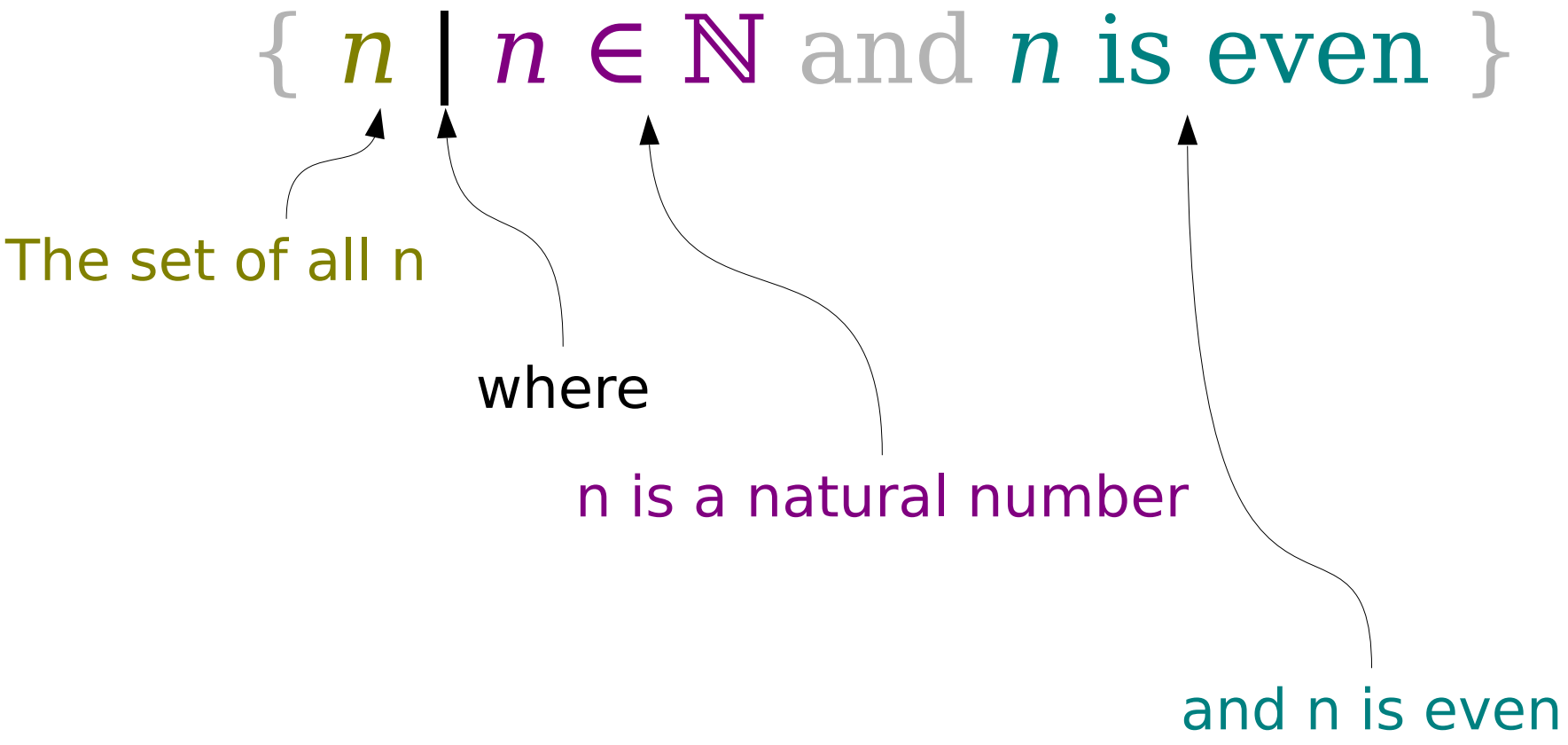


Even Natural Numbers

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

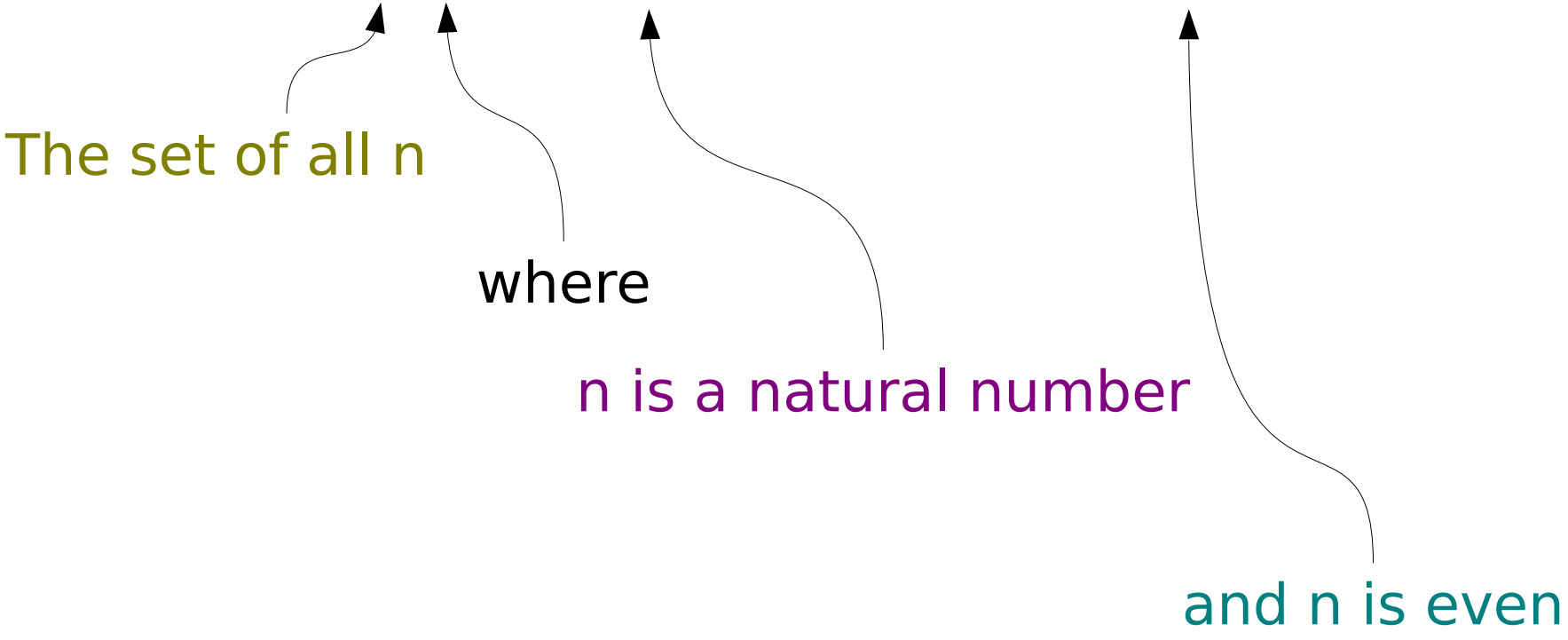


Even Natural Numbers



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***:

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

- For example:

  $\{ r \mid r \in \mathbb{R} \text{ and } r < 137 \}$

  $\{ n \mid n \text{ is an even natural number} \}$

  $\{ S \mid S \text{ is a set of US currency} \}$

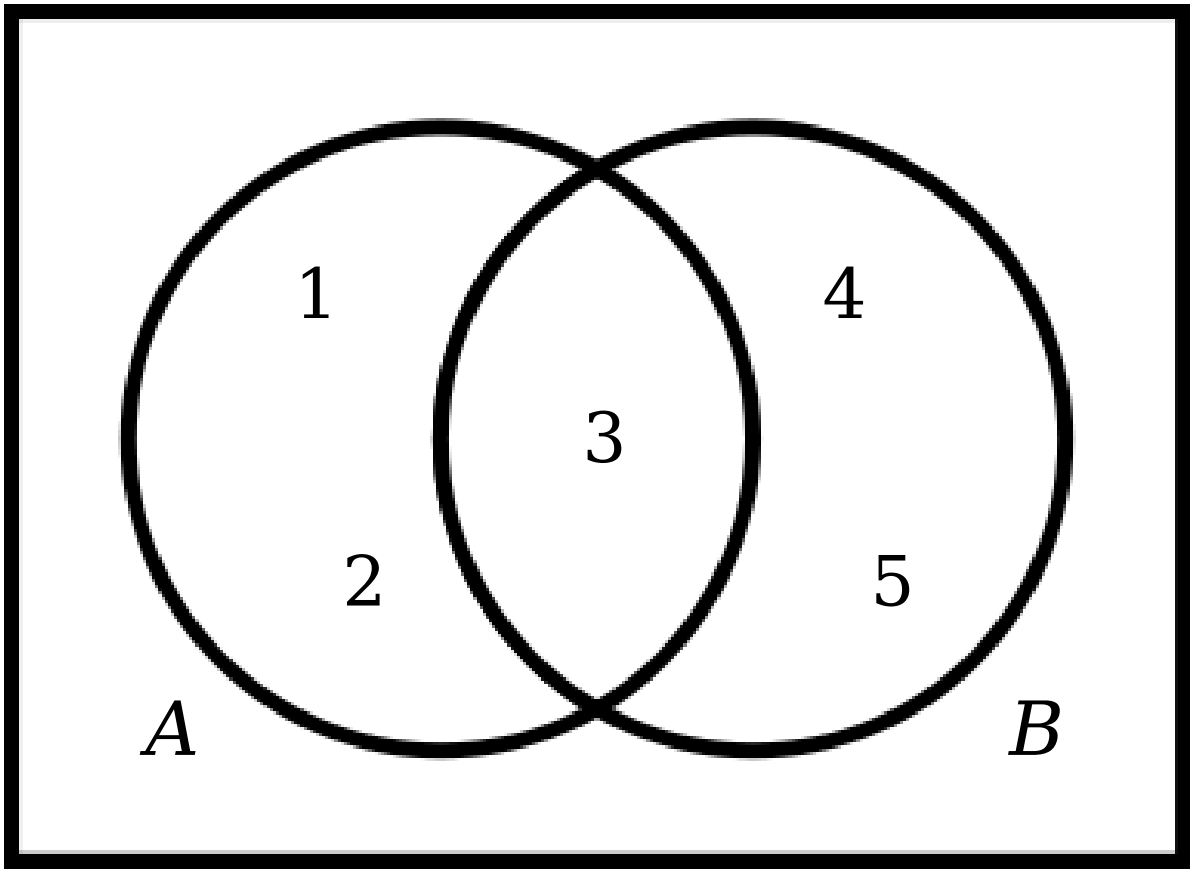
  $\{ a \mid a \text{ is cute animal} \}$

  $\{ r \in \mathbb{R} \mid r < 137 \}$

  $\{ n \in \mathbb{N} \mid n \text{ is odd} \}$

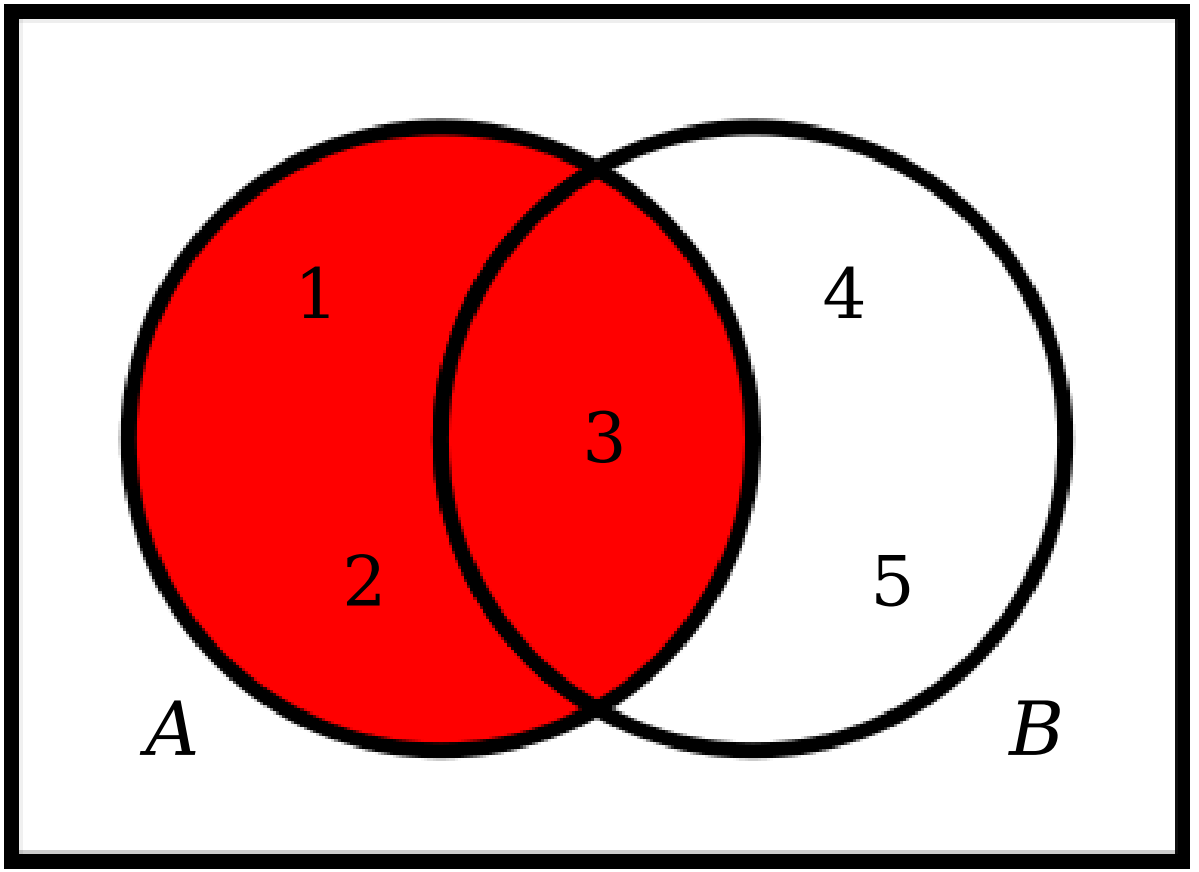
Combining Sets

Venn Diagrams



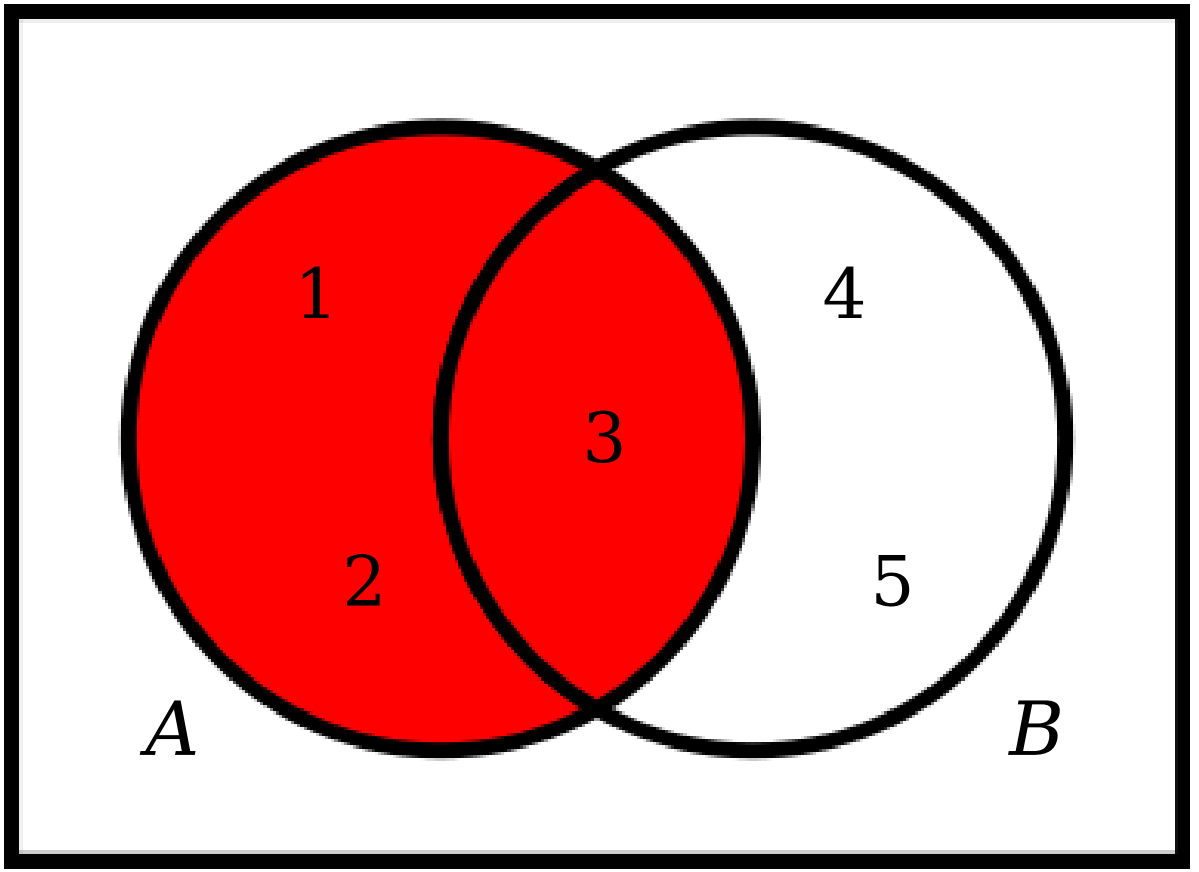
$A = \{ 1, 2, 3 \}$
$B = \{ 3, 4, 5 \}$

Venn Diagrams



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

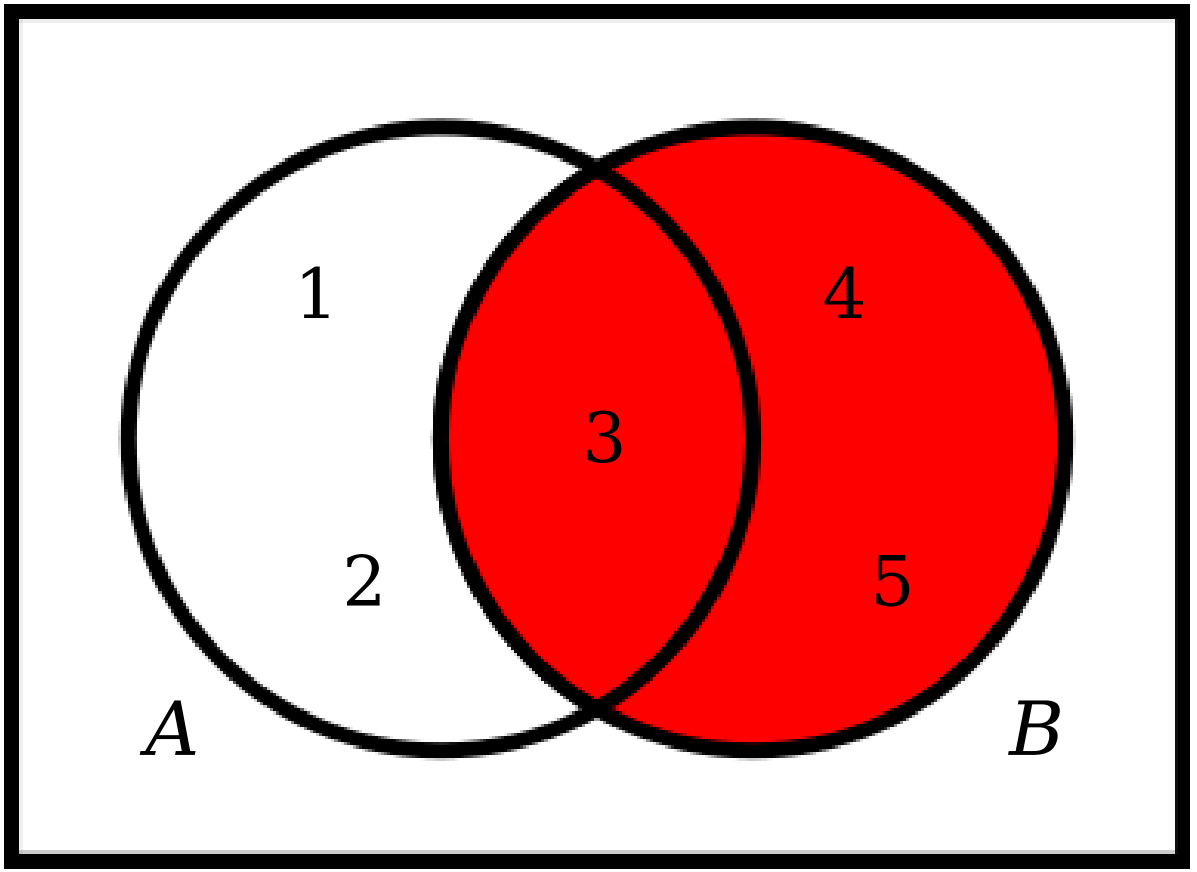
Venn Diagrams



A

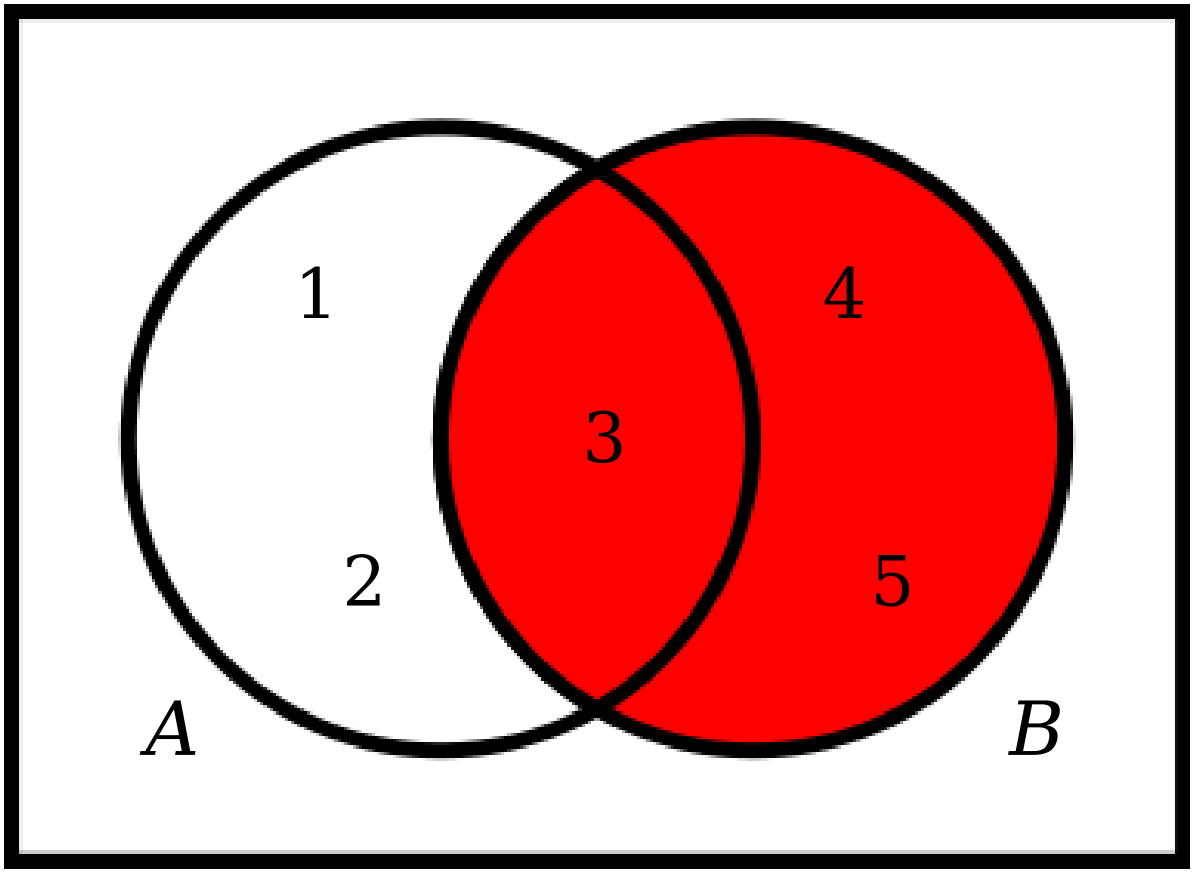
$A = \{ 1, 2, 3 \}$
$B = \{ 3, 4, 5 \}$

Venn Diagrams



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

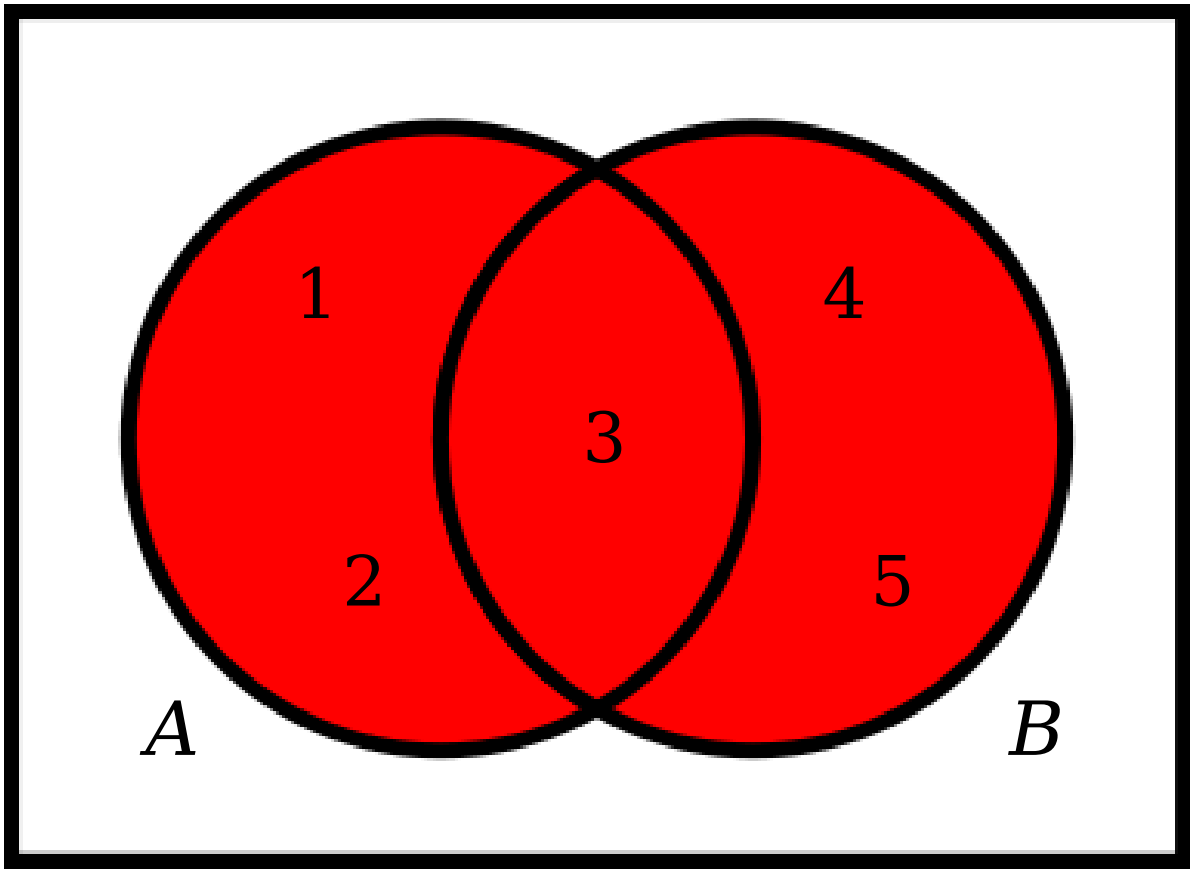
Venn Diagrams



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

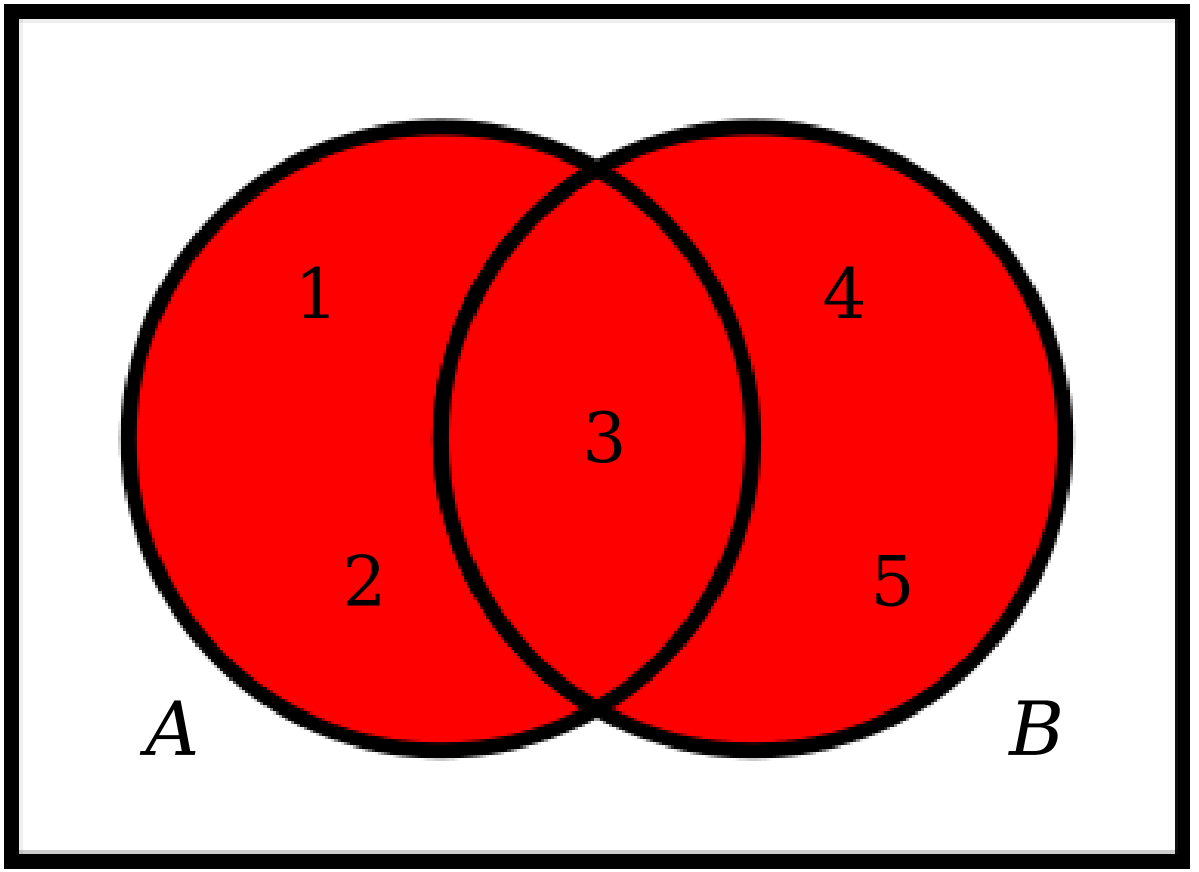
B

Venn Diagrams



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

Venn Diagrams



Union

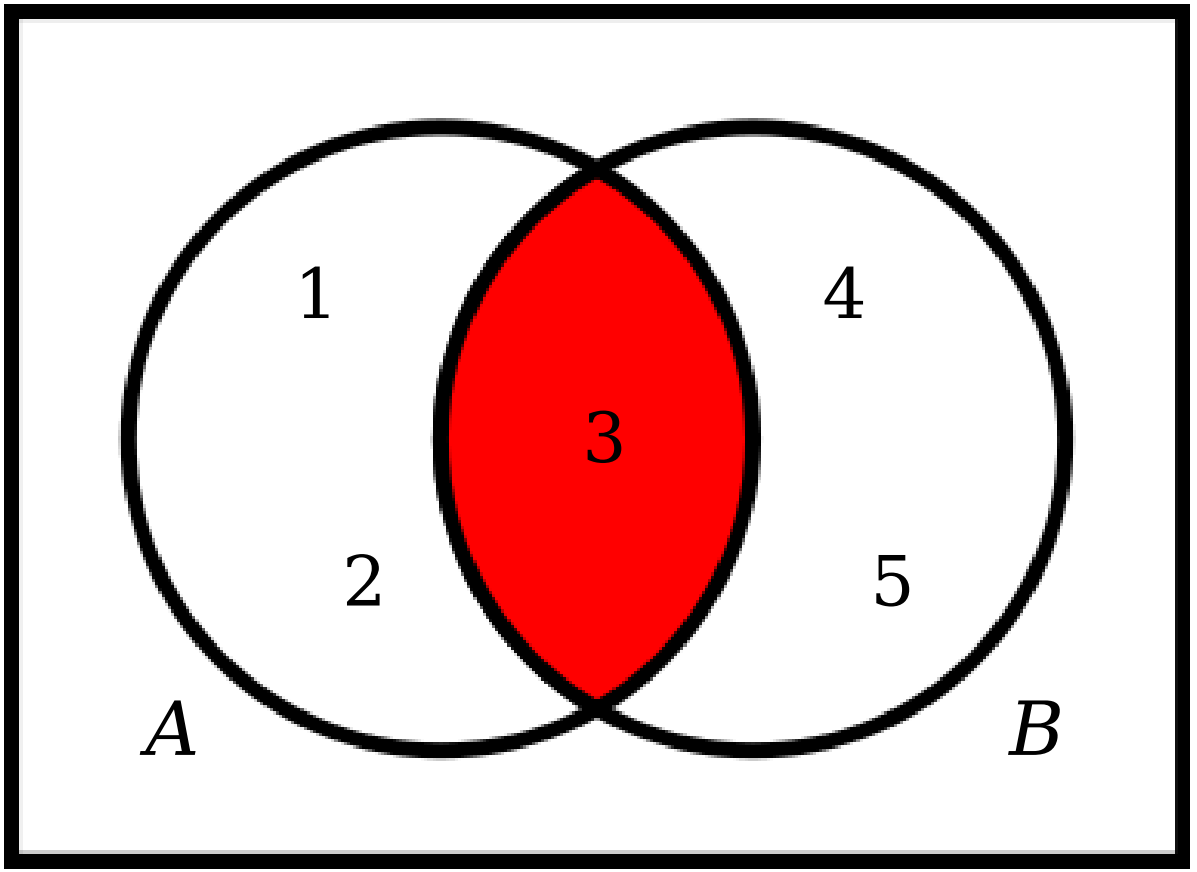
$A \cup B$

{ 1, 2, 3, 4, 5 }

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

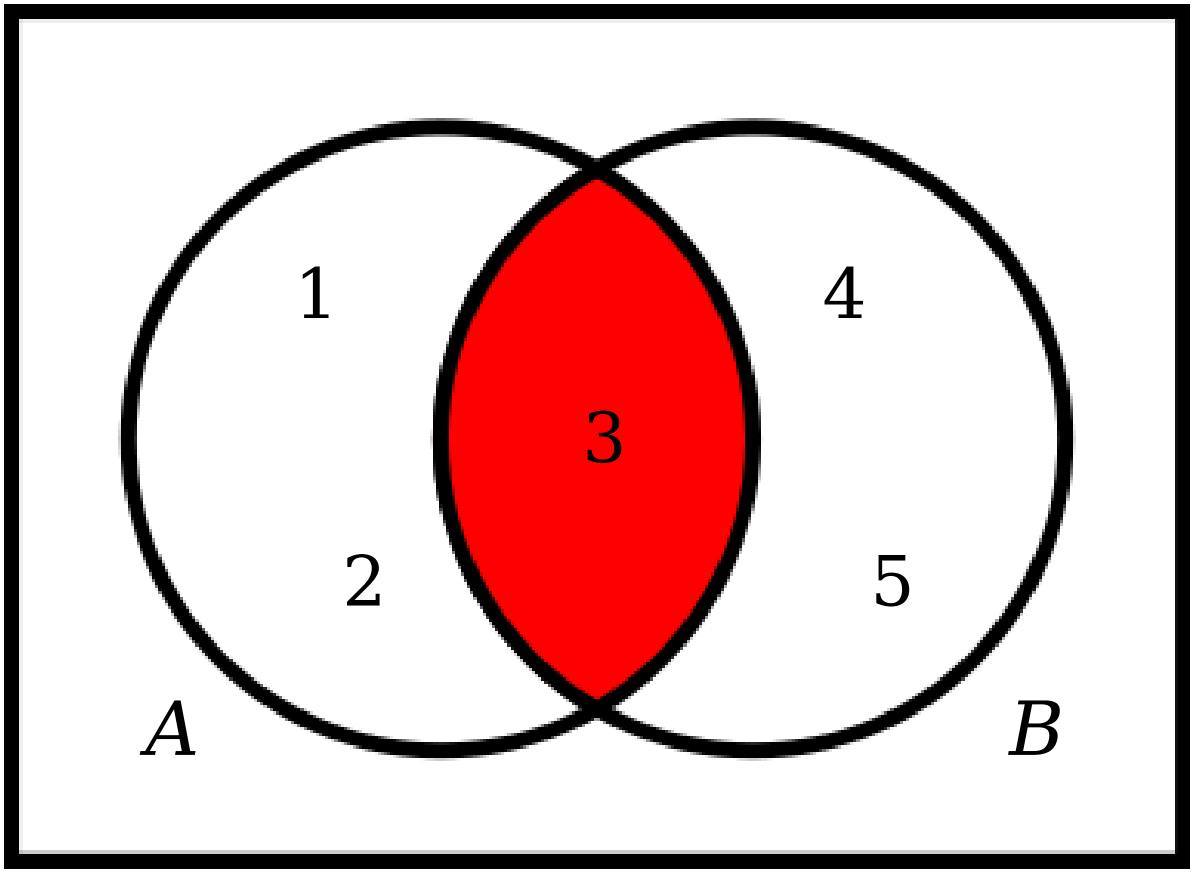
$B = \{ 3, 4, 5 \}$

Venn Diagrams



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

Venn Diagrams



Intersection

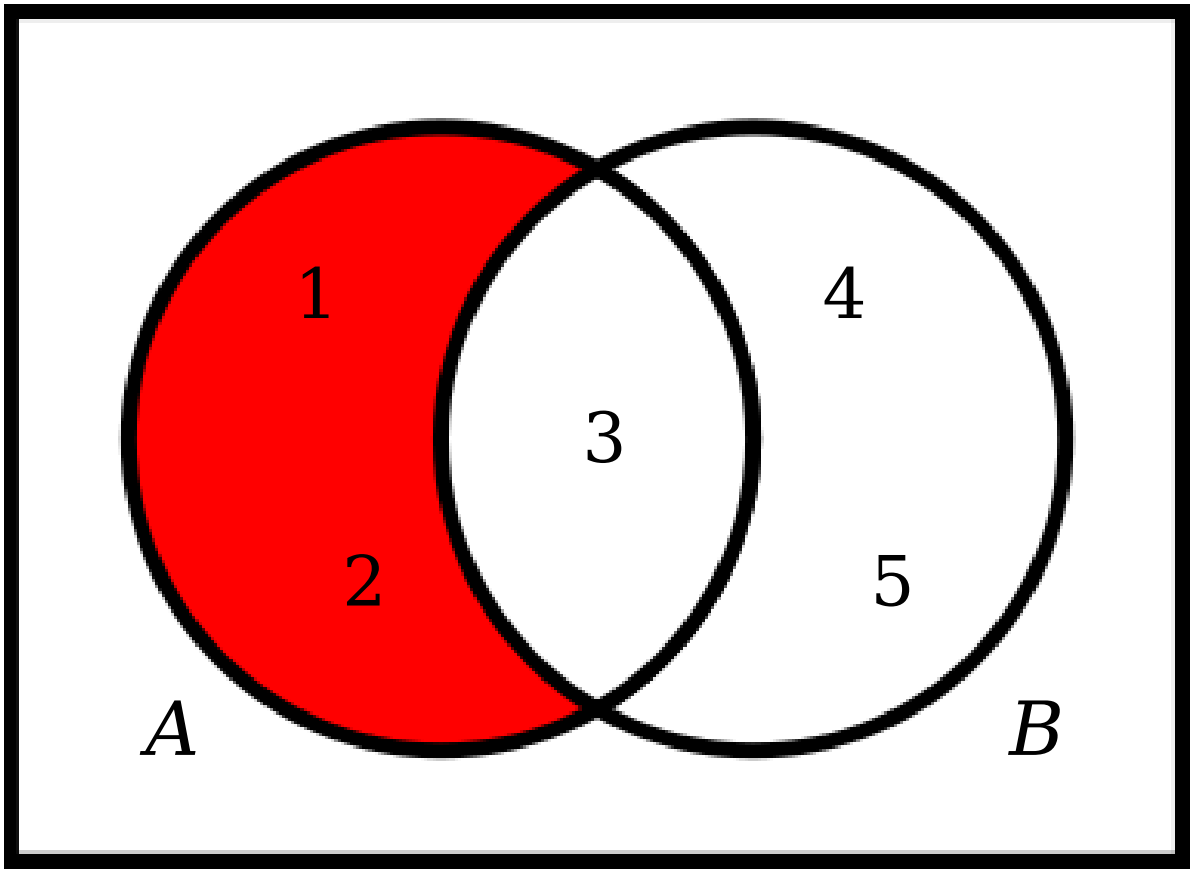
$A \cap B$

{ 3 }

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

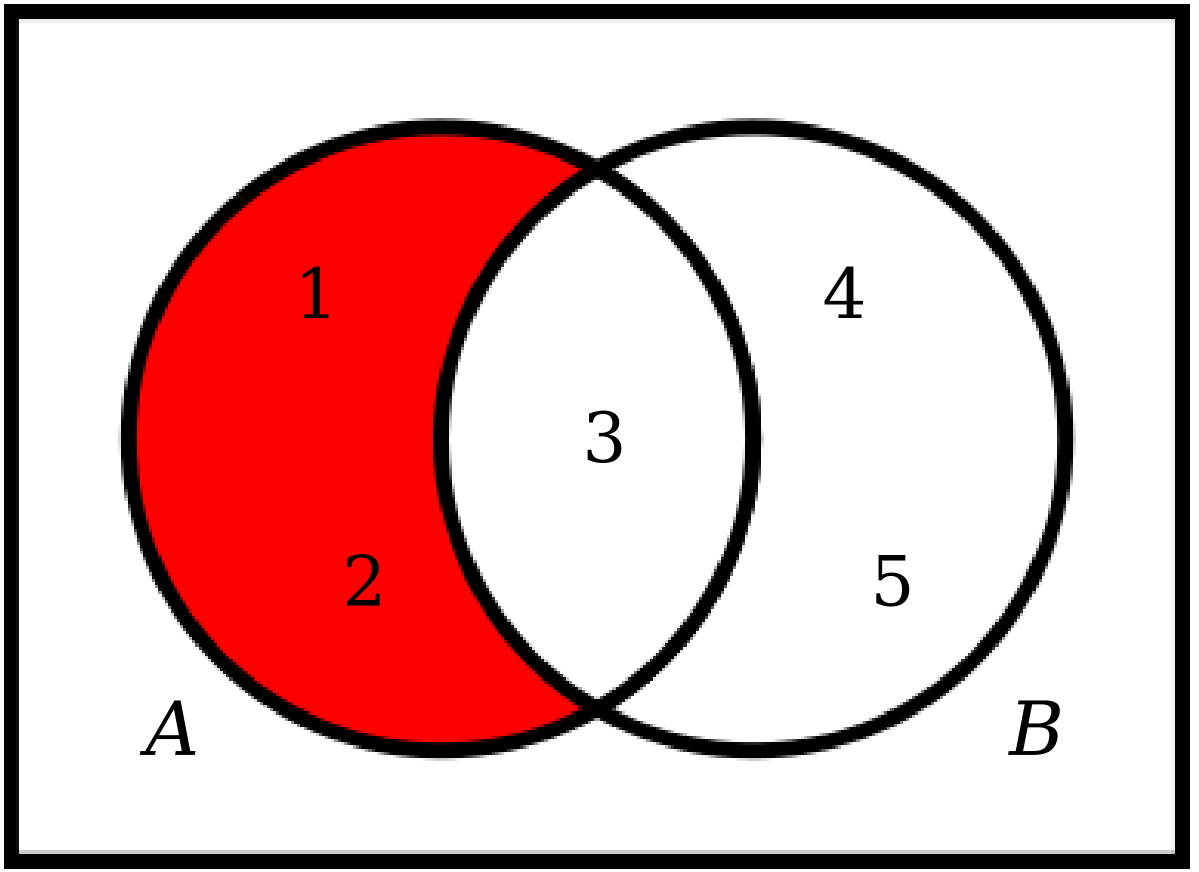
$B = \{ 3, 4, 5 \}$

Venn Diagrams



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

Venn Diagrams



Difference

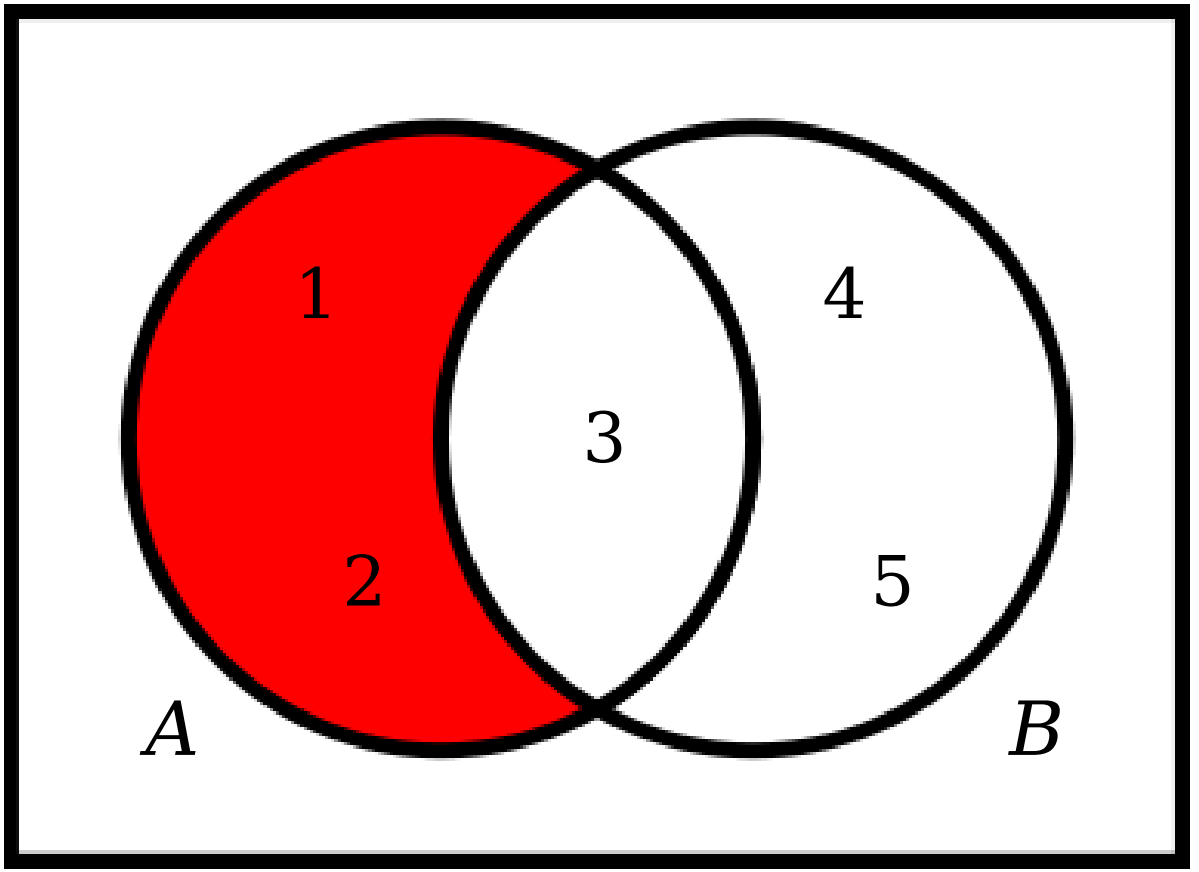
$A - B$

{ 1, 2 }

A = { 1, 2, 3 }

B = { 3, 4, 5 }

Venn Diagrams



Difference

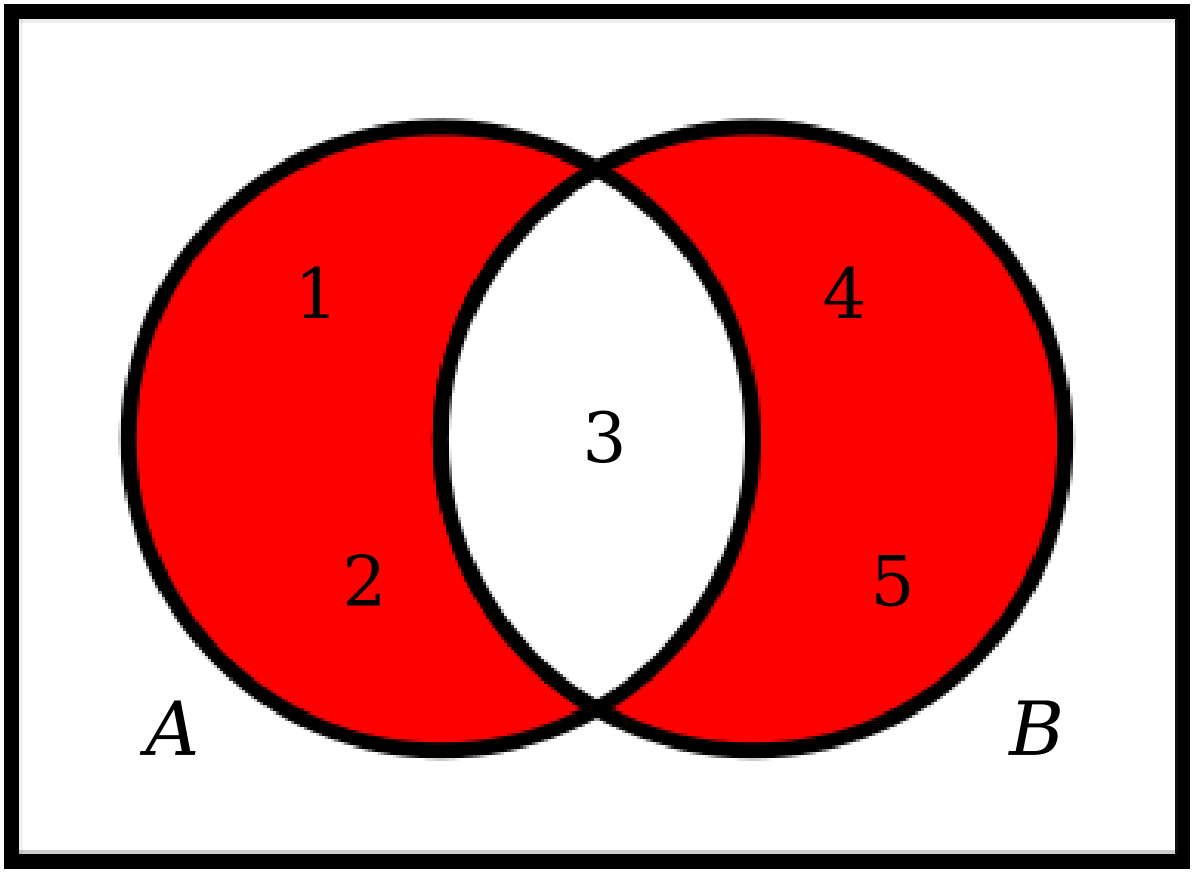
$A \setminus B$

{ 1, 2 }

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

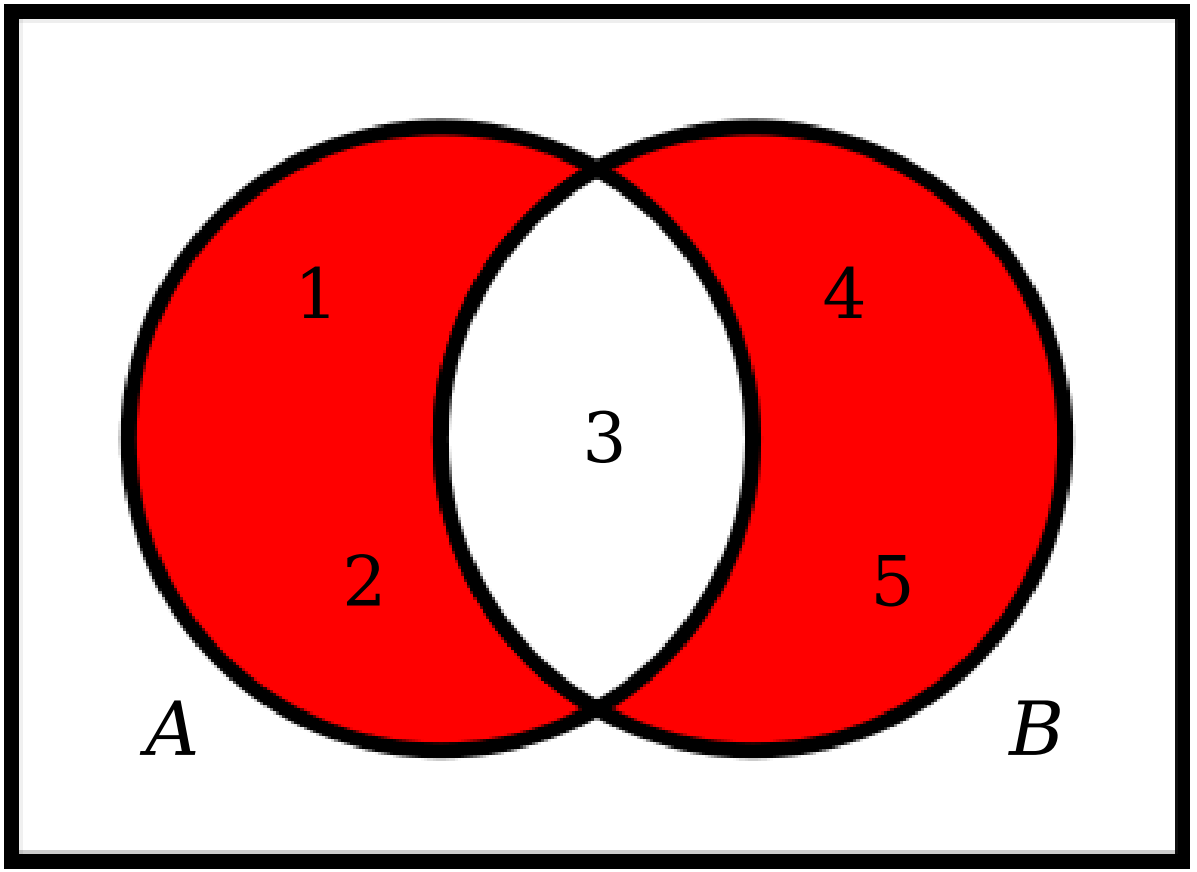
$B = \{ 3, 4, 5 \}$

Venn Diagrams



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

Venn Diagrams



Symmetric Difference

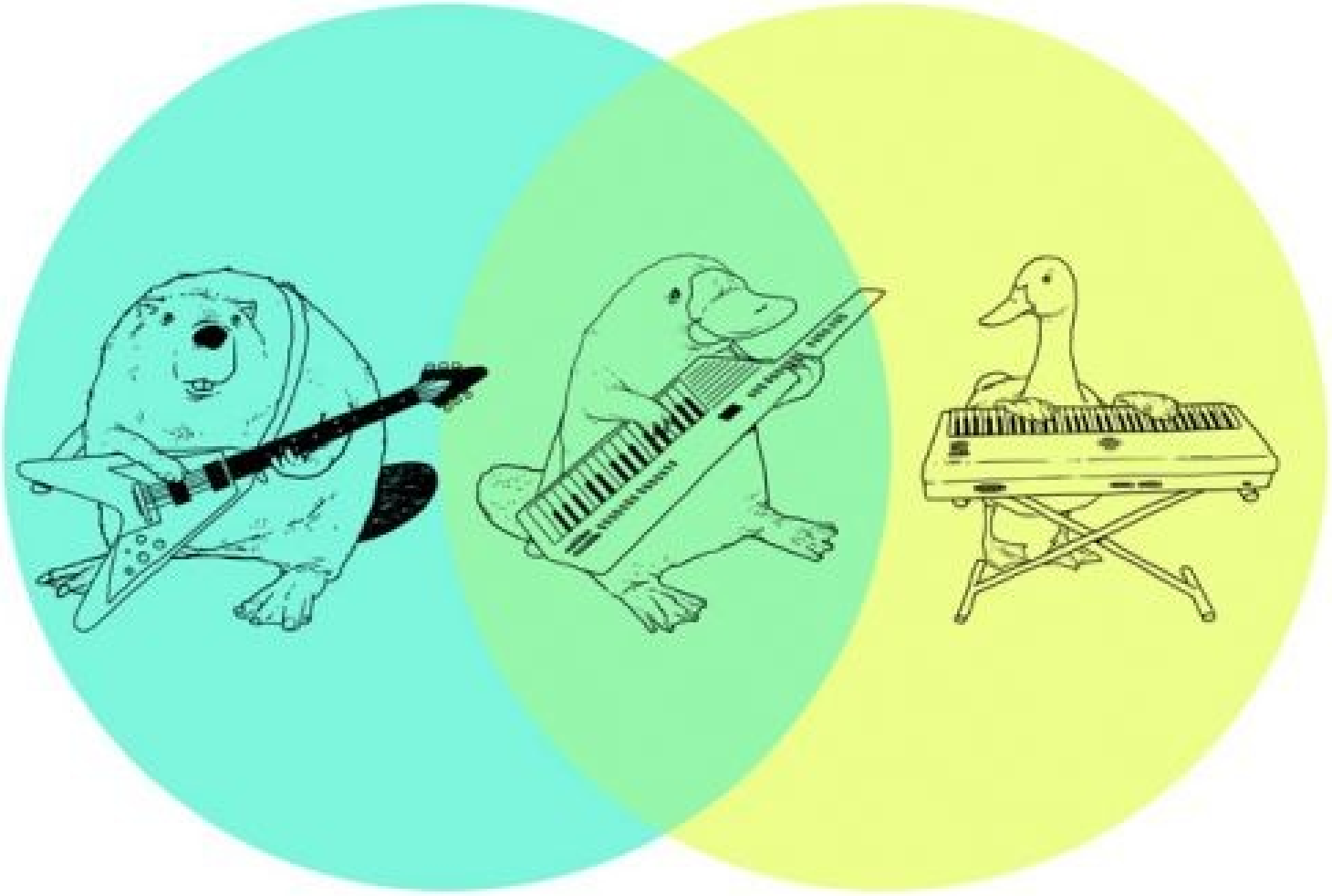
$A \Delta B$

{ 1, 2, 4, 5 }

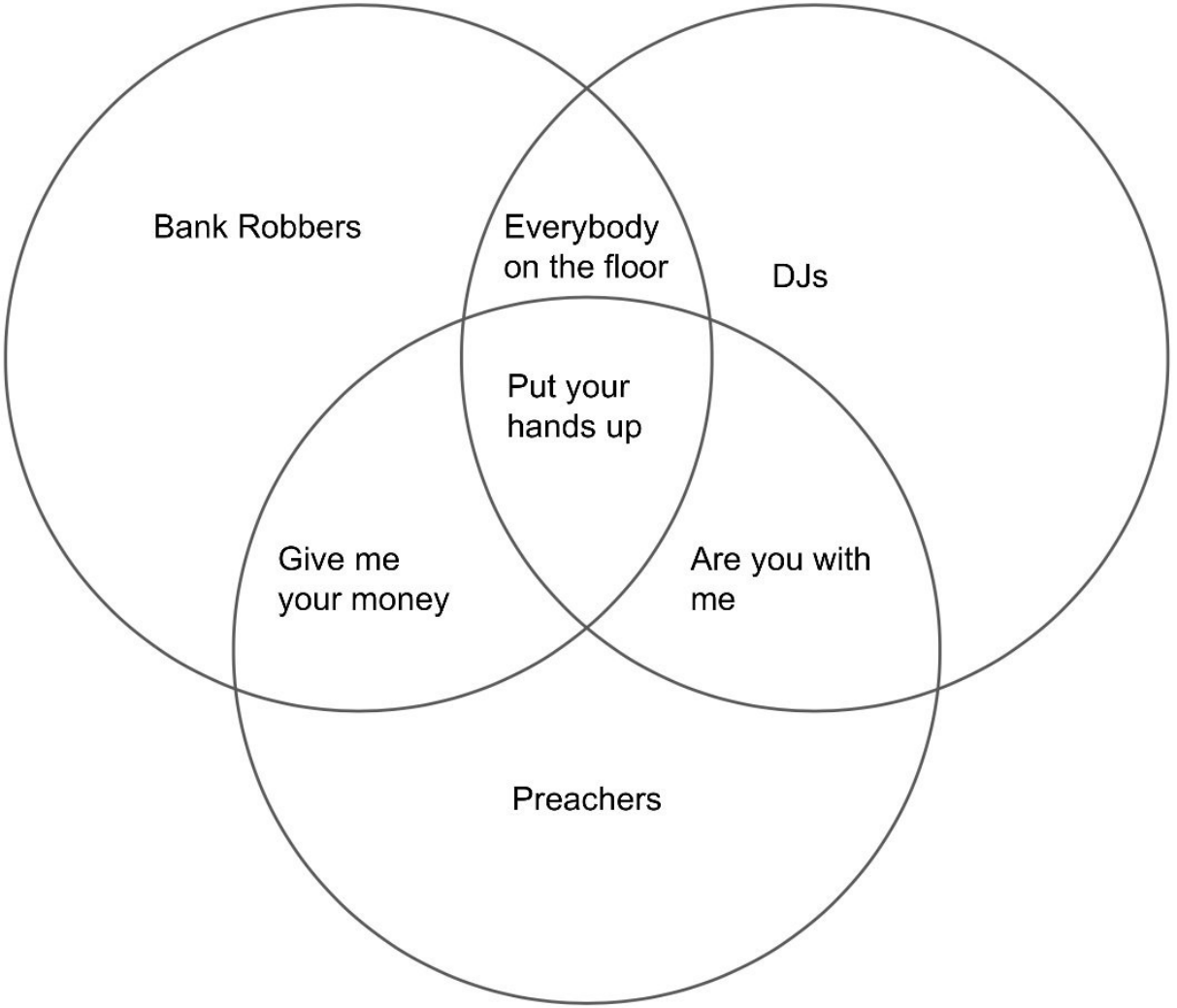
A = { 1, 2, 3 }

B = { 3, 4, 5 }

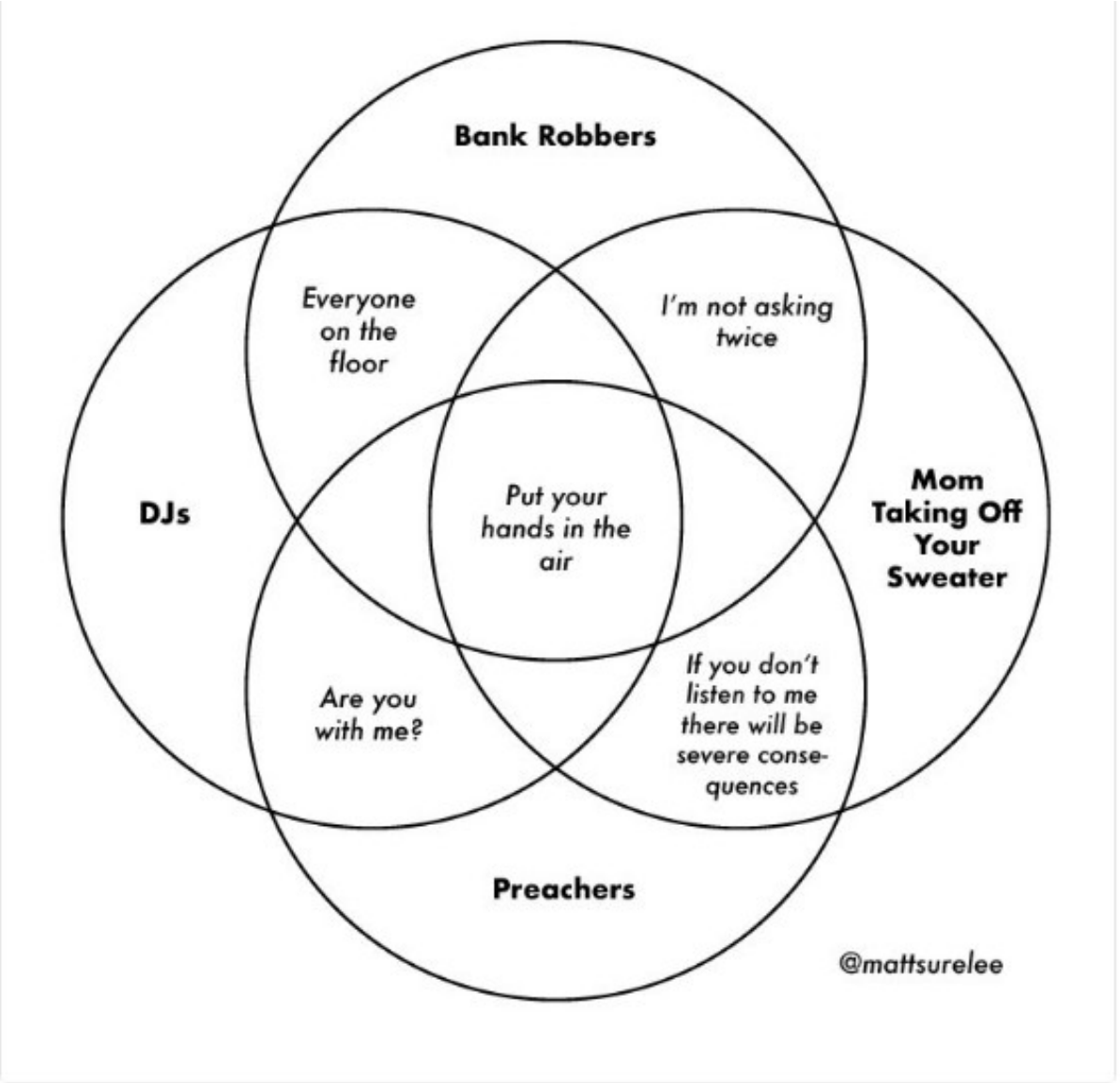
Venn Diagrams



Venn Diagrams for Three Sets



Venn Diagrams for Three Sets

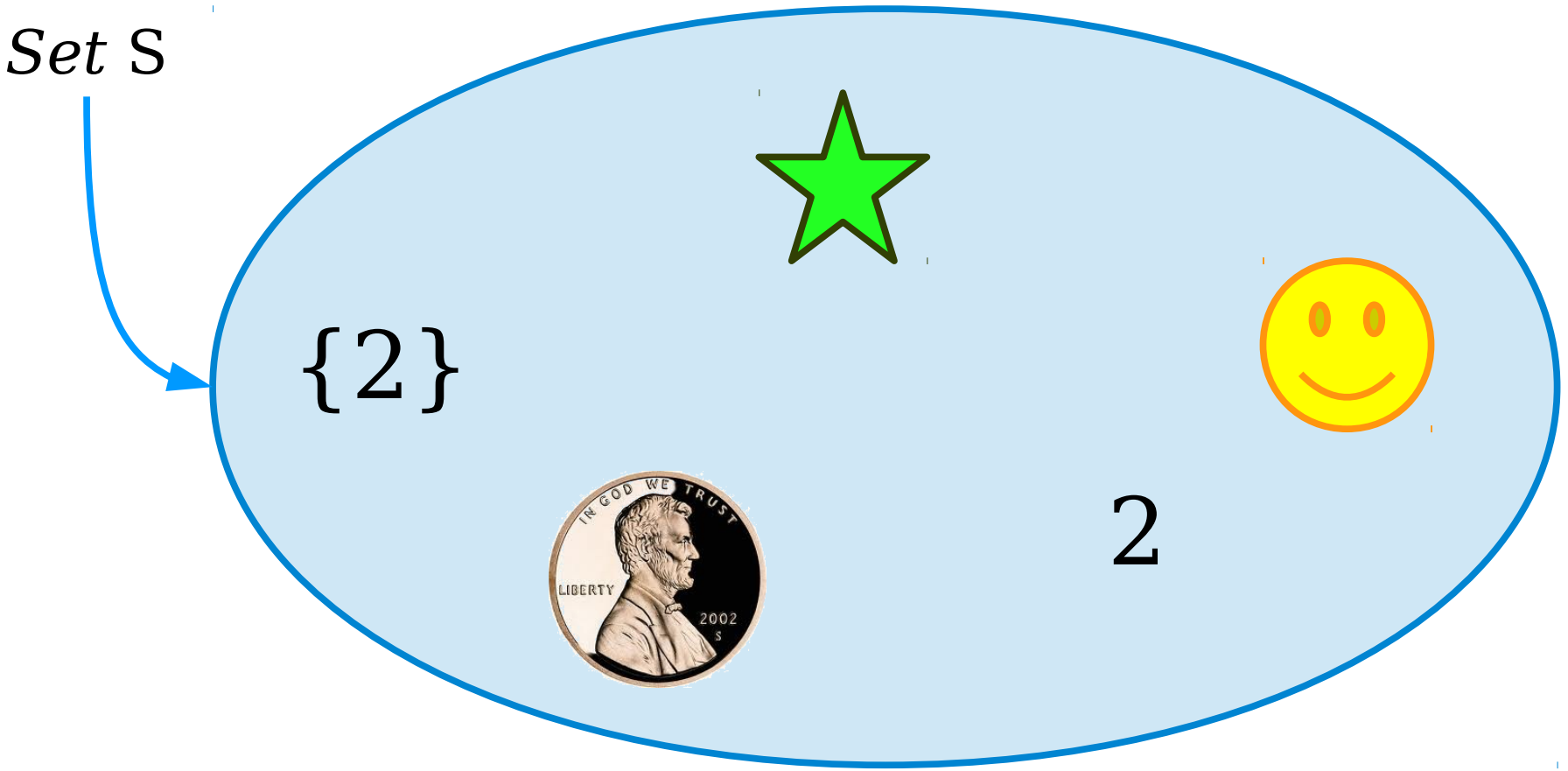


Subsets and Power Sets

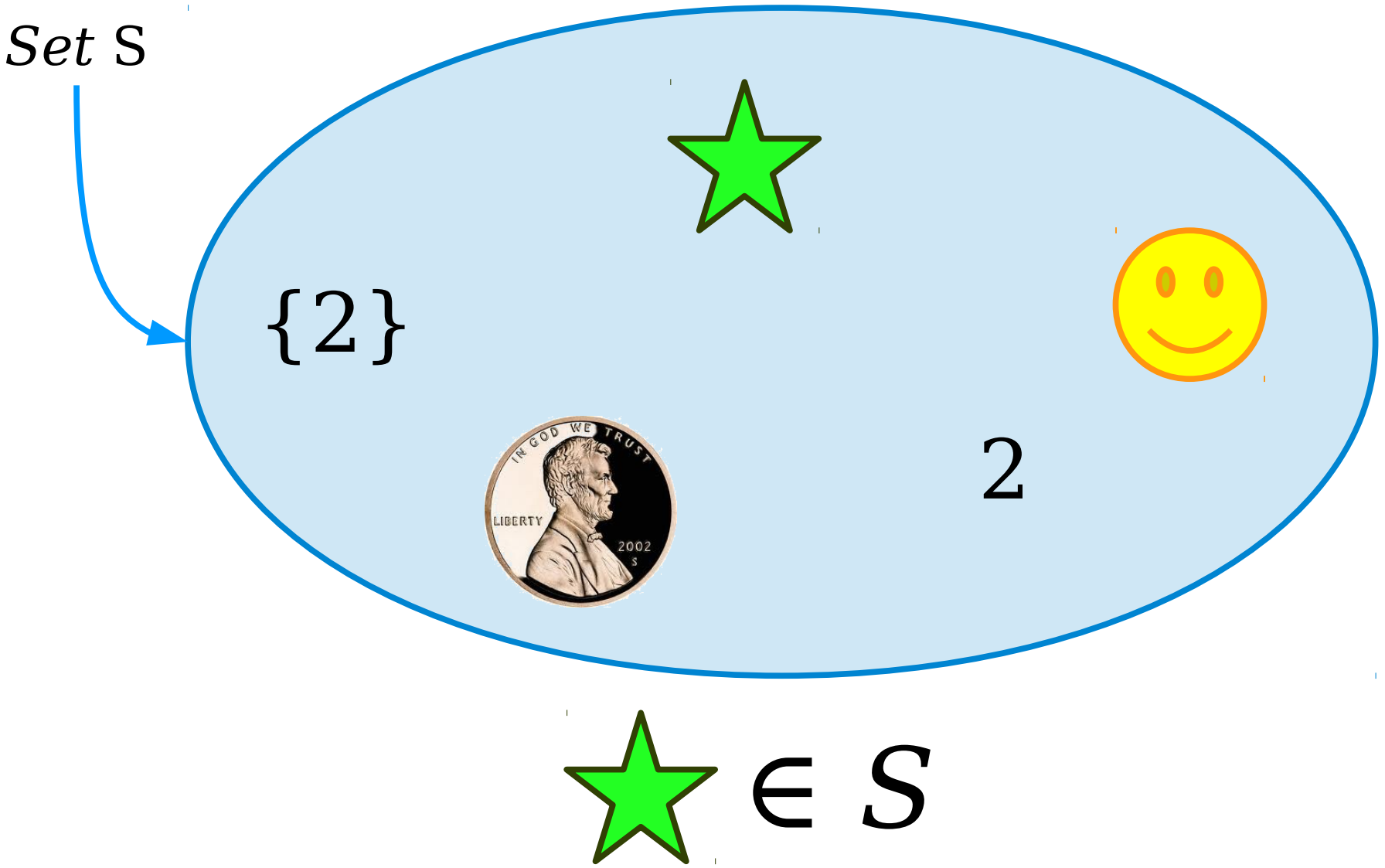
Subsets

- A set S is called a ***subset*** of a set T (denoted $\boldsymbol{S \subseteq T}$) if 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*)

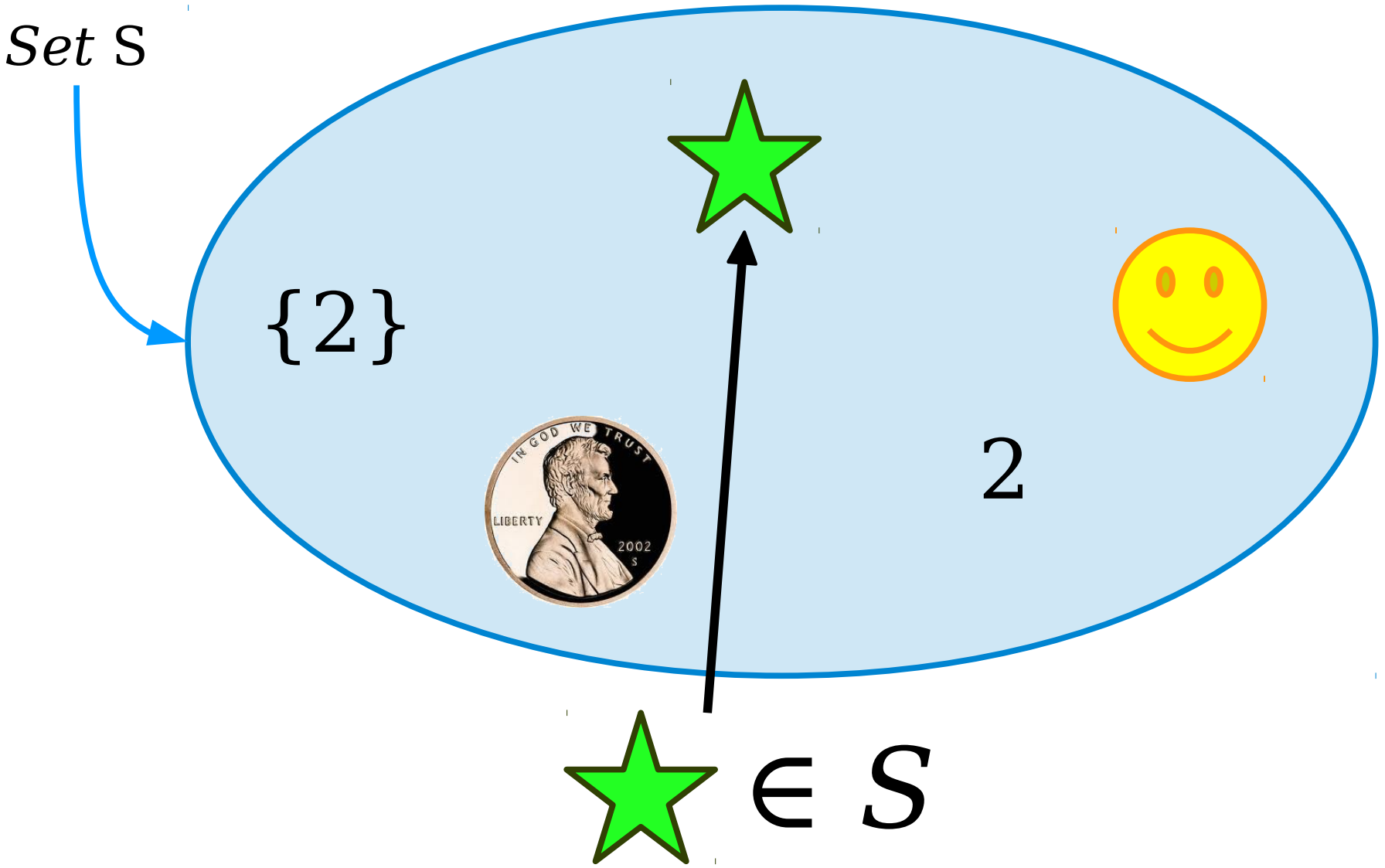
Subsets and Elements



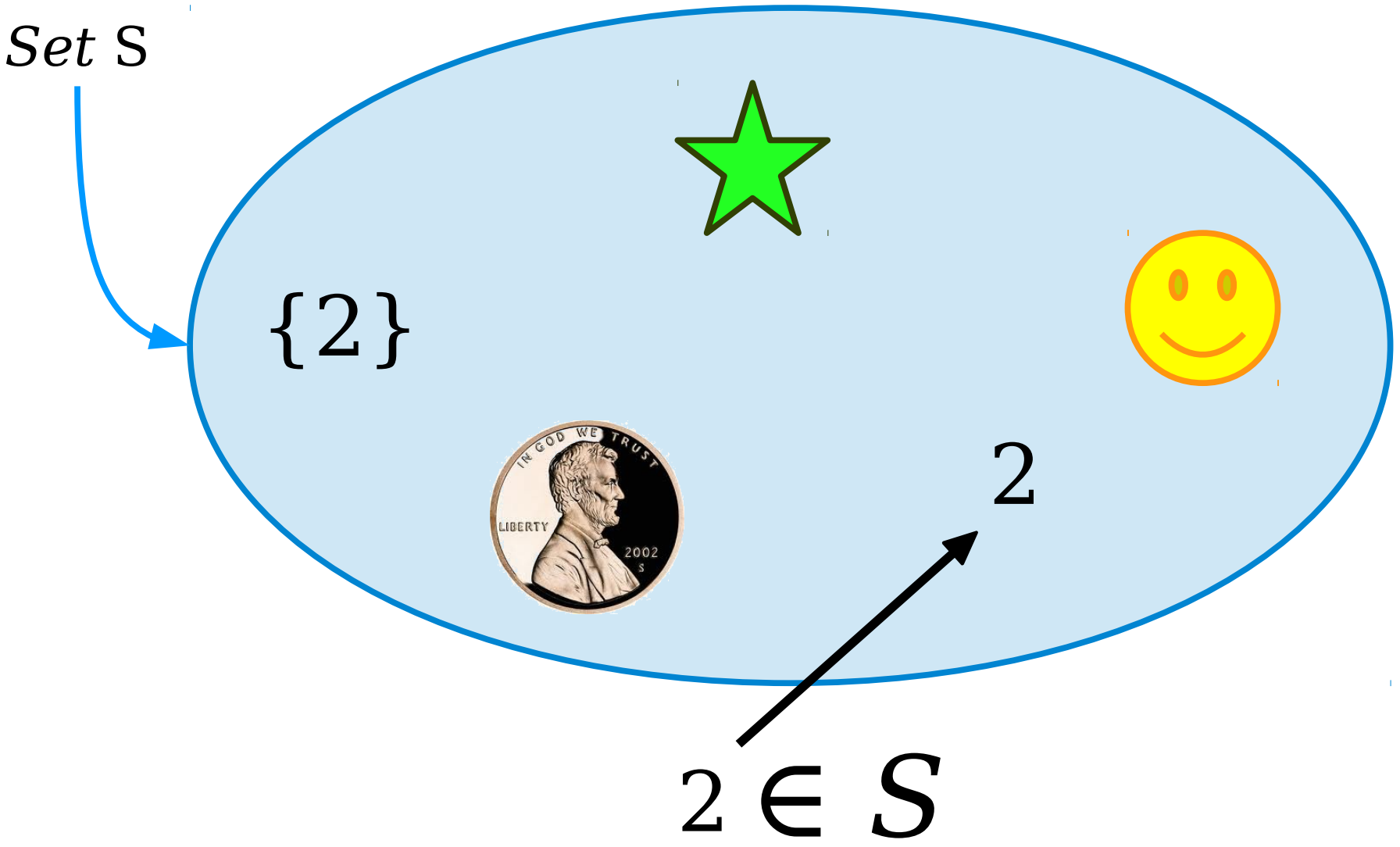
Subsets and Elements



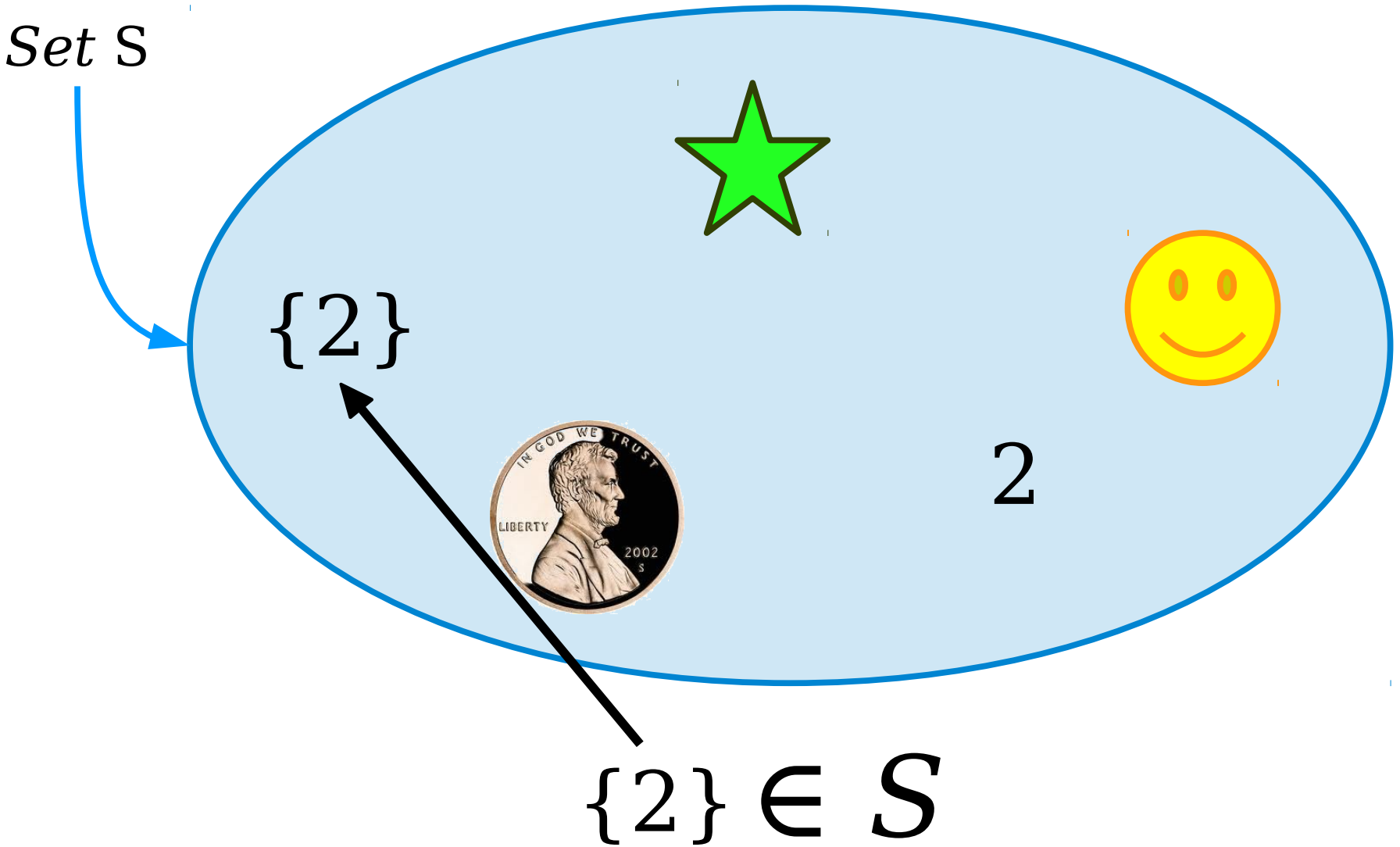
Subsets and Elements



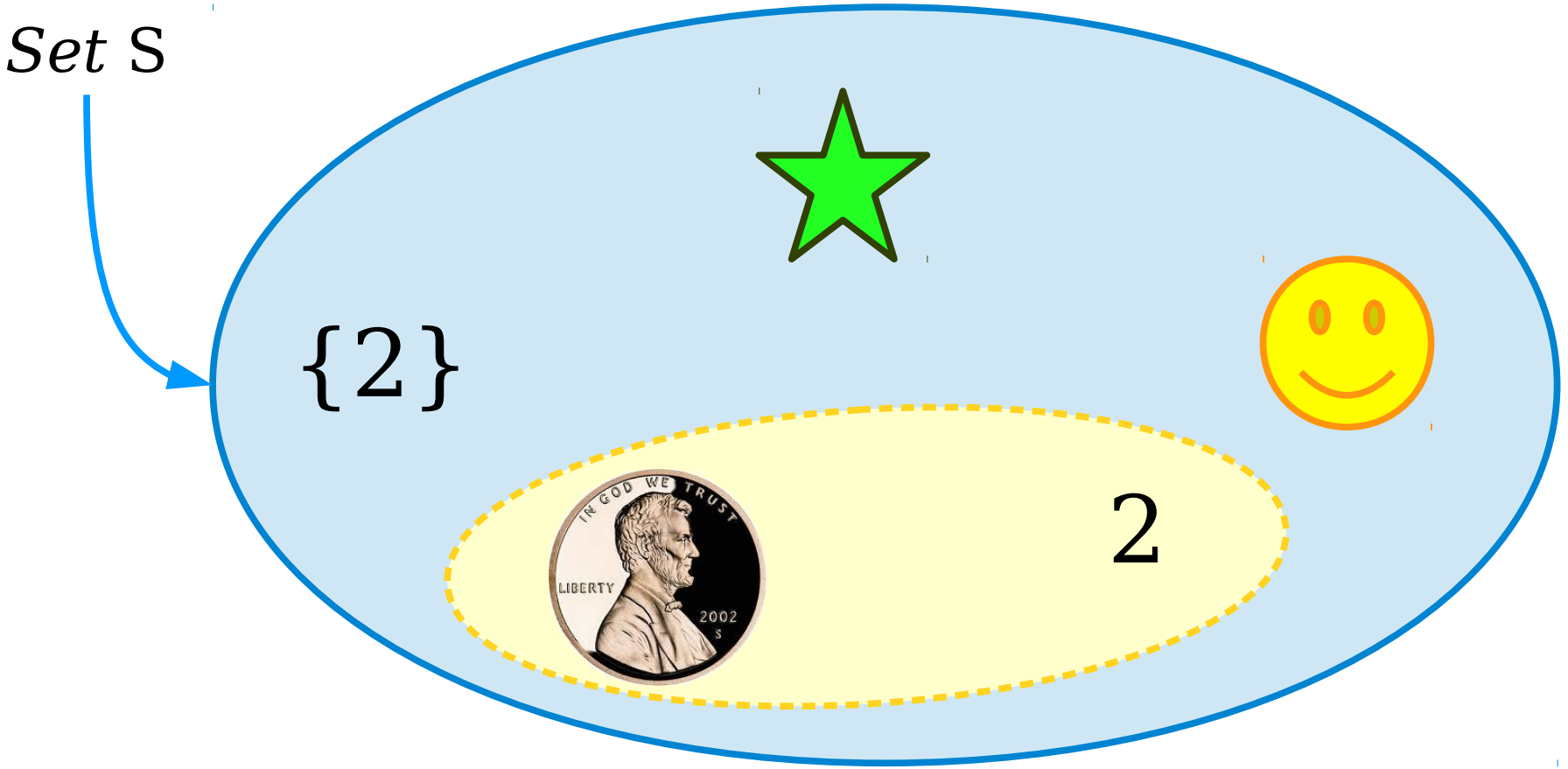
Subsets and Elements



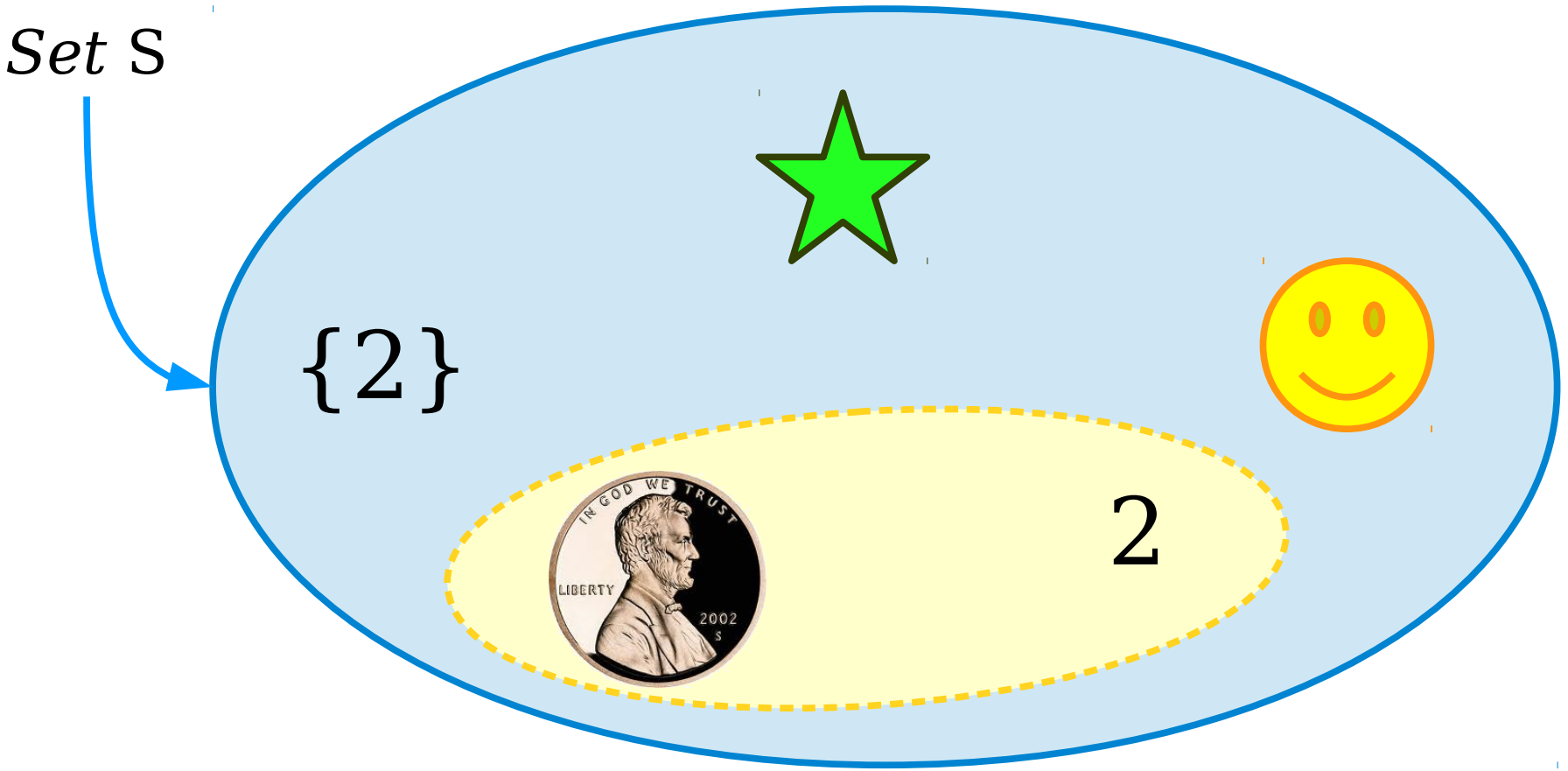
Subsets and Elements



Subsets and Elements

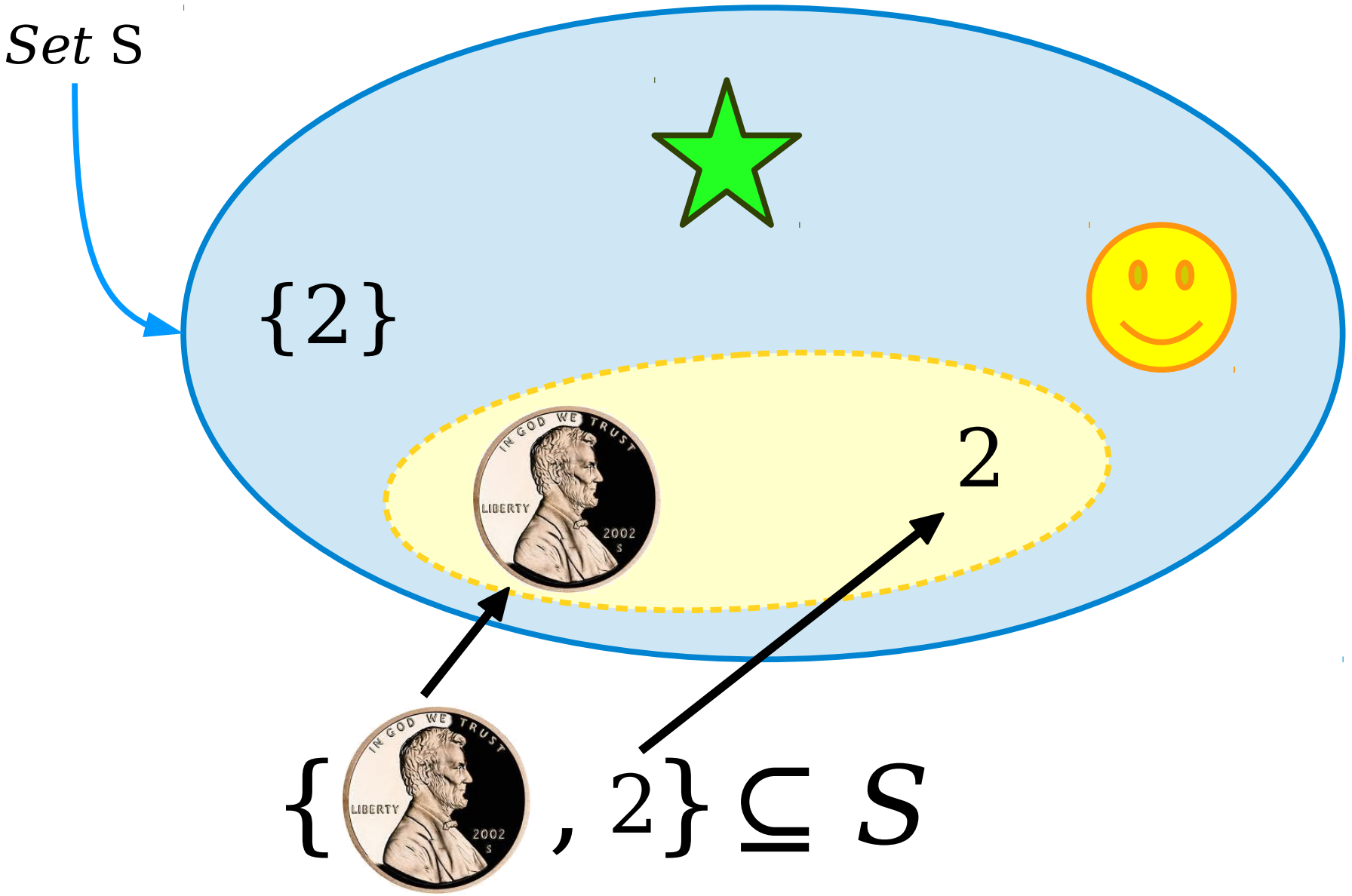


Subsets and Elements

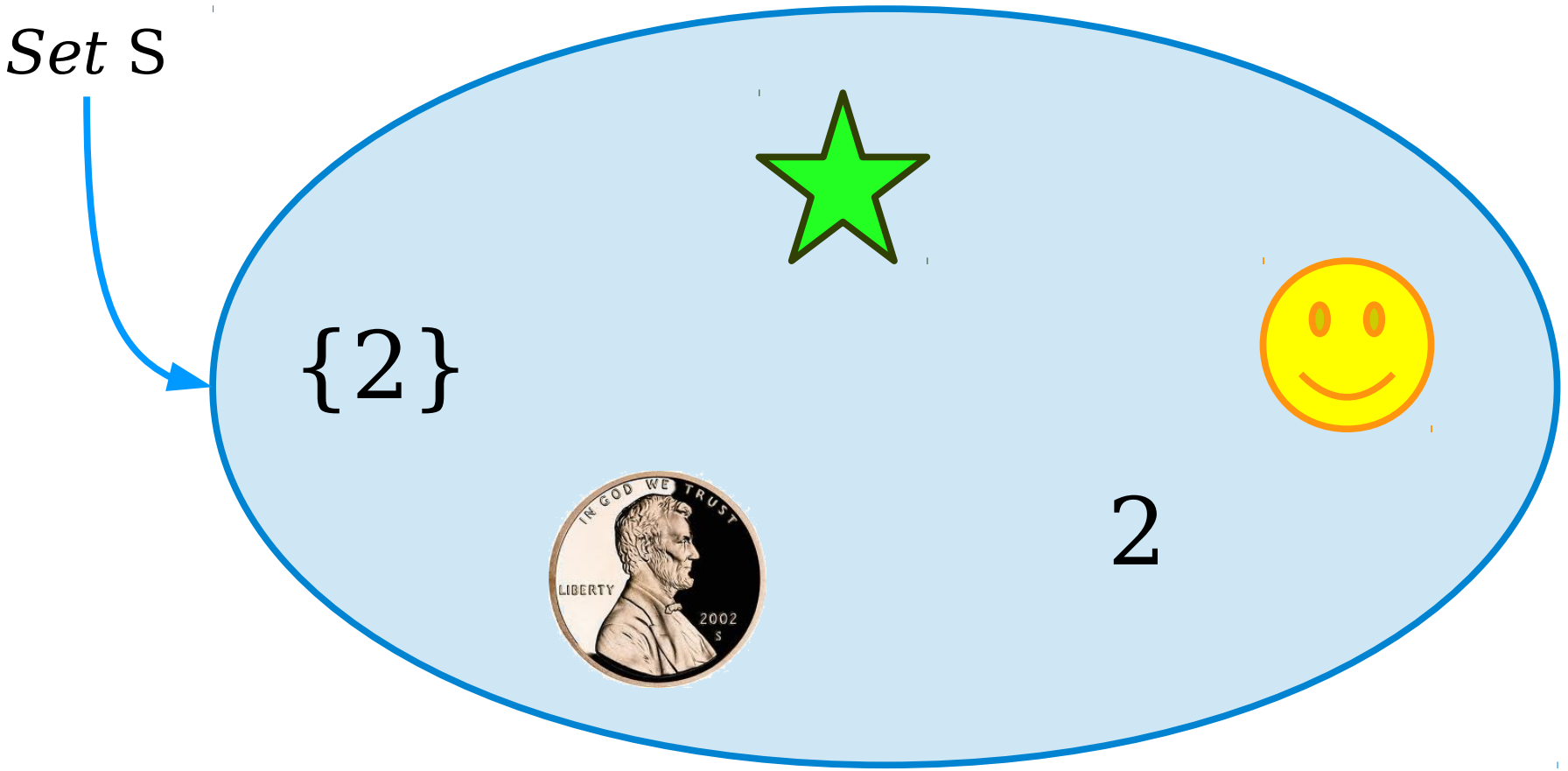


$\{$ $, 2\} \subseteq S$

Subsets and Elements

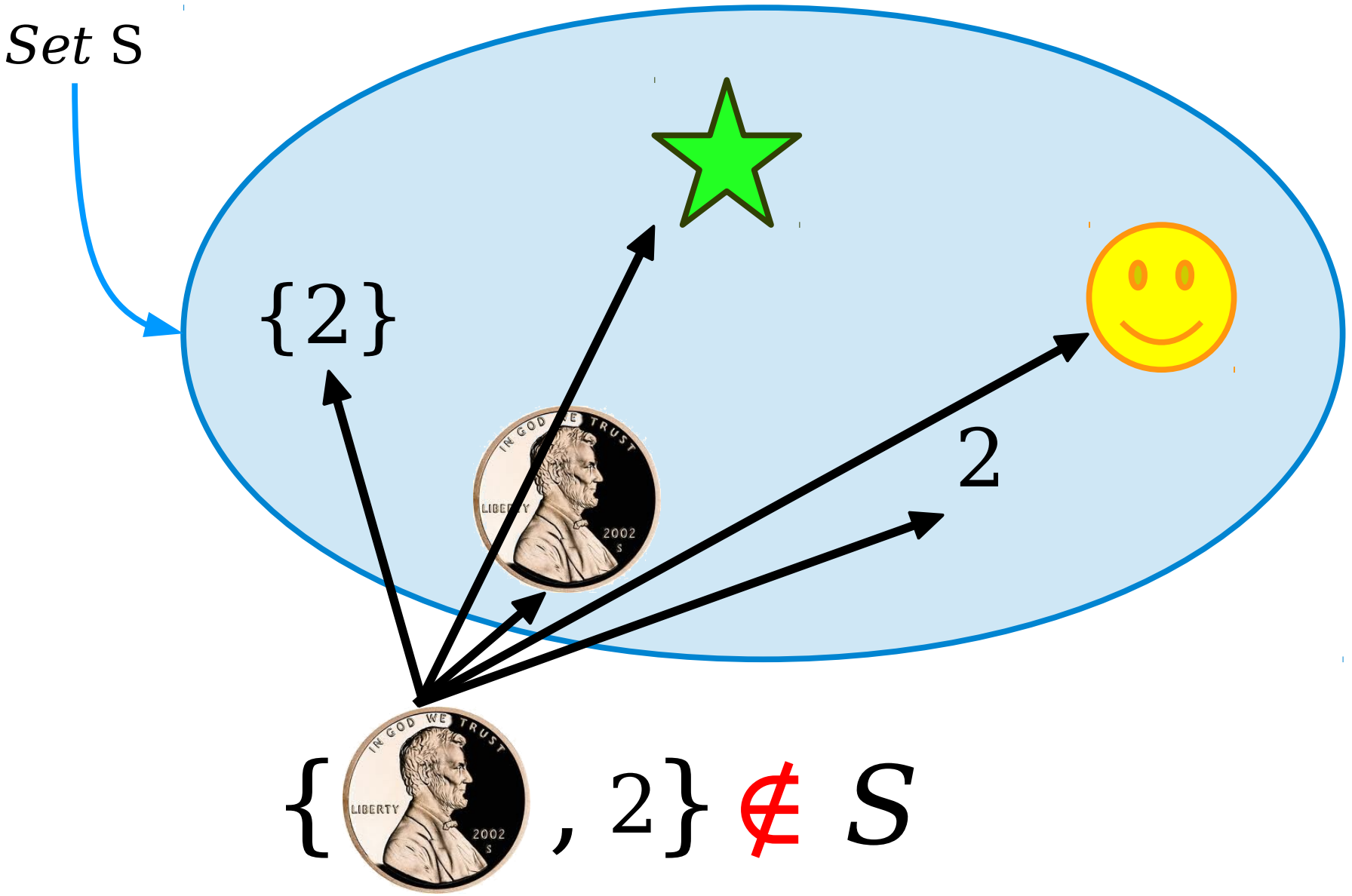


Subsets and Elements

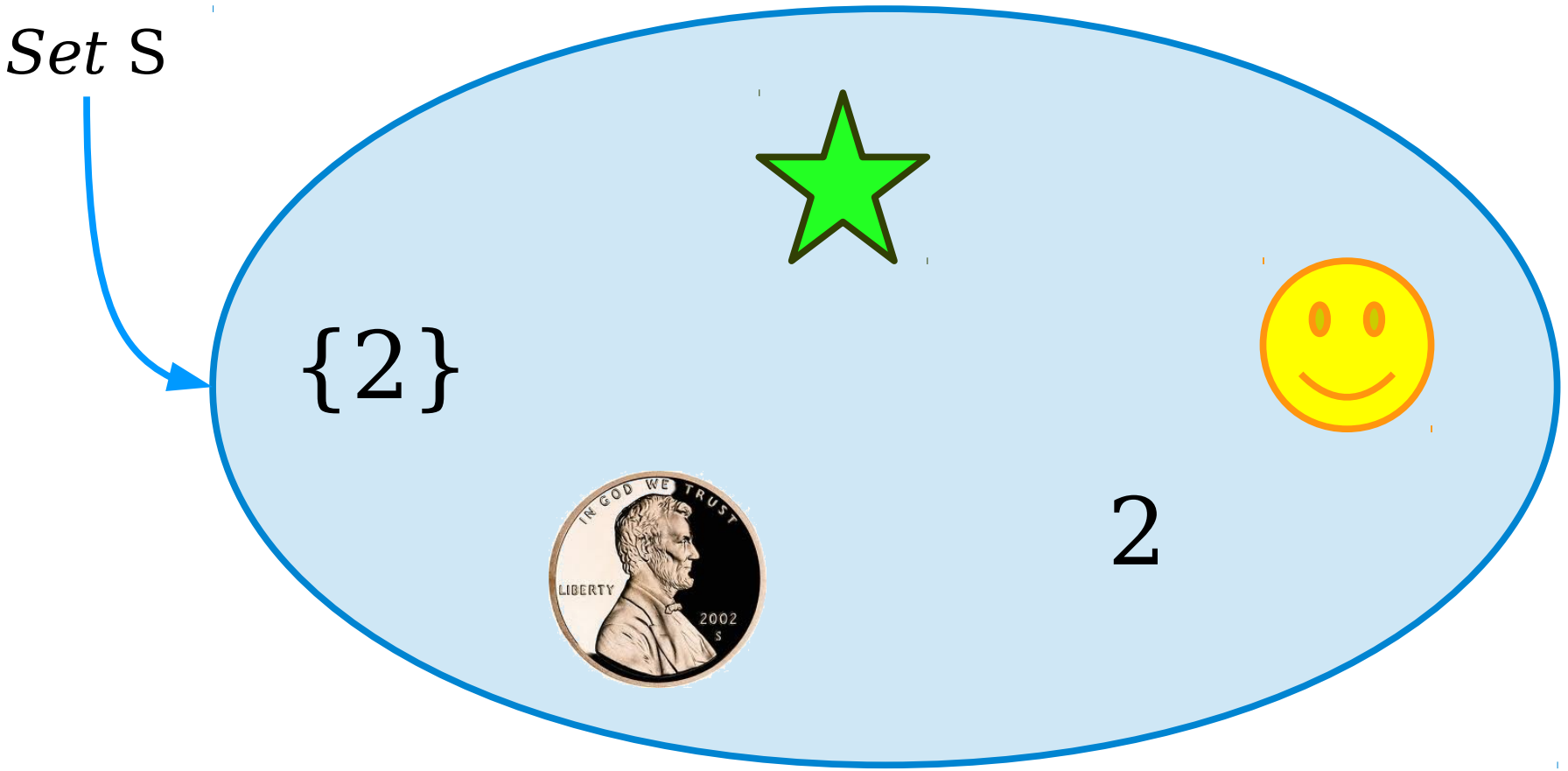


$\{$ $, 2\} \notin S$

Subsets and Elements

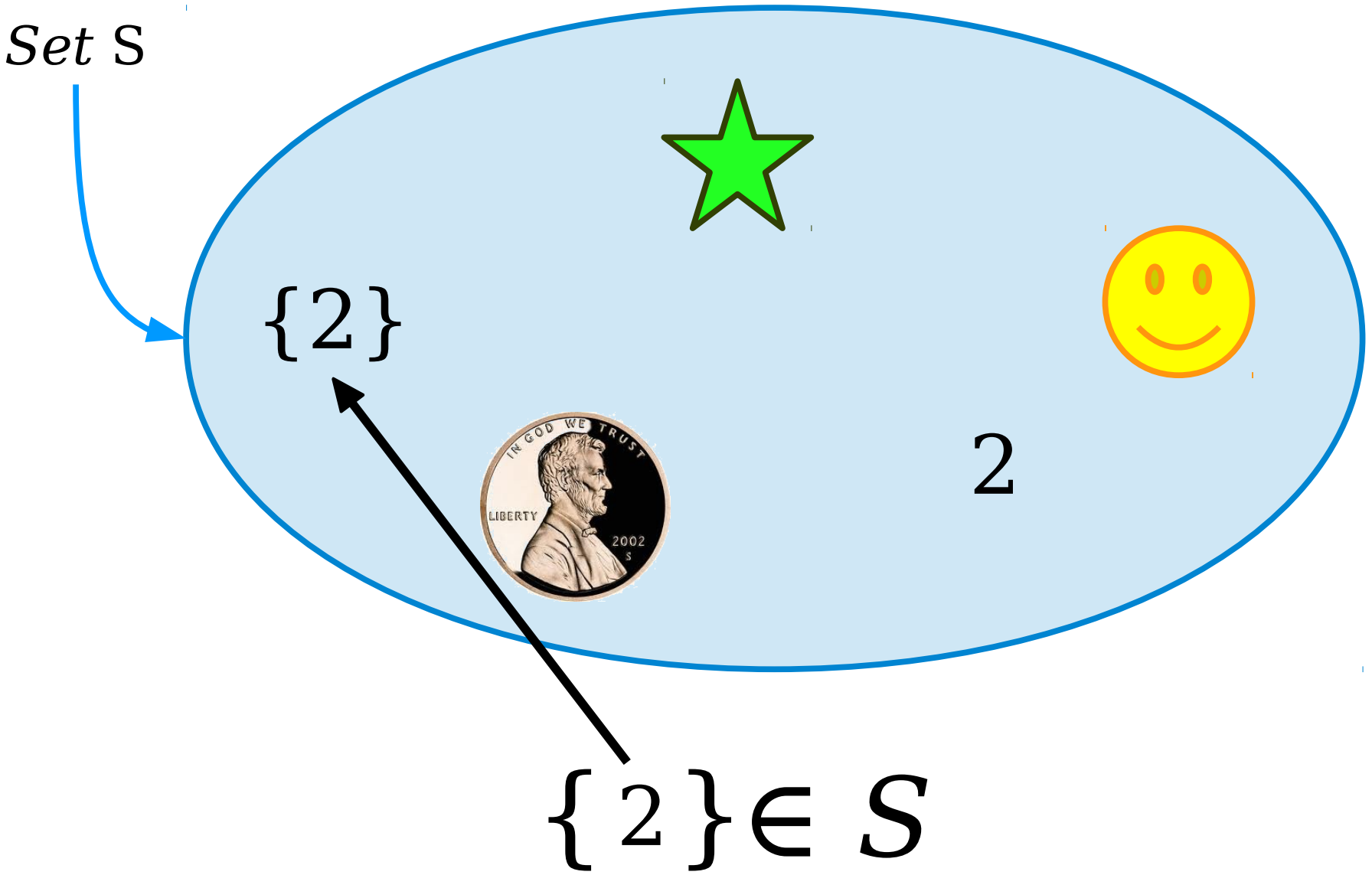


Subsets and Elements

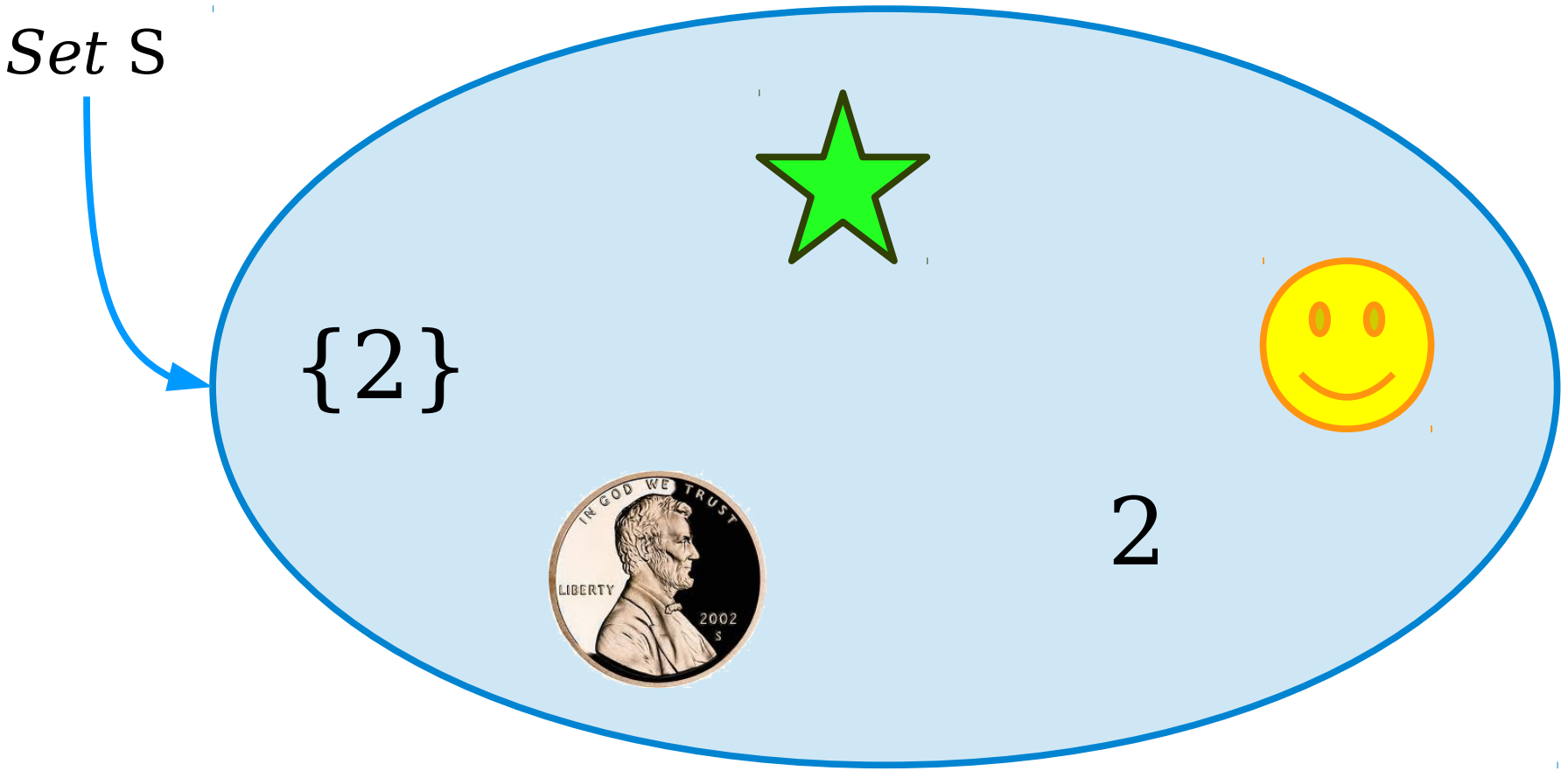


$$\{2\} \in S$$

Subsets and Elements

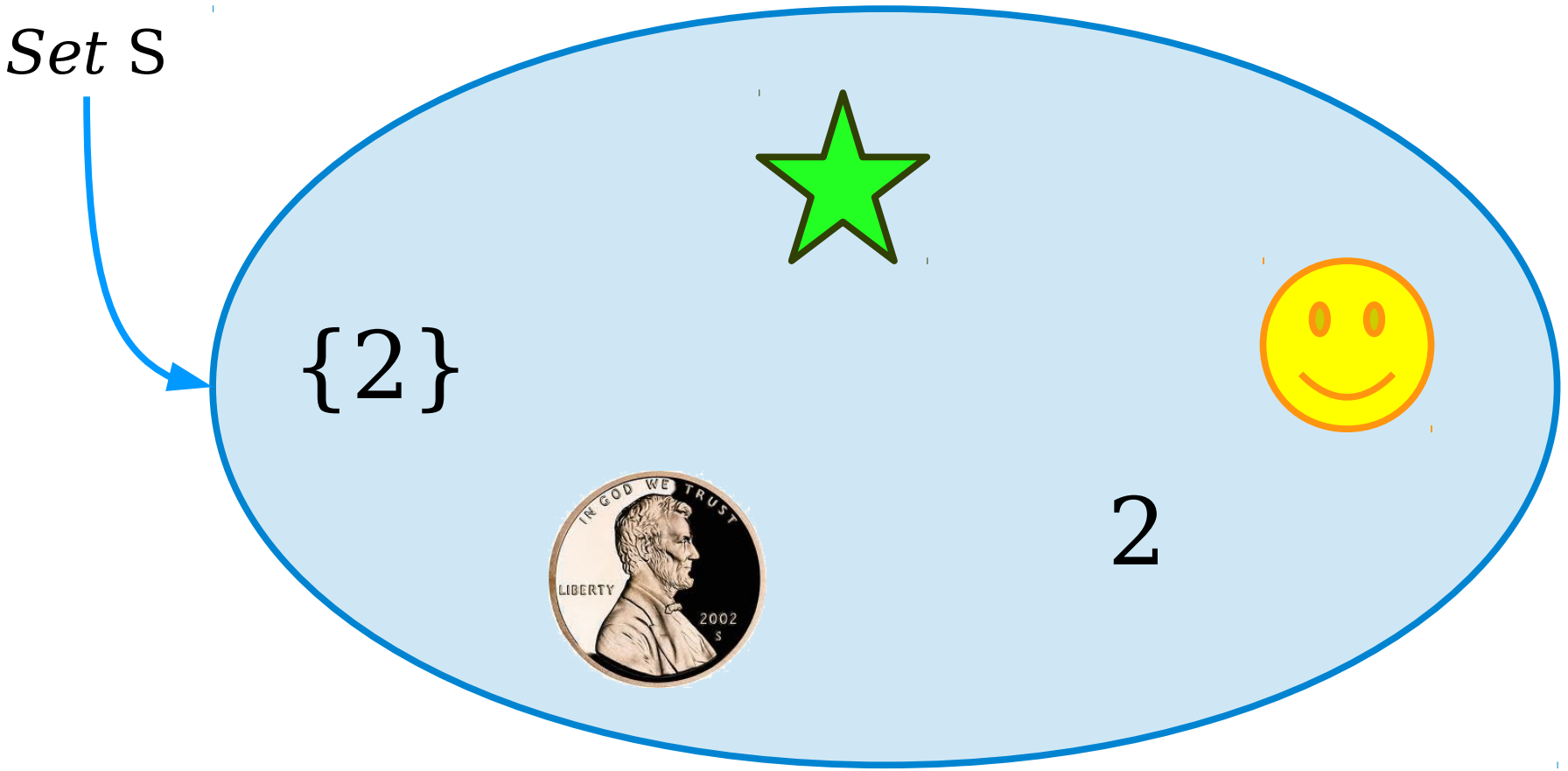


Subsets and Elements



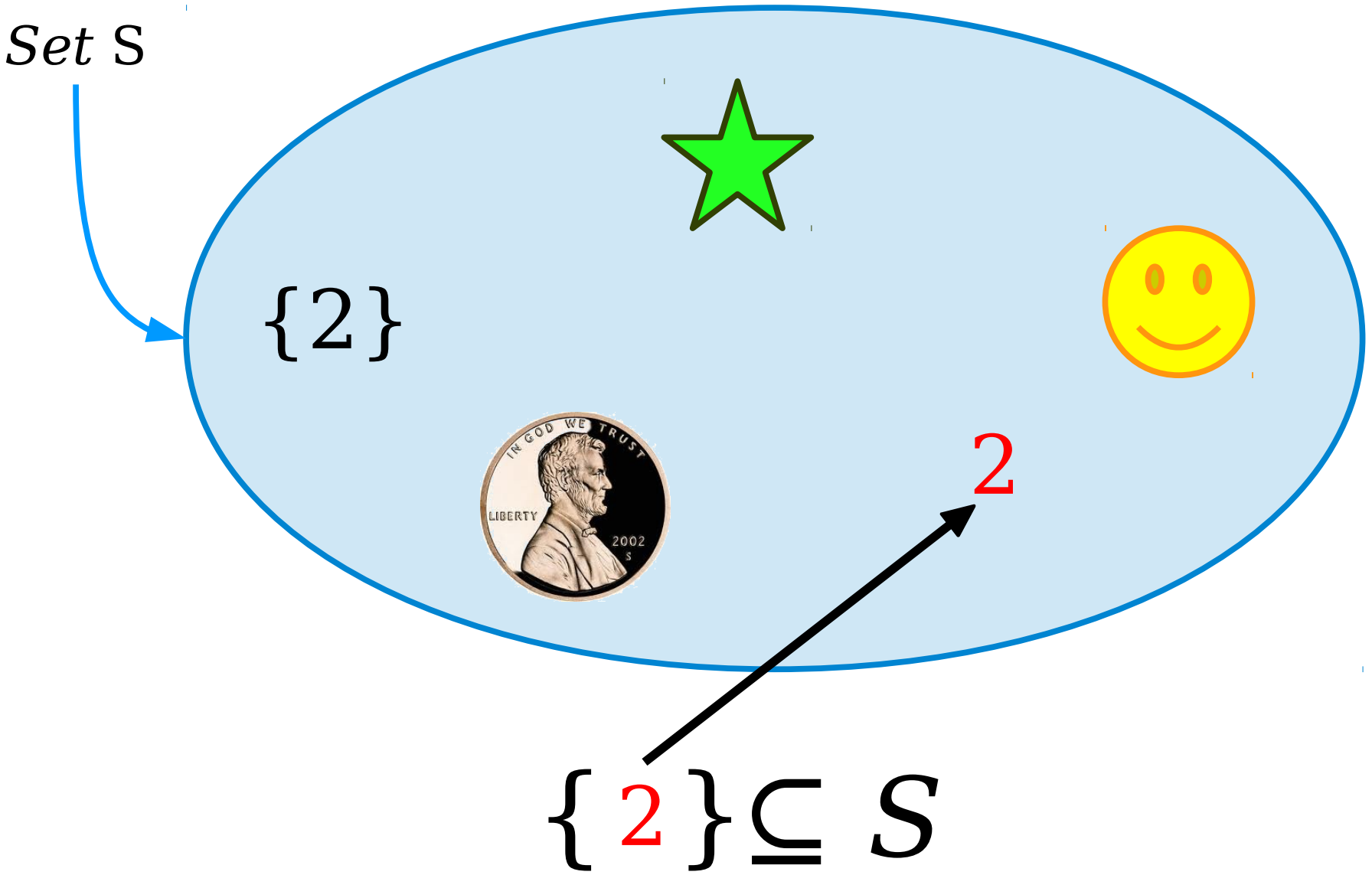
$$\{2\} \subseteq S$$

Subsets and Elements

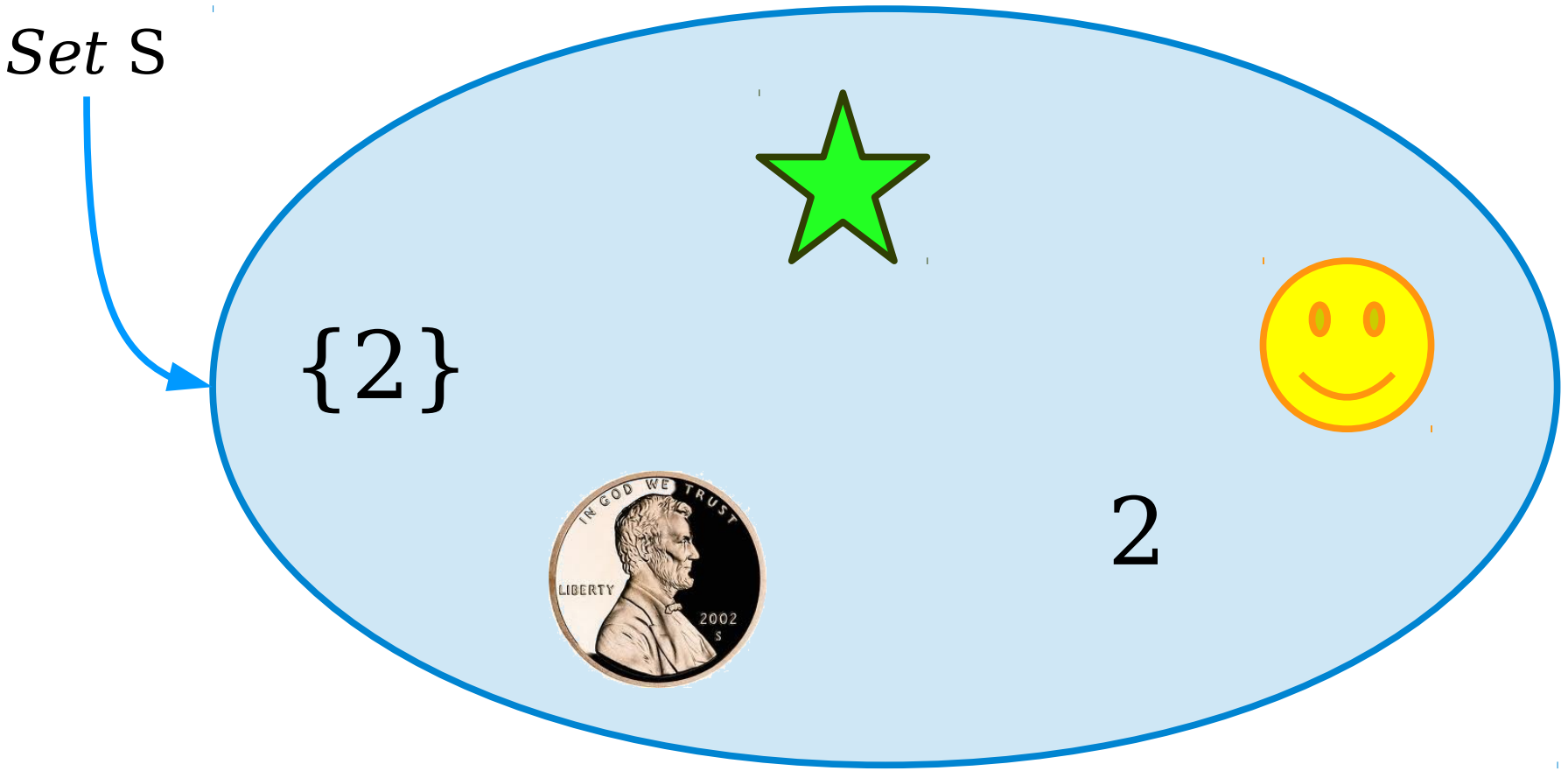


$\{2\} \subseteq S$

Subsets and Elements

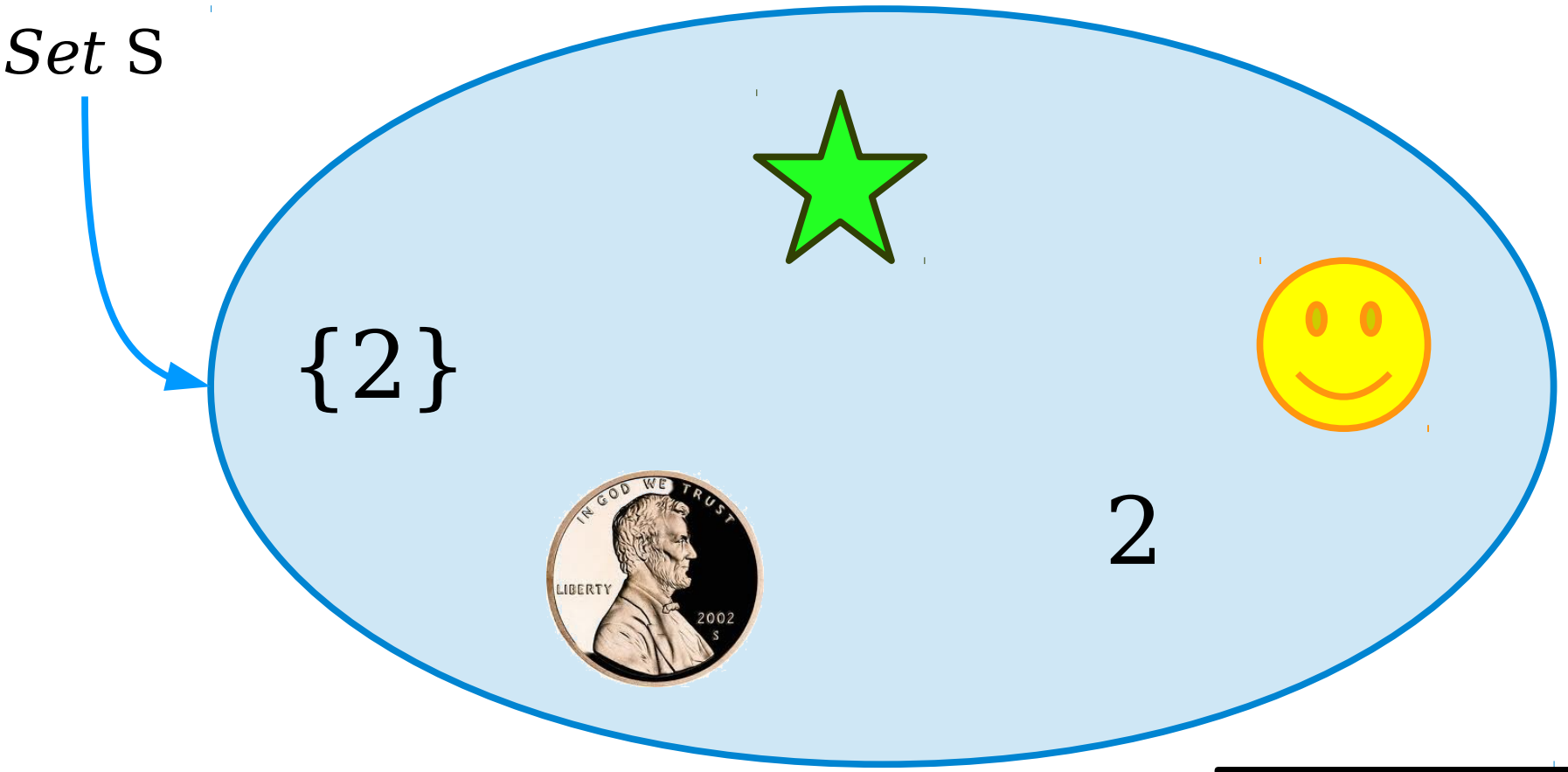


Subsets and Elements



$$2 \notin S$$

Subsets and Elements



$2 \notin S$

(Since 2 isn't a set.)

Subsets and Elements

- We say that $S \in T$ if, among the elements of T, one of them is *exactly* the object S.
- We say that $S \subseteq T$ if 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 meaningful.)
- 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.

What About the Empty Set?

- A set S is called a ***subset*** of a set T (denoted $S \subseteq T$) if all elements of S are also elements of T.
- Are there any sets S where $\varnothing \subseteq S$?
- Equivalently, is there a set S where the following statement is true?

"All elements of Ø are also elements of *S*"

- ***Yes!*** In fact, this statement is true for every choice of S!

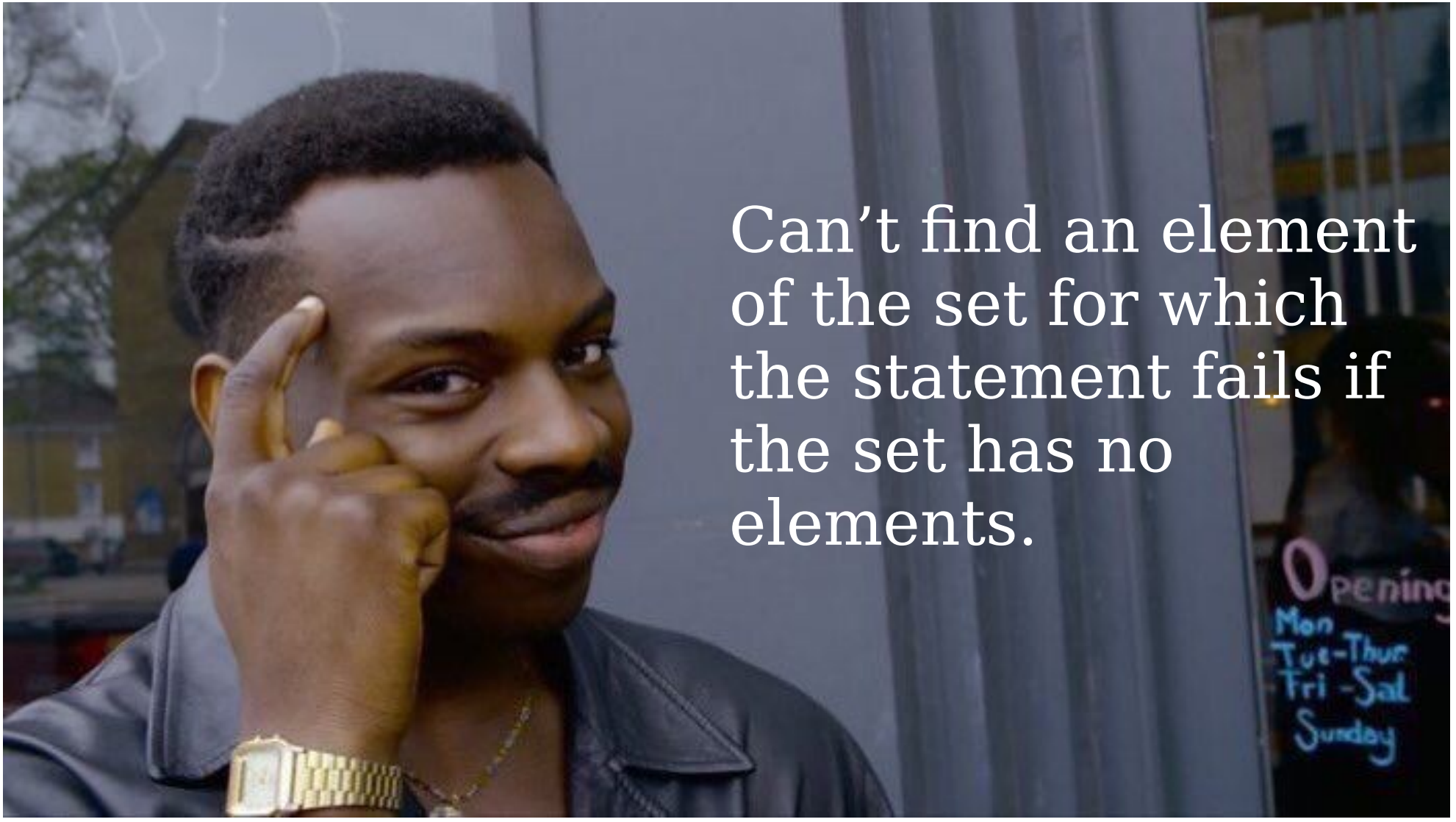
Vacuous Truth

- A statement of the form

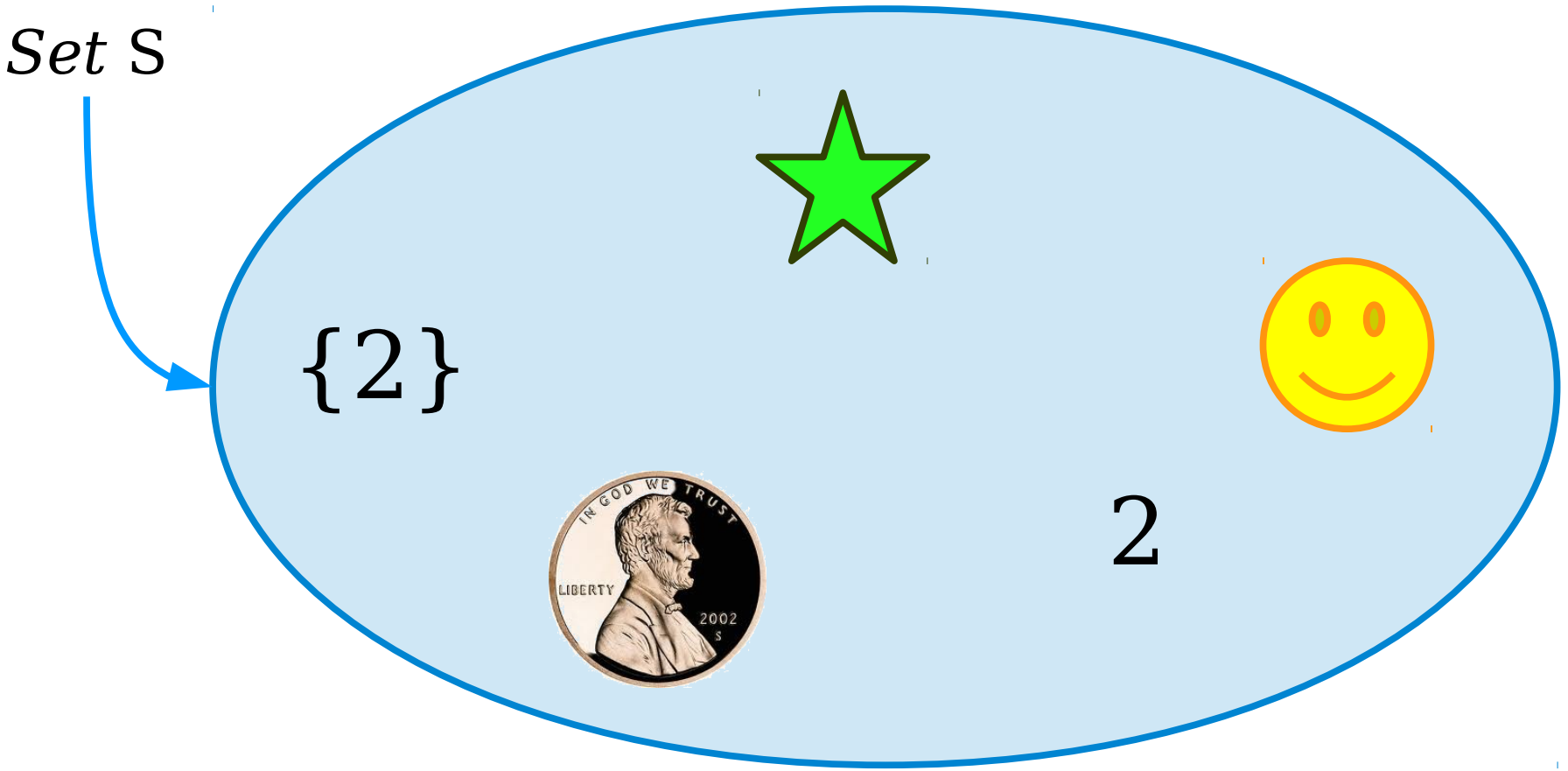
  **"All objects of type *P*
  are also of type *Q*"**

  is called ***vacuously true*** if there are no objects of type P.
- Vacuously true statements are true *by definition*. This is a convention used throughout mathematics.
- Some examples:
  - All unicorns are pink.
  - All unicorns are blue.
  - Every element of Ø is also an element of S.

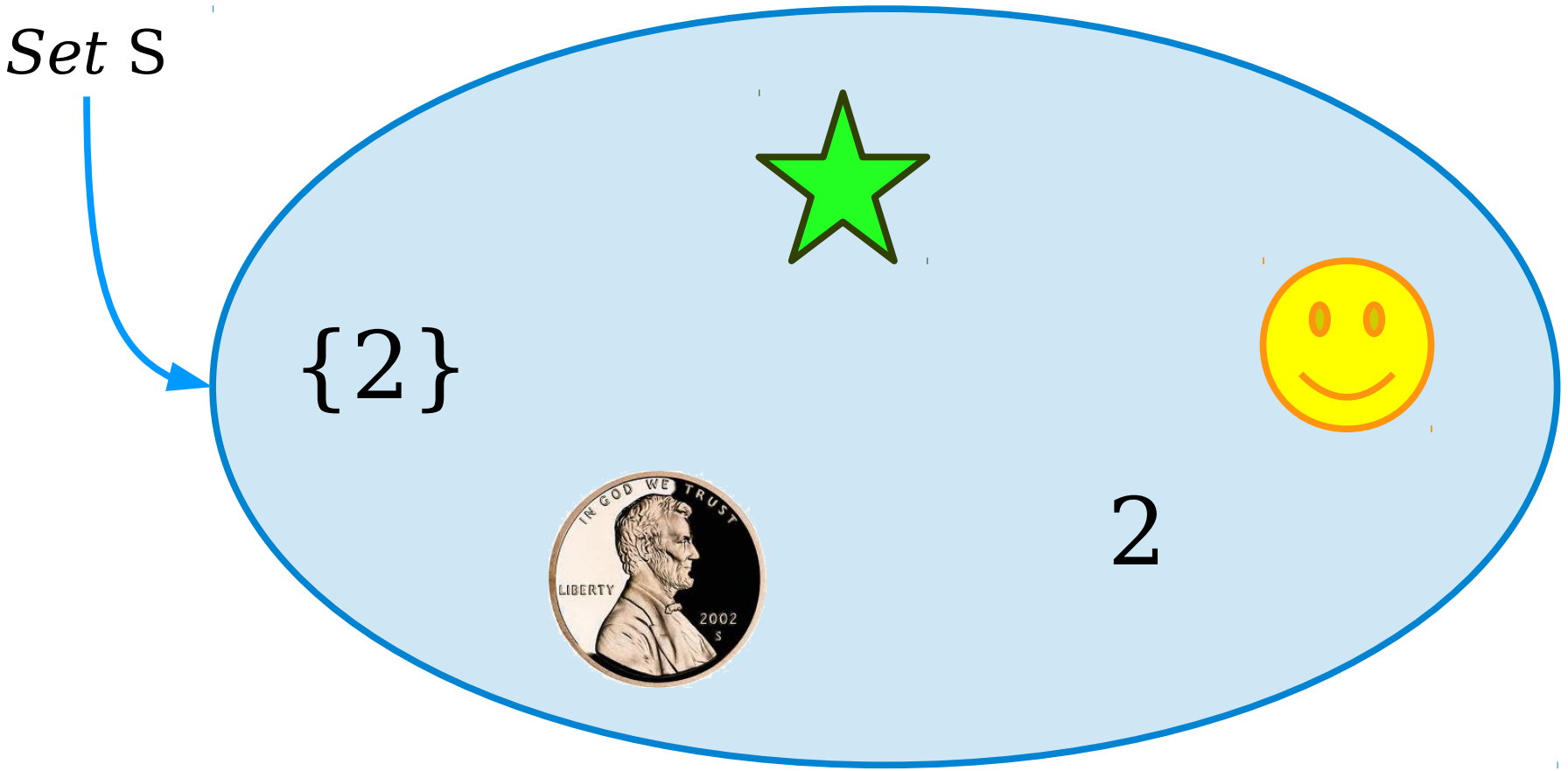
Vacuous Truth



Subsets and Elements

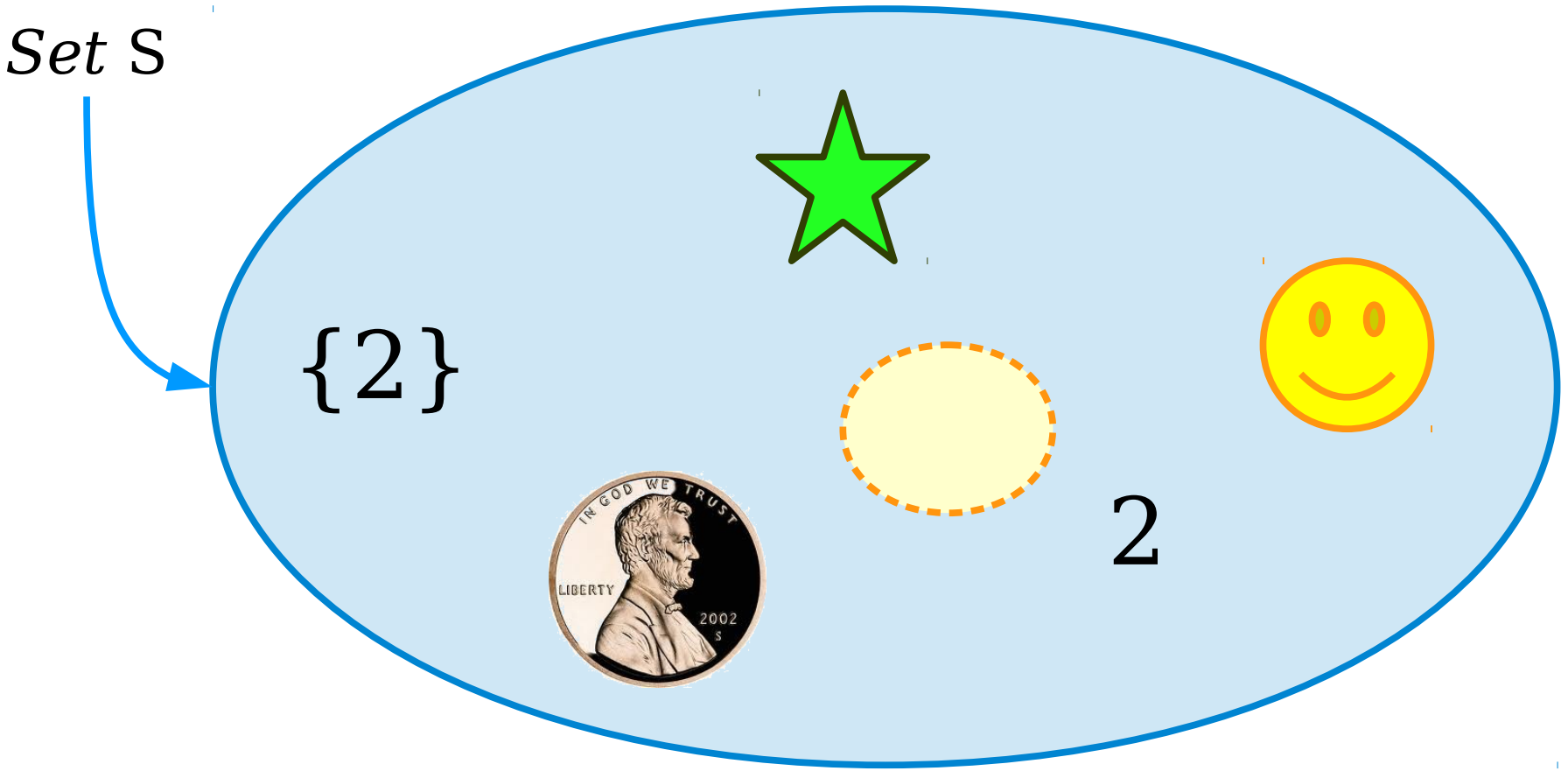


Subsets and Elements



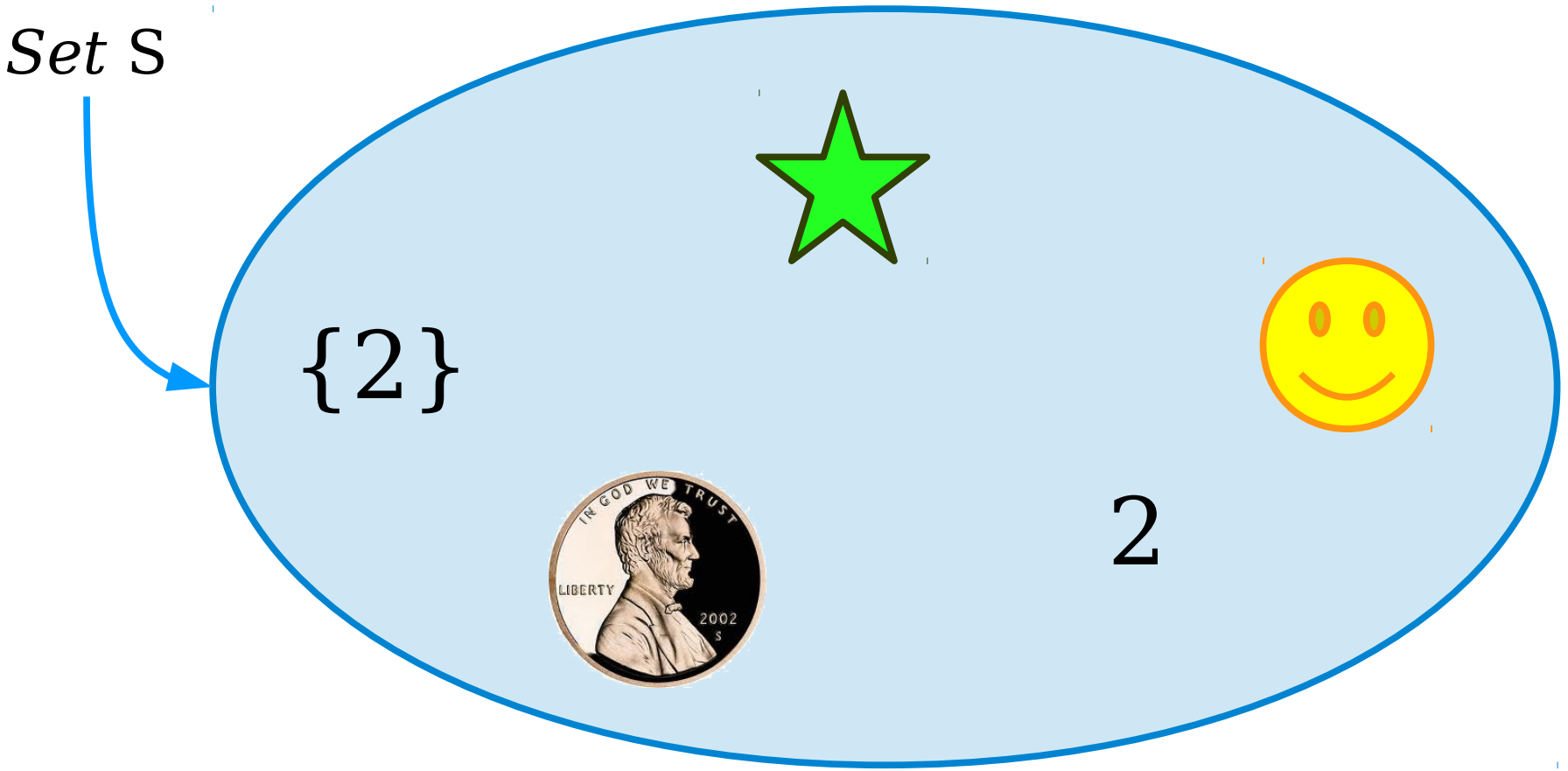
$$\emptyset \subseteq S$$

Subsets and Elements



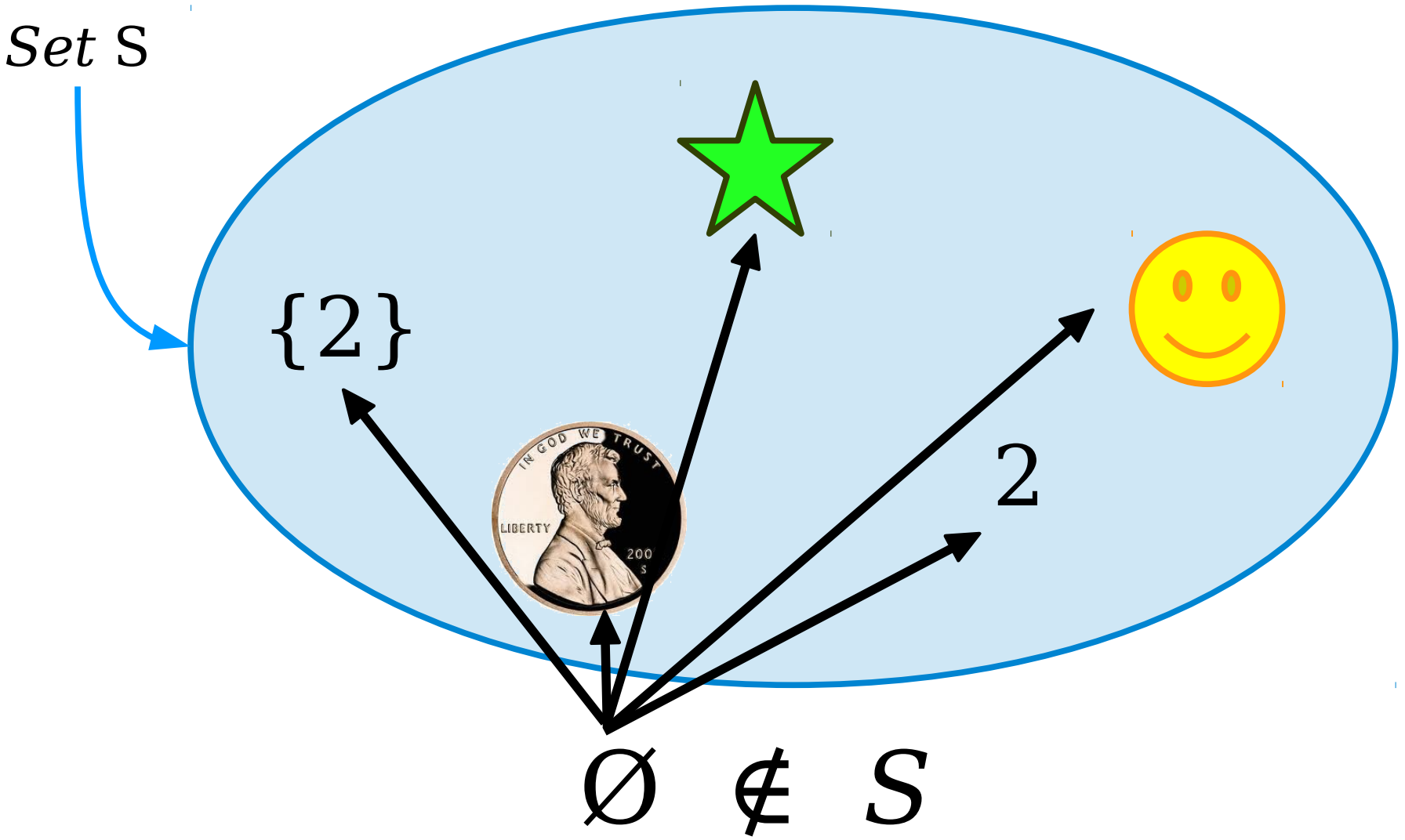
$$\emptyset \subseteq S$$

Subsets and Elements



$$\emptyset \notin S$$

Subsets and Elements



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

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

The notation $\wp(S)$ denotes the ***power set*** of S (the set of all subsets of S).

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

What is $\wp(\emptyset)$?

Answer: $\{\emptyset\}$

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

Cardinality

Cardinality

Cardinality

- The ***cardinality*** of a set is the number of elements it contains.
- If S is a set, we denote its cardinality by writing $|S|$.
- Examples:
  - $|\{a, b, c, d, e\}| = 5$
  - $|\{\{a, b\}, \{c, d, e, f, g\}, \{h\}\}| = 3$
  - $|\{1, 2, 3, 3, 3, 3, 3\}| = 3$
  - $|\{ n \in \mathbb{N} \mid n < 137\}| = 137$

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.

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 $\aleph_0 = |\mathbb{N}|$.
  - $\aleph_0$ is pronounced “aleph-zero,” “aleph-nought,” or “aleph-null.”

Consider the set

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

What is $|S|$?

How Big Are These Sets?

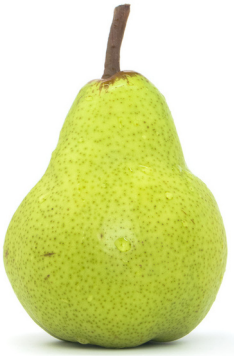
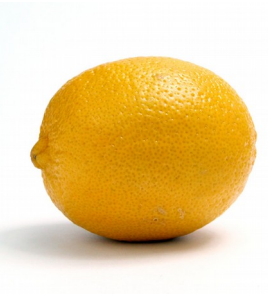
{

,

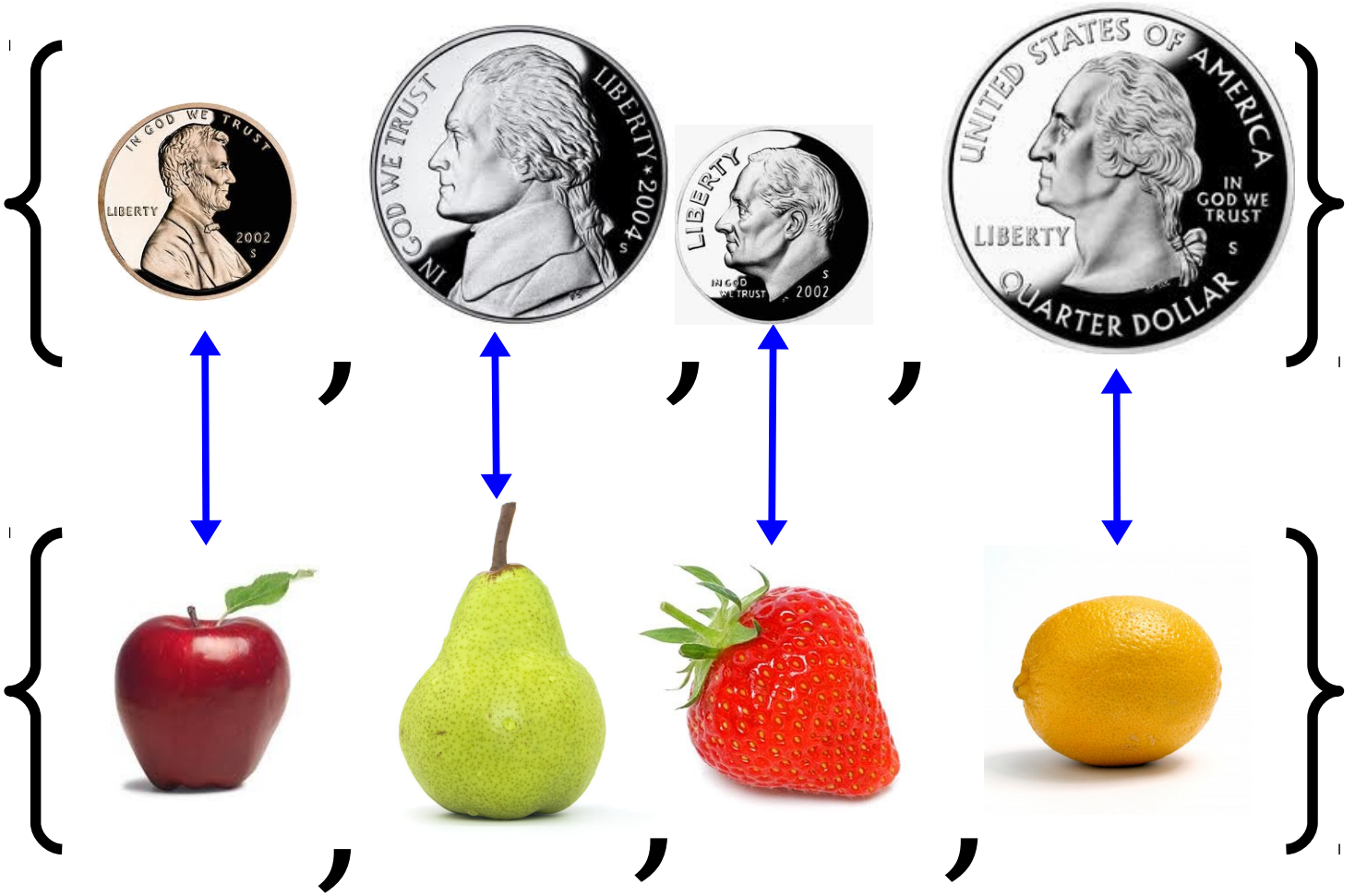
,

,

}

{
, 
,
, 
}

How Big Are These Sets?

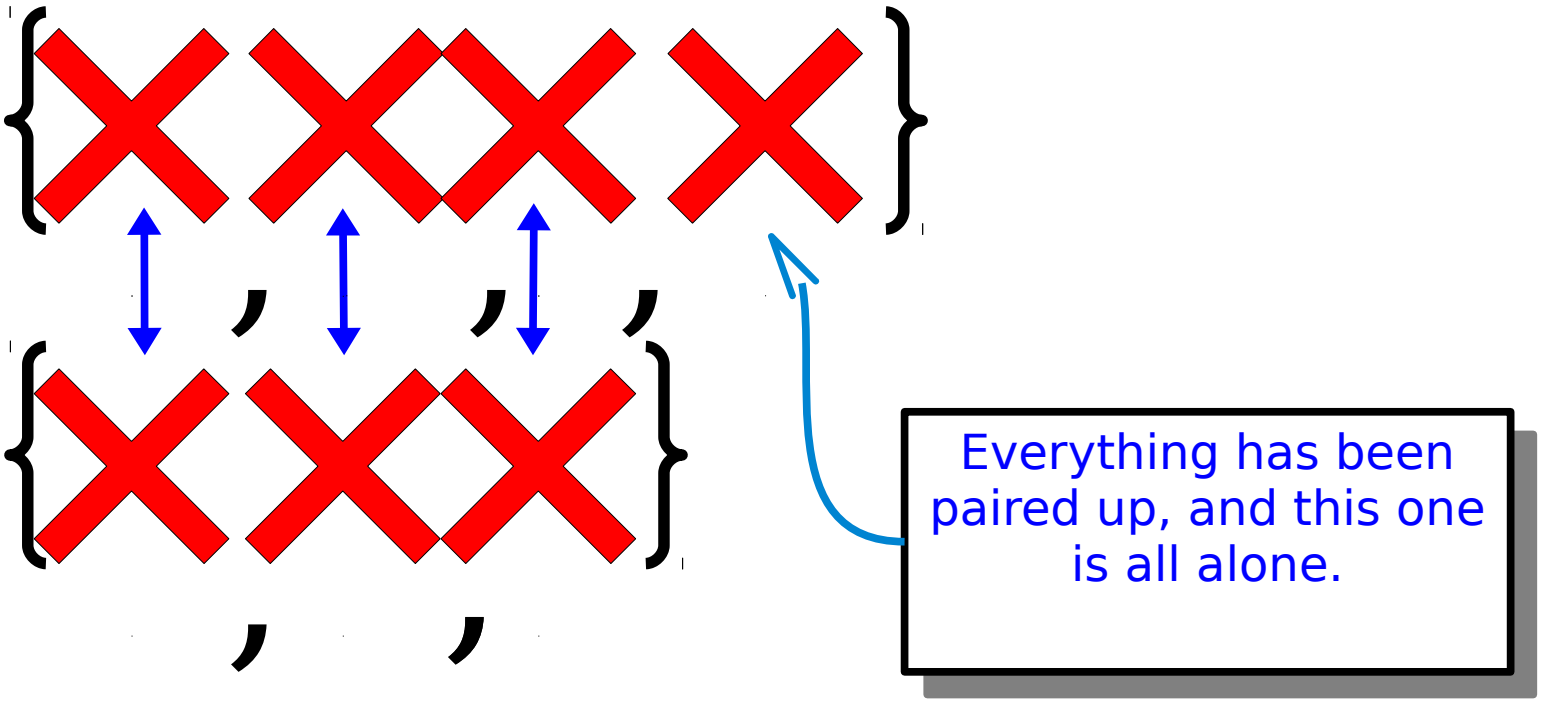


Comparing Cardinalities

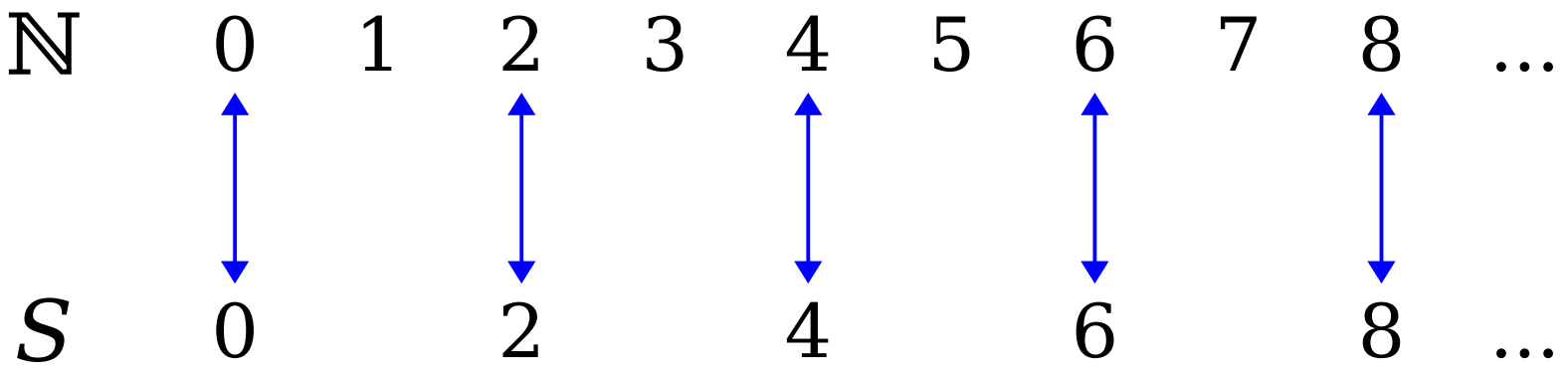
- *By definition,* two sets have the same size if there is a way to pair their elements off without leaving any elements uncovered.

Comparing Cardinalities

- *By definition,* two sets have the same size if there is a way to pair their elements off without leaving any elements uncovered.
- The intuition:



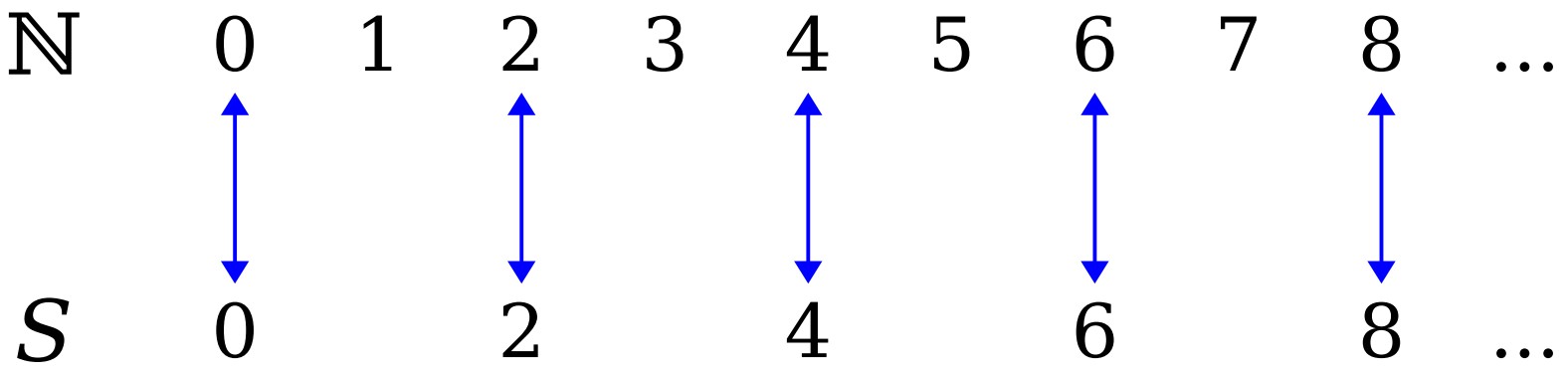
Infinite Cardinalities



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

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

Infinite Cardinalities



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

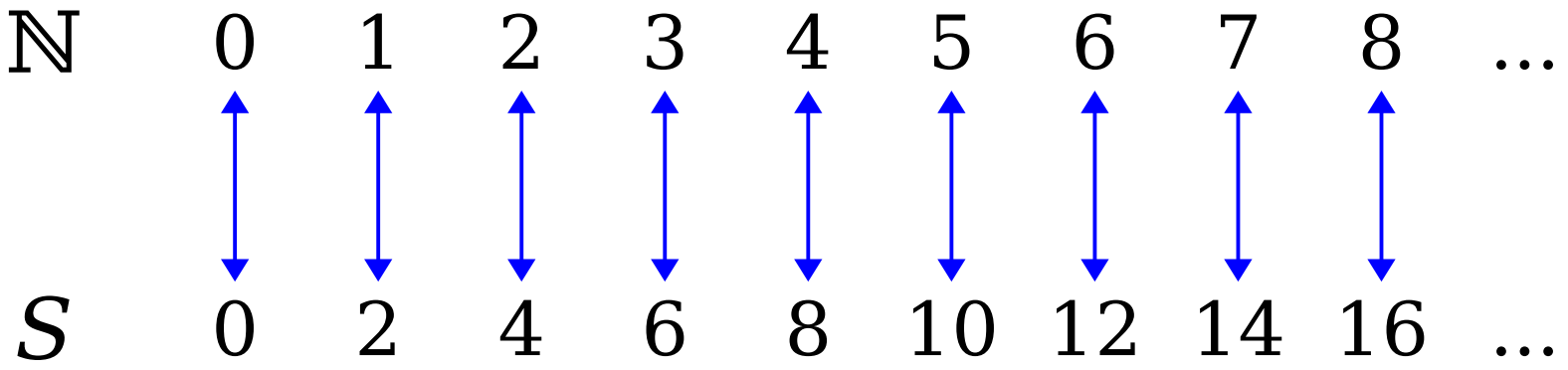
Two sets have the same size if *there is a way* to pair their elements off without leaving any elements uncovered

Infinite Cardinalities

$\mathbb{N}$	0	1	2	3	4	5	6	7	8	...
S	0		2		4		6		8	...

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

Infinite Cardinalities



$$n \leftrightarrow 2n$$

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

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

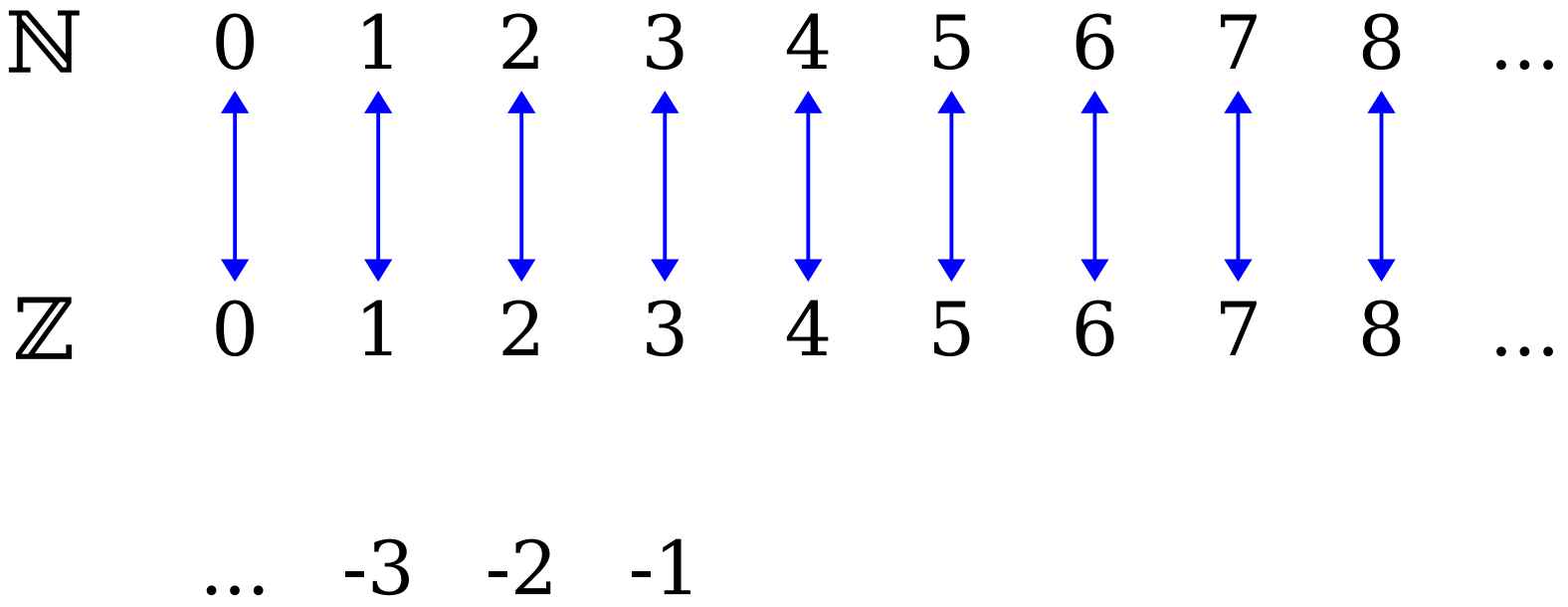
Infinite Cardinalities

$\mathbb{N}$	0	1	2	3	4	5	6	7	8	...
$\mathbb{Z}$	...	-3	-2	-1	0	1	2	3	4	...

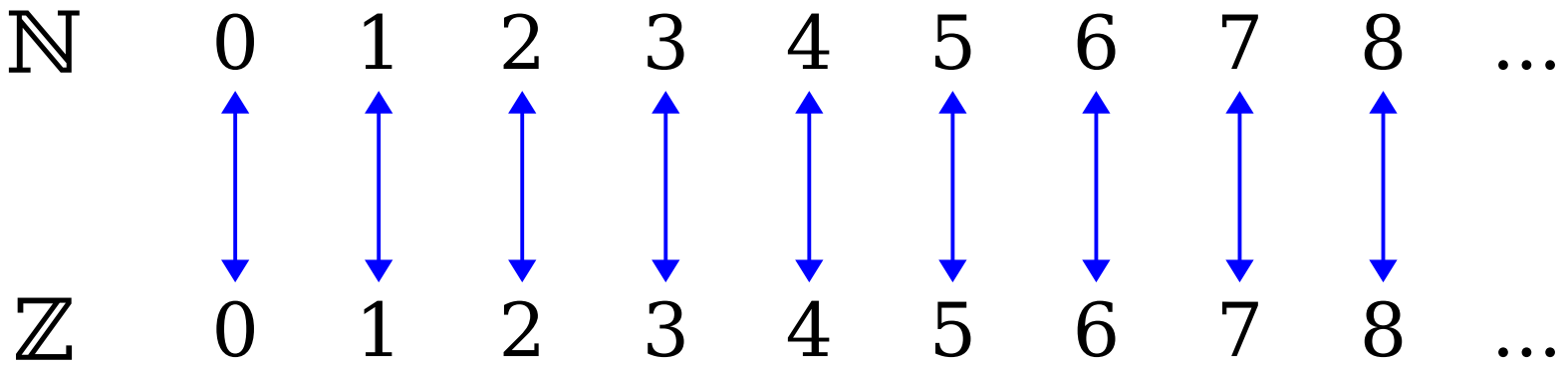
Infinite Cardinalities

$\mathbb{N}$	0	1	2	3	4	5	6	7	8	...
$\mathbb{Z}$					0	1	2	3	4	...
	...	-3	-2	-1						

Infinite Cardinalities



Infinite Cardinalities



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

Infinite Cardinalities

$\mathbb{N}$	0	1	2	3	4	5	6	7	8	...
$\mathbb{Z}$	...	-3	-2	-1	0	1	2	3	4	...

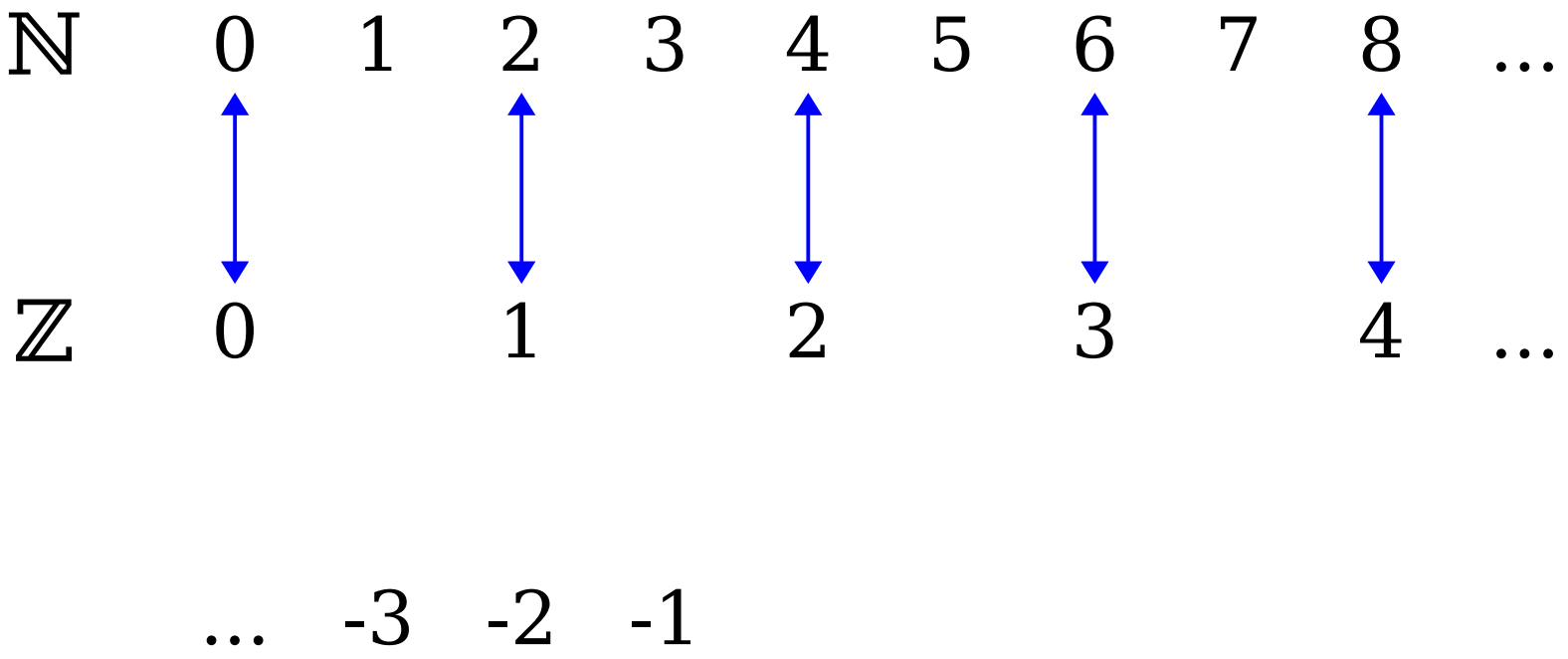
Infinite Cardinalities

$\mathbb{N}$	0	1	2	3	4	5	6	7	8	...
$\mathbb{Z}$										
	...	-3	-2	-1	0	1	2	3	4	...

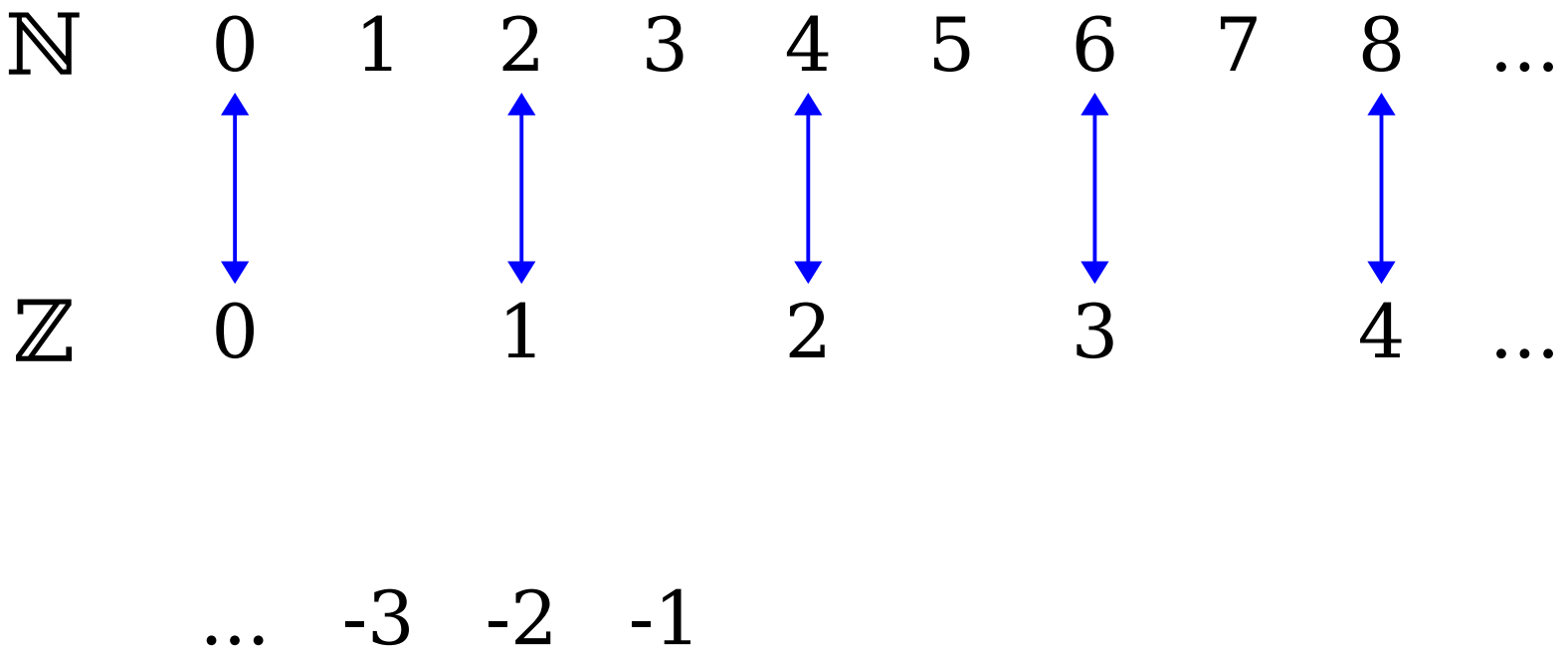
Infinite Cardinalities

$\mathbb{N}$	0	1	2	3	4	5	6	7	8	...
$\mathbb{Z}$	0		1		2		3		4	...
	...	-3	-2	-1						

Infinite Cardinalities

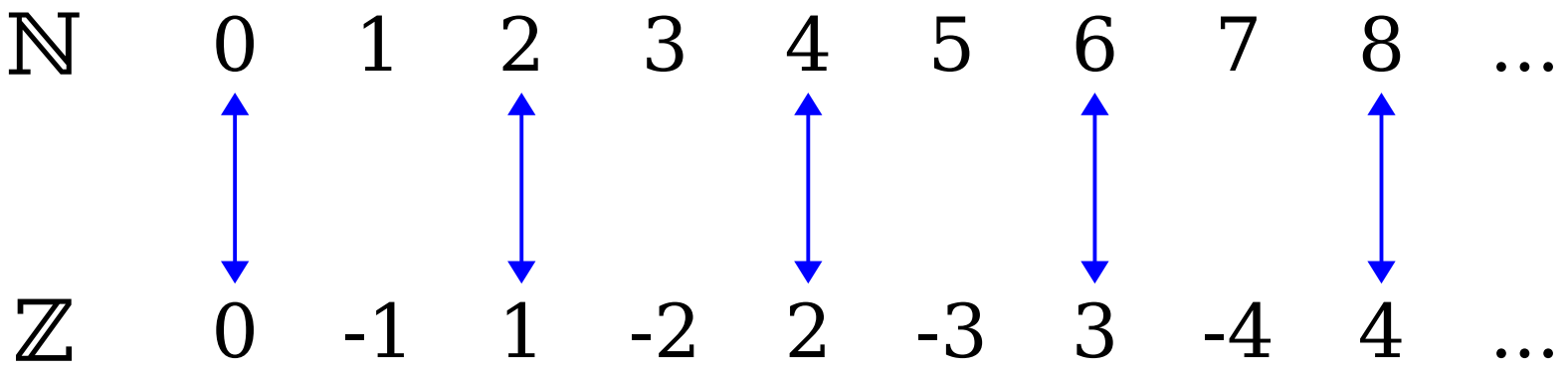


Infinite Cardinalities



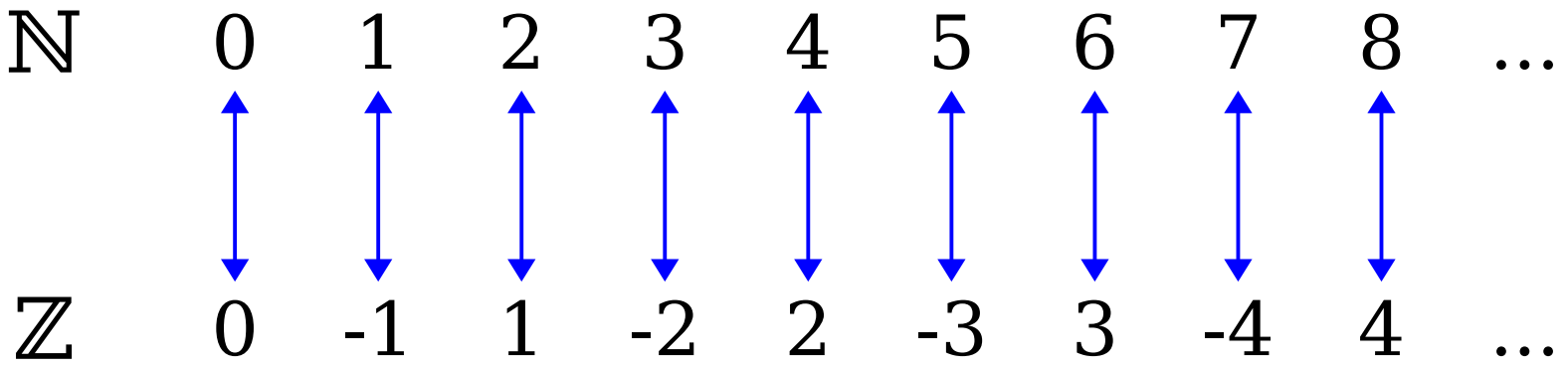
Pair nonnegative integers with even natural numbers.

Infinite Cardinalities



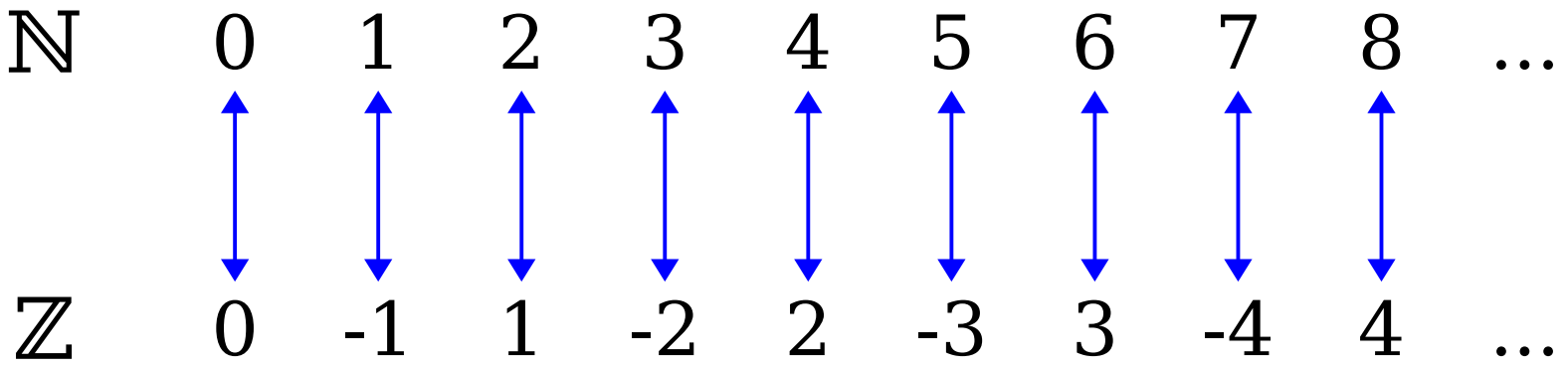
Pair nonnegative integers with even natural numbers.

Infinite Cardinalities



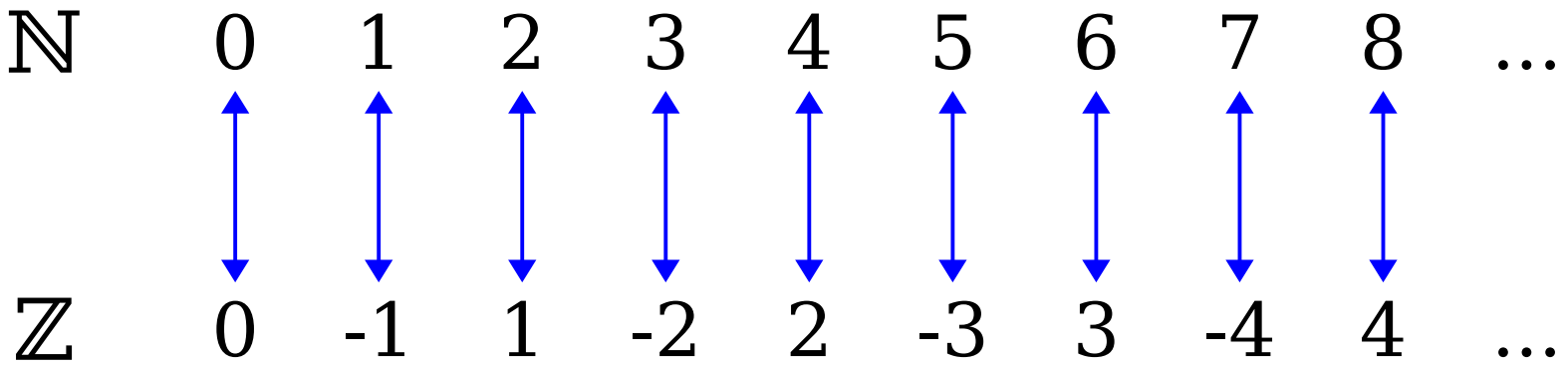
Pair nonnegative integers with even natural numbers.

Infinite Cardinalities



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

Infinite Cardinalities



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

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

Important Question:

Do all infinite sets have
the same cardinality?

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

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

$$|S| < |\wp(S)|$$

$$S = \{ \text{penny}, \text{nickel}, \text{button} \}$$

$$\wp(S) = \{ \emptyset, \{\text{penny}\}, \{\text{nickel}\}, \{\text{button}\}, \{\text{penny}, \text{nickel}\}, \{\text{penny}, \text{button}\}, \{\text{nickel}, \text{button}\}, \{\text{penny}, \text{nickel}, \text{button}\} \}$$

$$|S| < |\wp(S)|$$

$$S = \{a, b, c, d\}$$

$$\wp(S) = \{$$
$$\emptyset,$$
$$\{a\}, \{b\}, \{c\}, \{d\},$$
$$\{a, b\}, \{a, c\}, \{a, d\}, \{b, c\}, \{b, d\}, \{c, d\}$$
$$\{a, b, c\}, \{a, b, d\}, \{a, c, d\}, \{b, c, d\},$$
$$\{a, b, c, d\}$$
$$\}$$

$$|S| < |\wp(S)|$$

If $|S|$ is infinite, what is the relation between $|S|$ and $|\wp(S)|$?

Does $|S| = |\wp(S)|$?

If $|S| = |\wp(S)|$, we can pair up the elements of S and the elements of $\wp(S)$ without leaving anything out.

If $|S| = |\wp(S)|$, we can pair up the elements of S and the elements of $\wp(S)$ without leaving anything out.

If $|S| = |\wp(S)|$, we can pair up the elements of S and the subsets of S without leaving anything out.

If $|S| = |\wp(S)|$, we can pair up the elements of S and the subsets of S without leaving anything out.

If $|S| = |\wp(S)|$, we can pair up the elements of S and the subsets of S without leaving anything out.

What would that look like?

x_0

x_1

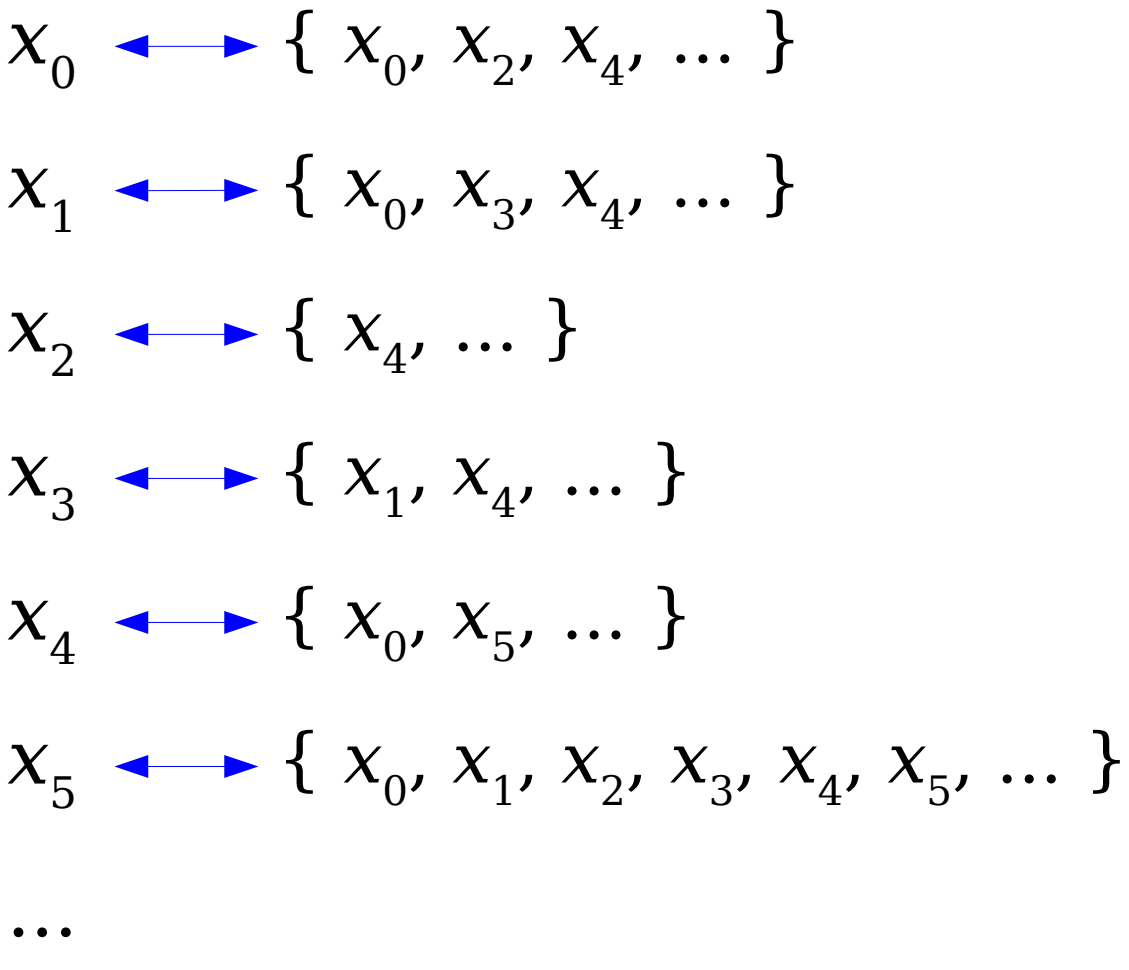
x_2

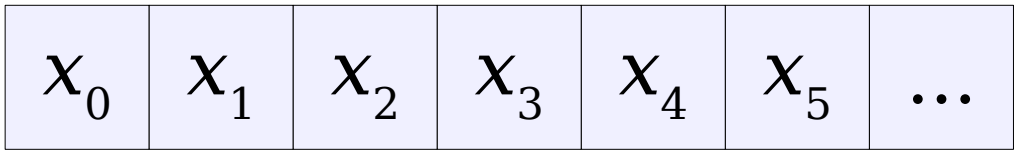
x_3

x_4

x_5

…


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



$x_0 \leftrightarrow \{ x_0, x_2, x_4, \ldots \}$

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

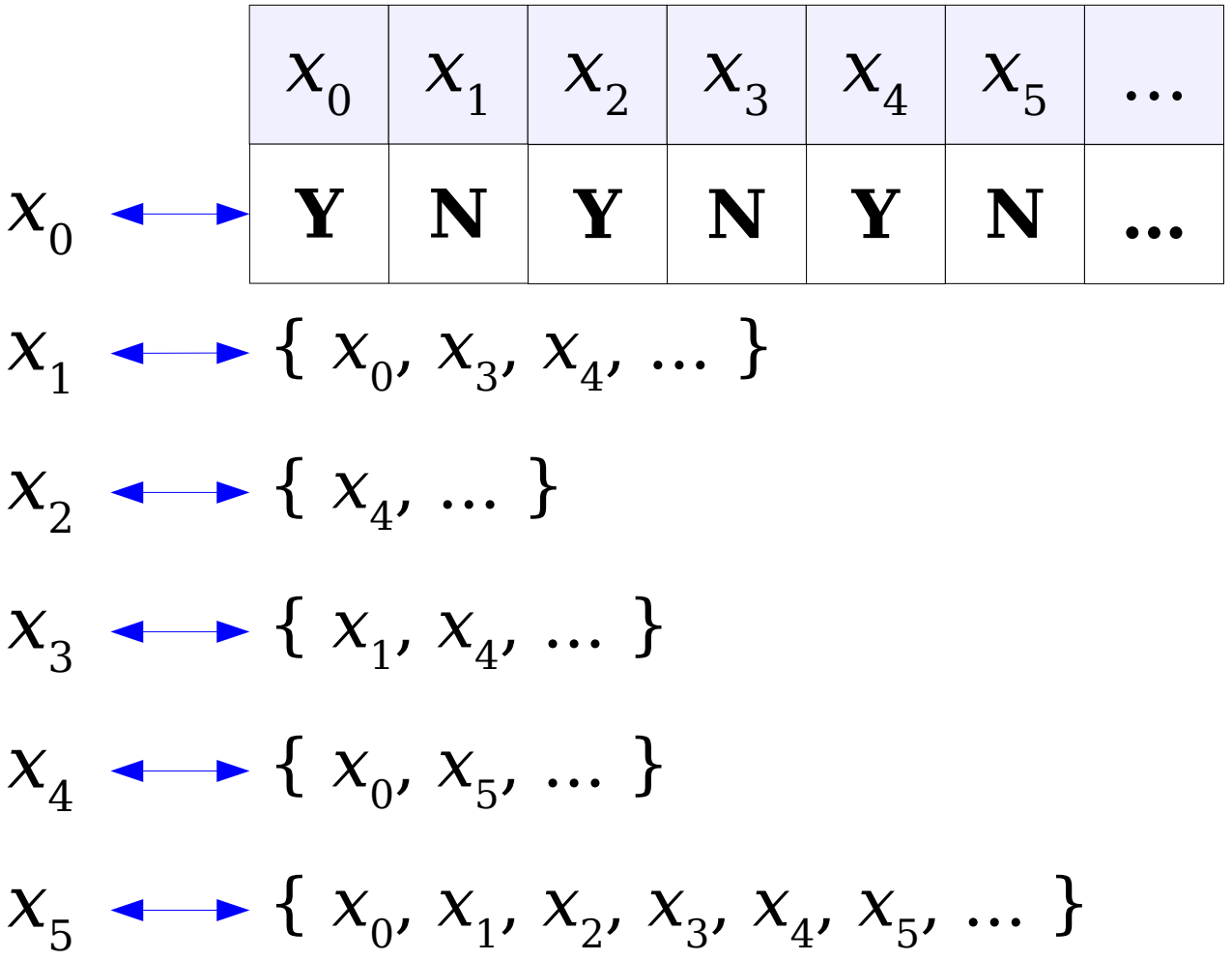
$x_2 \leftrightarrow \{ x_4, \ldots \}$

$x_3 \leftrightarrow \{ x_1, x_4, \ldots \}$

$x_4 \leftrightarrow \{ x_0, x_5, \ldots \}$

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

…


x_0 x_1 x_2 x_3 x_4 x_5 …
x_0 Y N Y N Y N …
x_1 { x_0, x_3, x_4, … }
x_2 { x_4, … }
x_3 { x_1, x_4, … }
x_4 { x_0, x_5, … }
x_5 { x_0, x_1, x_2, x_3, x_4, x_5, … }
…

	x_0	x_1	x_2	x_3	x_4	x_5	…
x_0 ⟷	**Y**	**N**	**Y**	**N**	**Y**	**N**	**…**
x_1 ⟷	**Y**	**N**	**N**	**Y**	**Y**	**N**	**…**

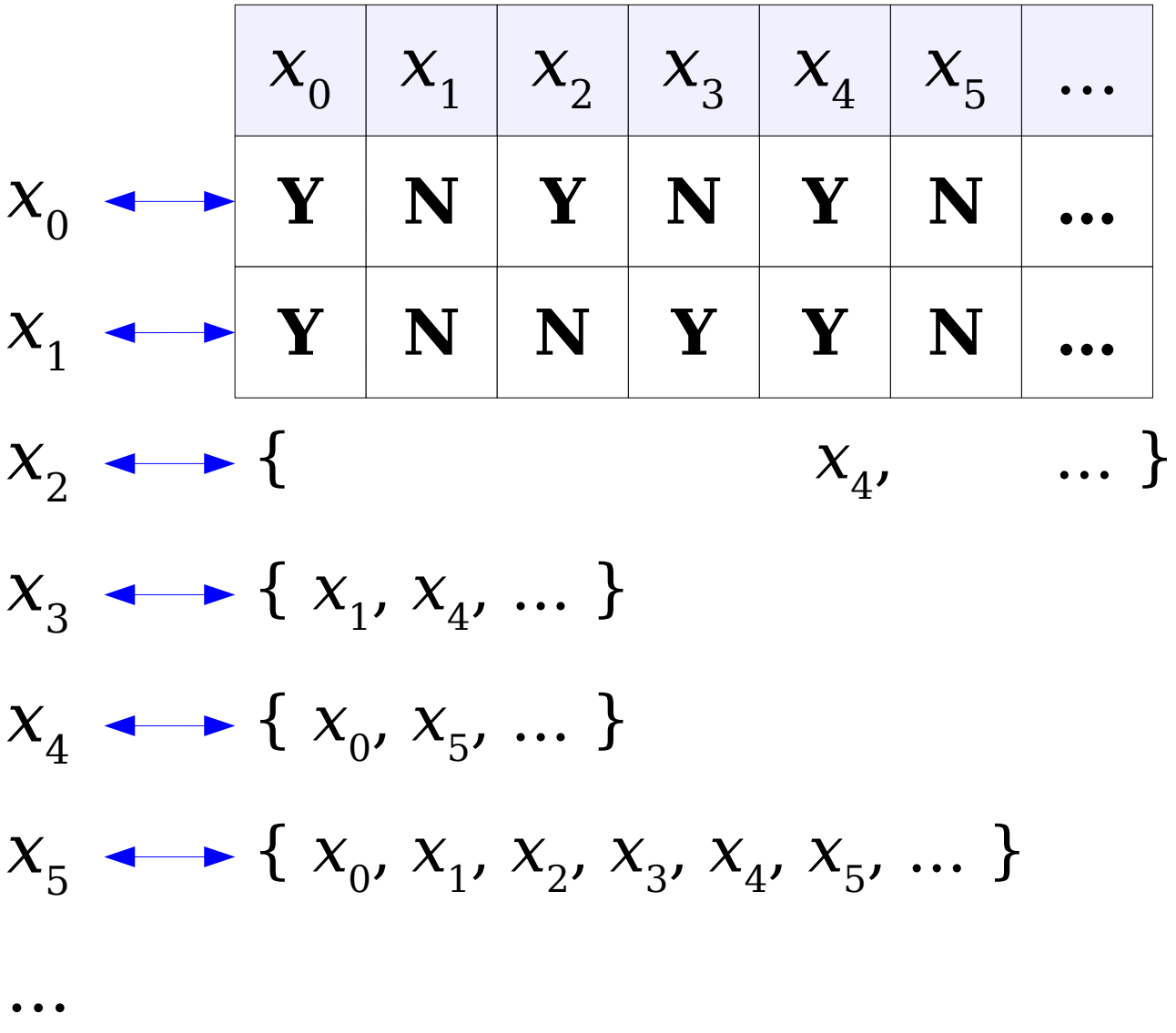
$x_2 \longleftrightarrow \{ x_4, \ldots \}$

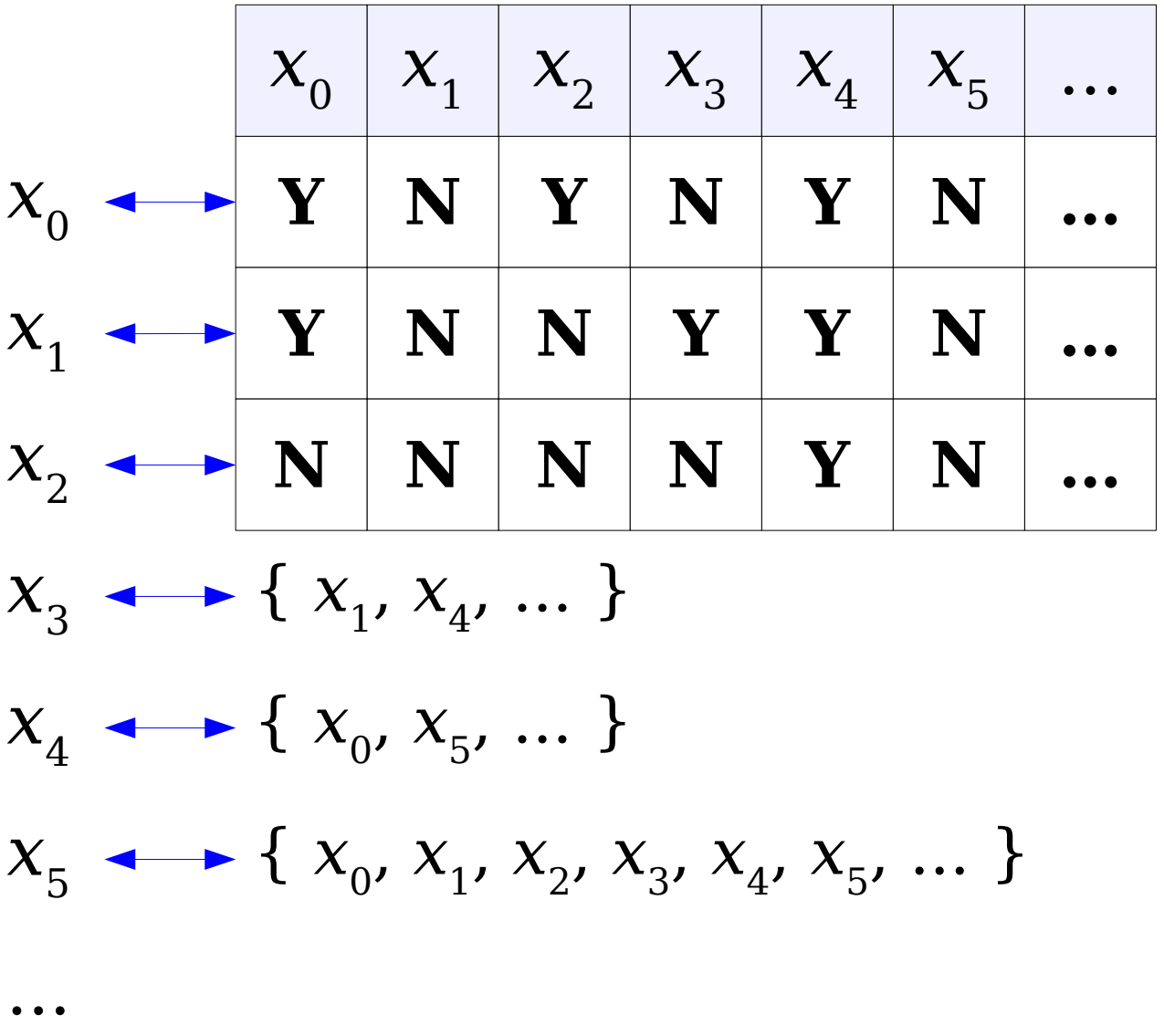
$x_3 \longleftrightarrow \{ x_1, x_4, \ldots \}$

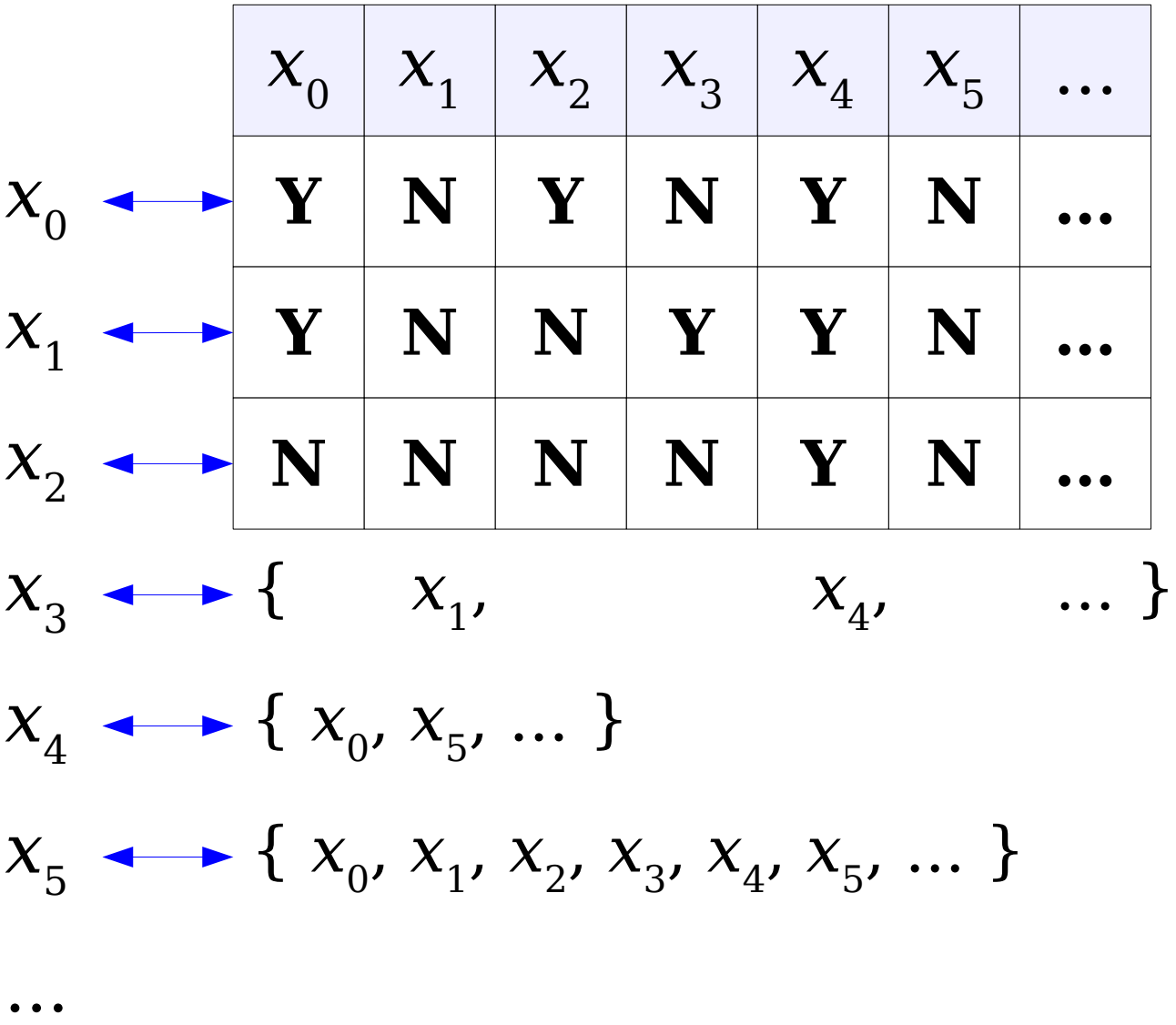
$x_4 \longleftrightarrow \{ x_0, x_5, \ldots \}$

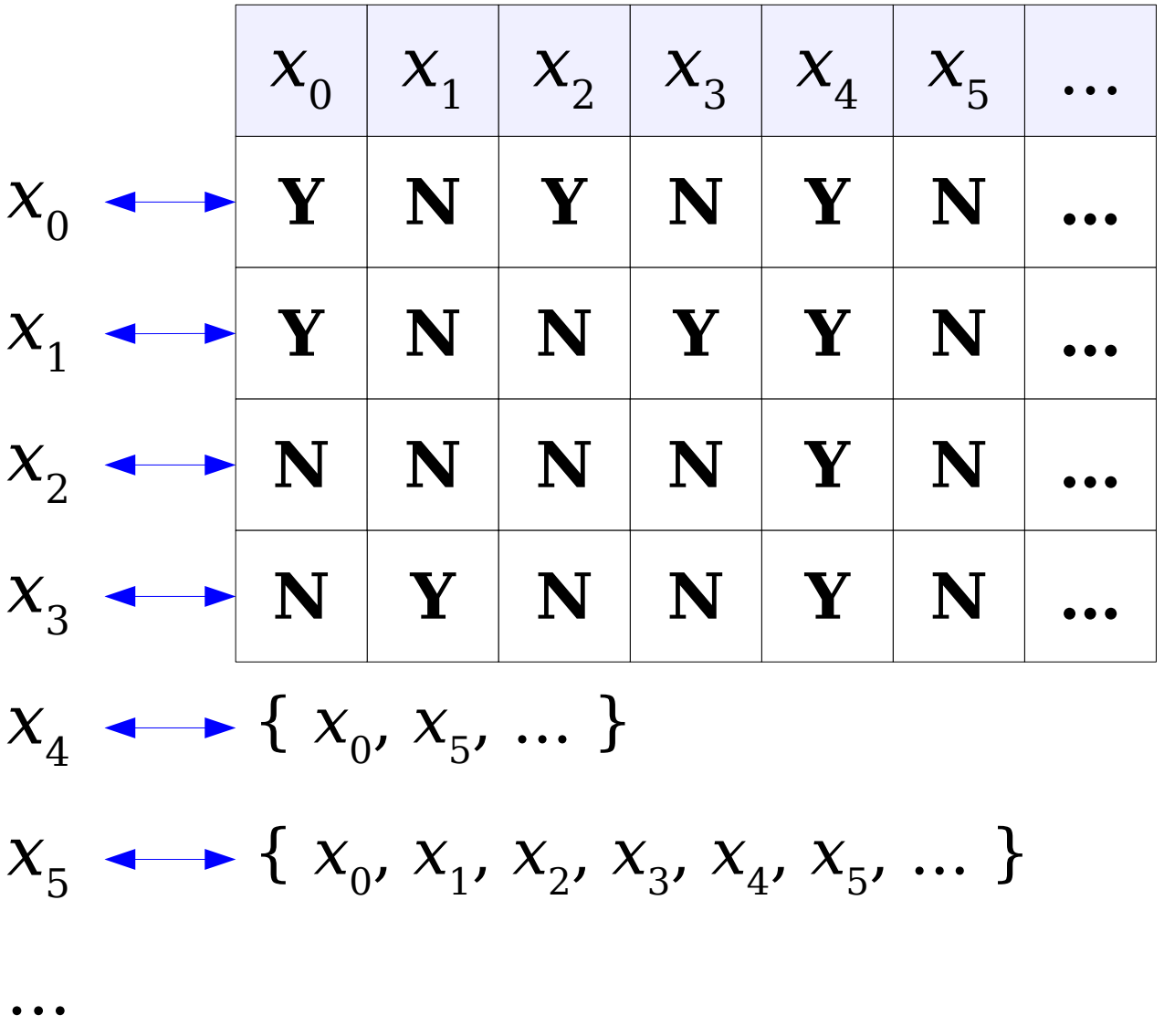
$x_5 \longleftrightarrow \{ x_0, x_1, x_2, x_3, x_4, x_5, \ldots \}$

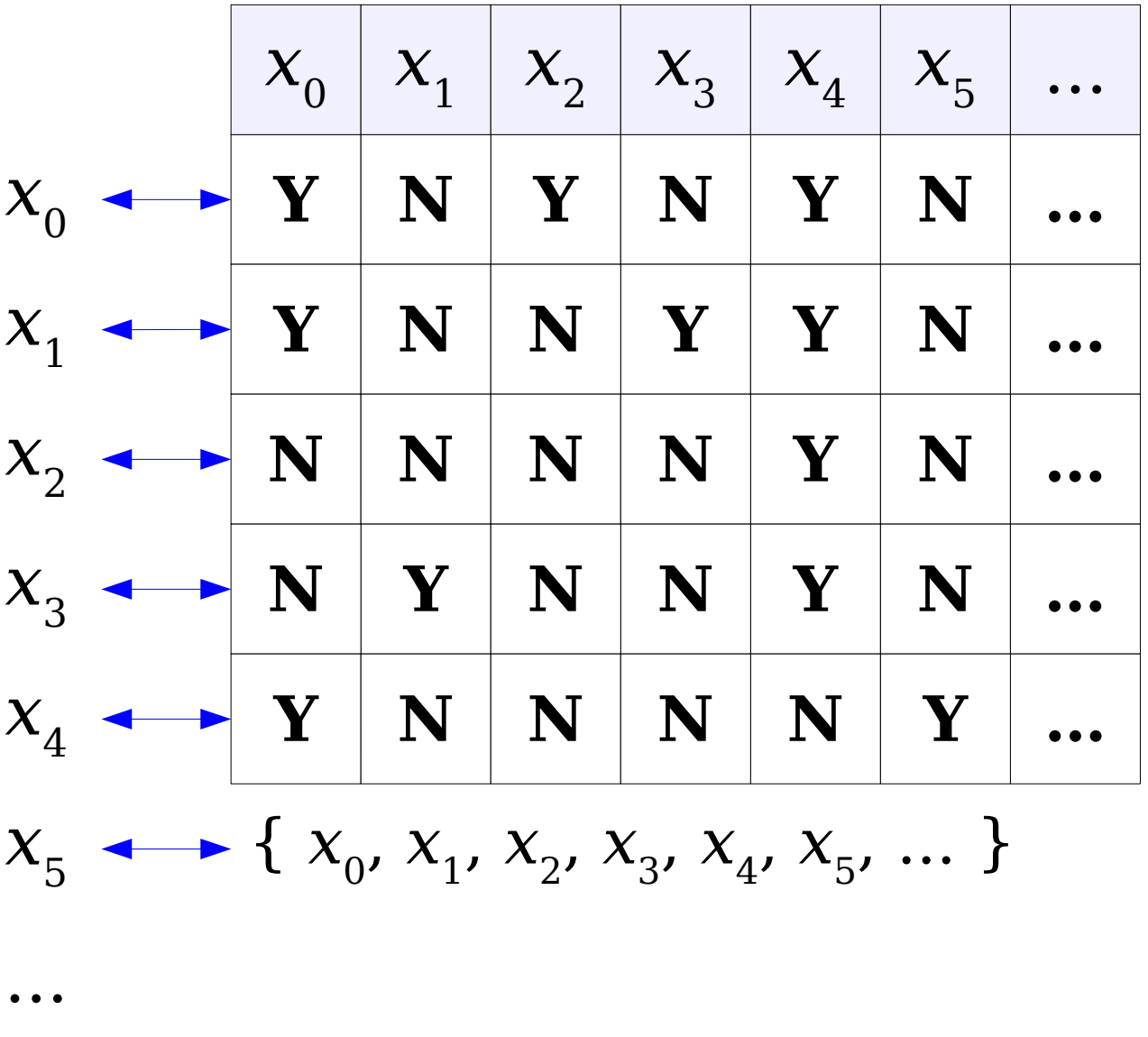
…

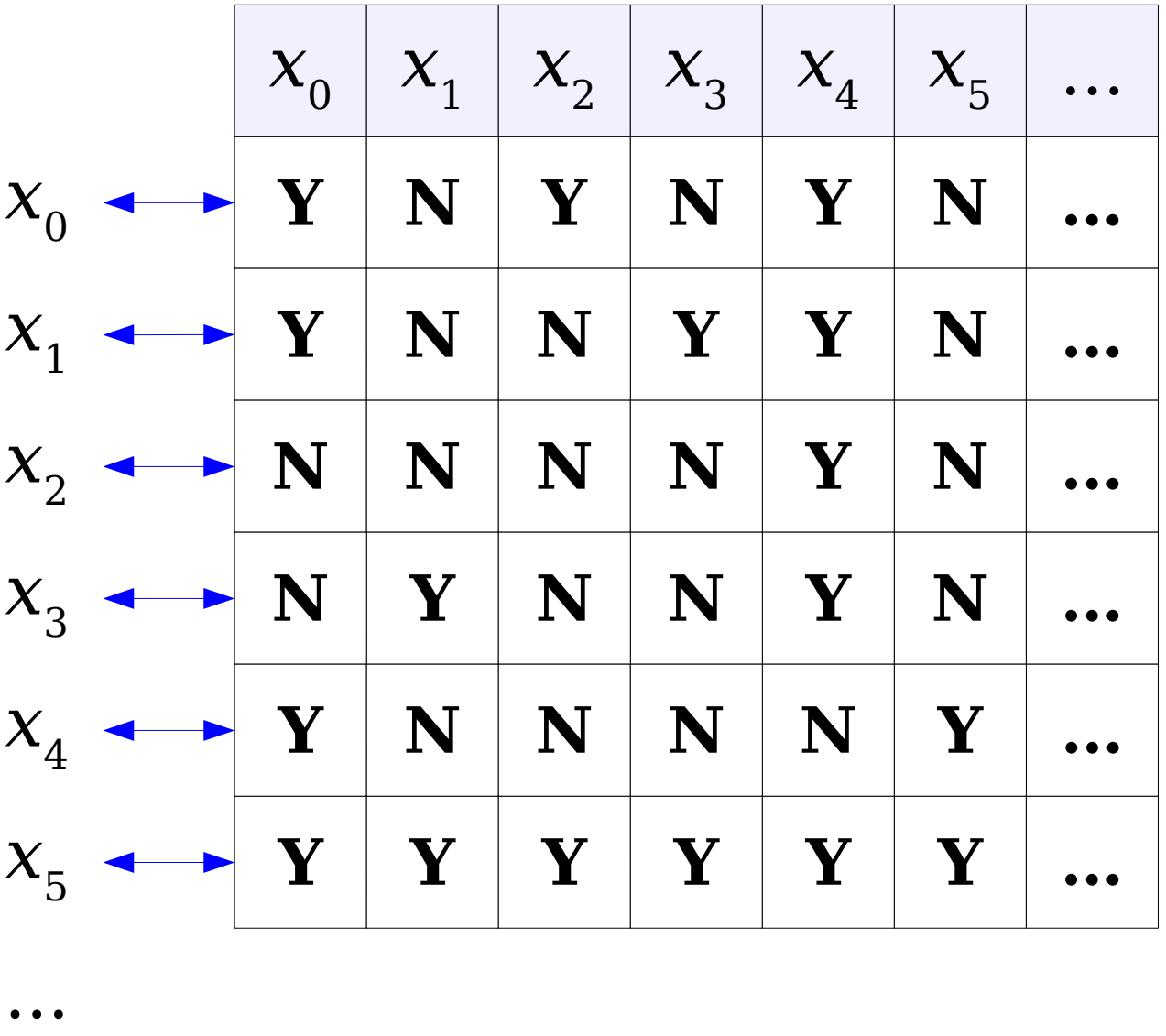

x_0 x_1 x_2 x_3 x_4 x_5 …
x_0 ↔ Y N Y N Y N …
x_1 ↔ Y N N Y Y N …
x_2 ↔ { x_4, … }
x_3 ↔ { x_1, x_4, … }
x_4 ↔ { x_0, x_5, … }
x_5 ↔ { x_0, x_1, x_2, x_3, x_4, x_5, … }
…

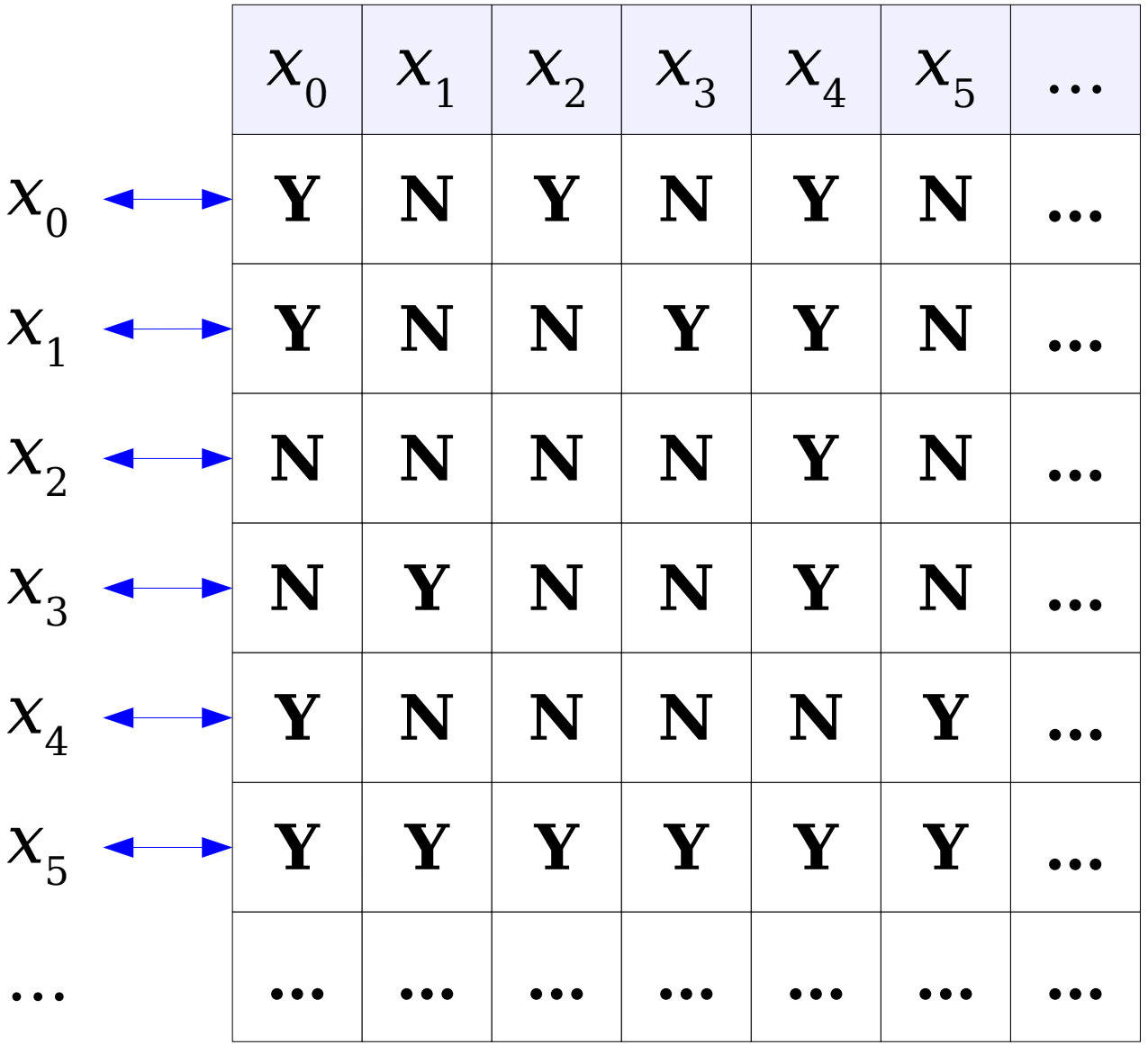

x_0 x_1 x_2 x_3 x_4 x_5 …
x_0 Y N Y N Y N …
x_1 Y N N Y Y N …
x_2 N N N N Y N …
x_3 { x_1, x_4, … }
x_4 { x_0, x_5, … }
x_5 { x_0, x_1, x_2, x_3, x_4, x_5, … }
…


x_0 x_1 x_2 x_3 x_4 x_5 …
x_0 Y N Y N Y N …
x_1 Y N N Y Y N …
x_2 N N N N Y N …
x_3 { x_1, x_4, … }
x_4 { x_0, x_5, … }
x_5 { x_0, x_1, x_2, x_3, x_4, x_5, … }
…


x_0 x_1 x_2 x_3 x_4 x_5 …
x_0 Y N Y N Y N …
x_1 Y N N Y Y N …
x_2 N N N N Y N …
x_3 N Y N N Y N …
x_4 { x_0, x_5, … }
x_5 { x_0, x_1, x_2, x_3, x_4, x_5, … }
…


x_0 x_1 x_2 x_3 x_4 x_5 …
x_0 Y N Y N Y N …
x_1 Y N N Y Y N …
x_2 N N N N Y N …
x_3 N Y N N Y N …
x_4 Y N N N N Y …
x_5 { x_0, x_1, x_2, x_3, x_4, x_5, … }
…


x_0 x_1 x_2 x_3 x_4 x_5 …
x_0 Y N Y N Y N …
x_1 Y N N Y Y N …
x_2 N N N N Y N …
x_3 N Y N N Y N …
x_4 Y N N N N Y …
x_5 Y Y Y Y Y Y …
…

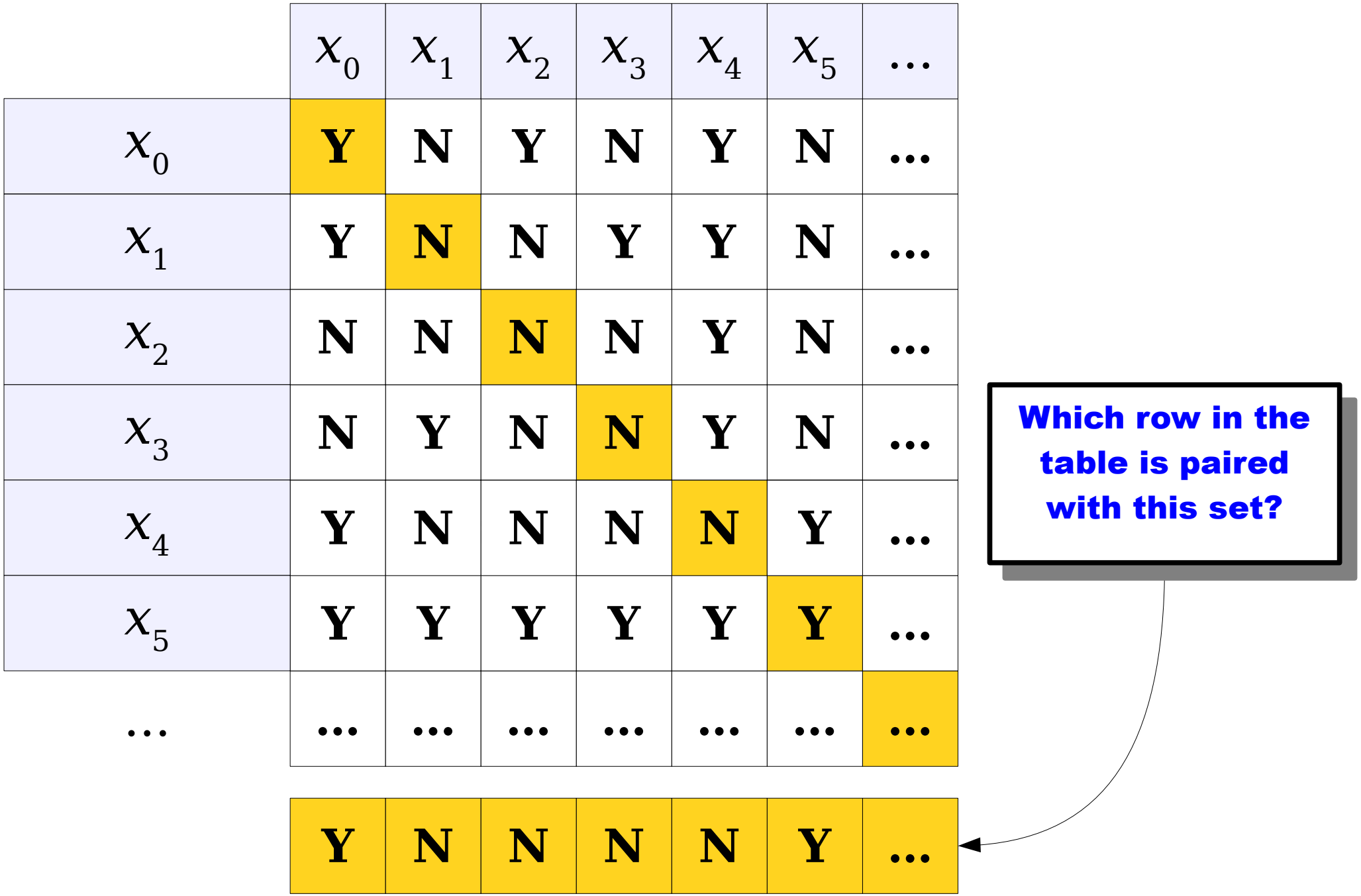

x_0 x_1 x_2 x_3 x_4 x_5 …
x_0 Y N Y N Y N …
x_1 Y N N Y Y N …
x_2 N N N N Y N …
x_3 N Y N N Y N …
x_4 Y N N N N Y …
x_5 Y Y Y Y Y Y …
… … … … … … … …

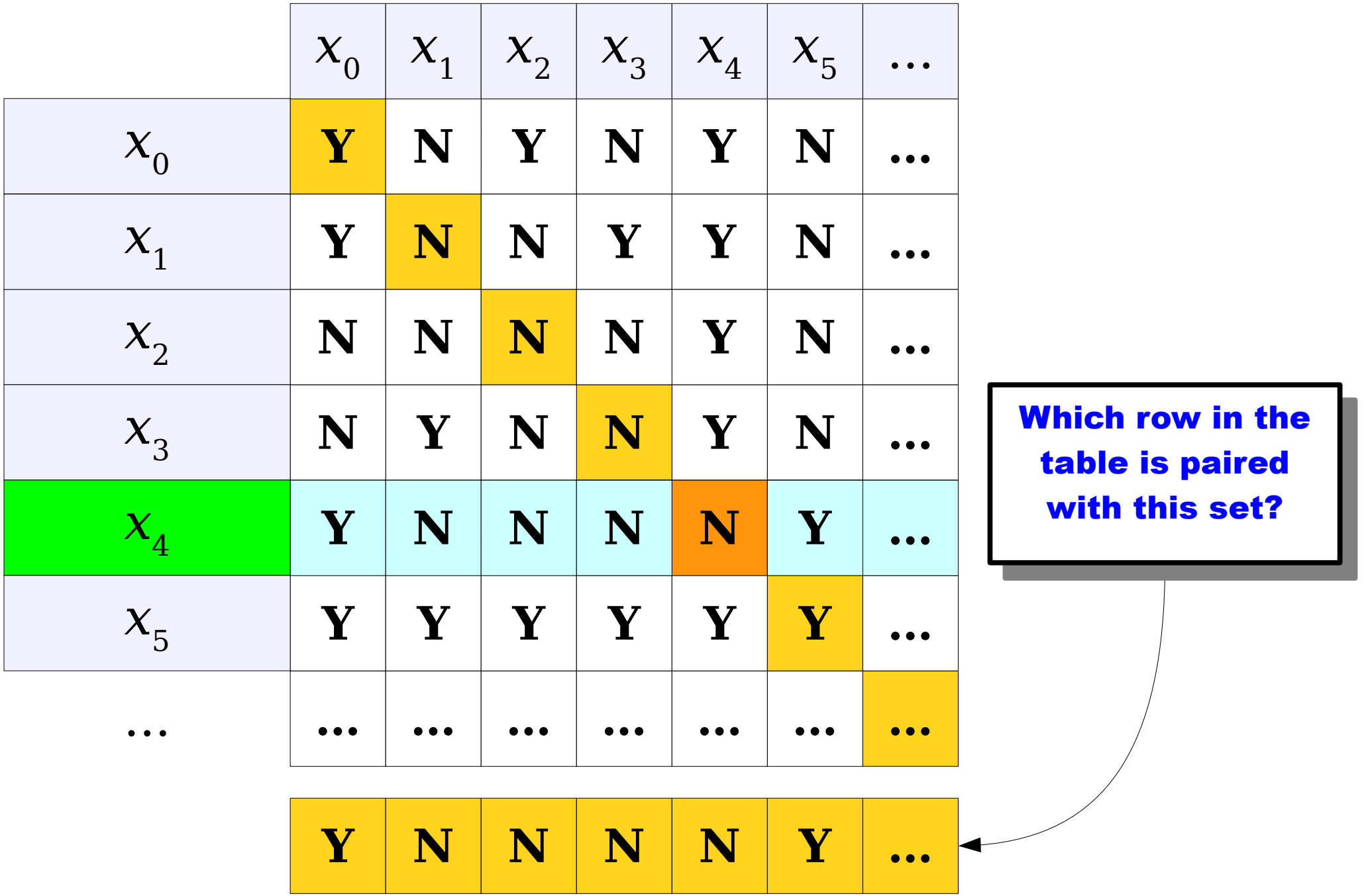
	x_0	x_1	x_2	x_3	x_4	x_5	…
x_0	**Y**	**N**	**Y**	**N**	**Y**	**N**	…
x_1	**Y**	**N**	**N**	**Y**	**Y**	**N**	…
x_2	**N**	**N**	**N**	**N**	**Y**	**N**	…
x_3	**N**	**Y**	**N**	**N**	**Y**	**N**	…
x_4	**Y**	**N**	**N**	**N**	**N**	**Y**	…
x_5	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**	…
…	…	…	…	…	…	…	…

	x_0	x_1	x_2	x_3	x_4	x_5	…
x_0	**Y**	**N**	**Y**	**N**	**Y**	**N**	…
x_1	**Y**	**N**	**N**	**Y**	**Y**	**N**	…
x_2	**N**	**N**	**N**	**N**	**Y**	**N**	…
x_3	**N**	**Y**	**N**	**N**	**Y**	**N**	…
x_4	**Y**	**N**	**N**	**N**	**N**	**Y**	…
x_5	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**	…
…	…	…	…	…	…	…	…

	x_0	x_1	x_2	x_3	x_4	x_5	…
x_0	**Y**	**N**	**Y**	**N**	**Y**	**N**	…
x_1	**Y**	**N**	**N**	**Y**	**Y**	**N**	…
x_2	**N**	**N**	**N**	**N**	**Y**	**N**	…
x_3	**N**	**Y**	**N**	**N**	**Y**	**N**	…
x_4	**Y**	**N**	**N**	**N**	**N**	**Y**	…
x_5	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**	…
…	…	…	…	…	…	…	…

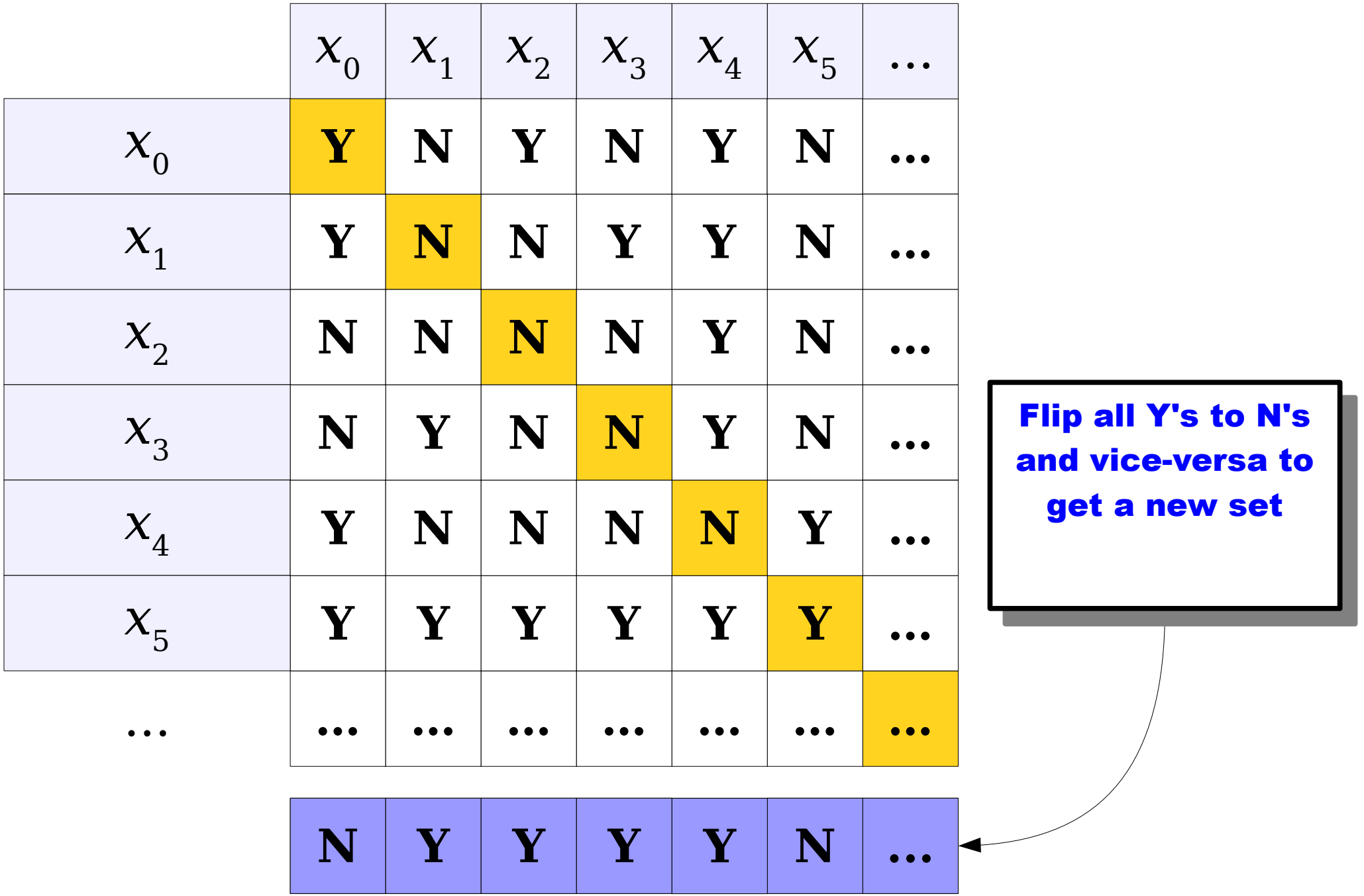
{ **x_0,** **x_5,** … }

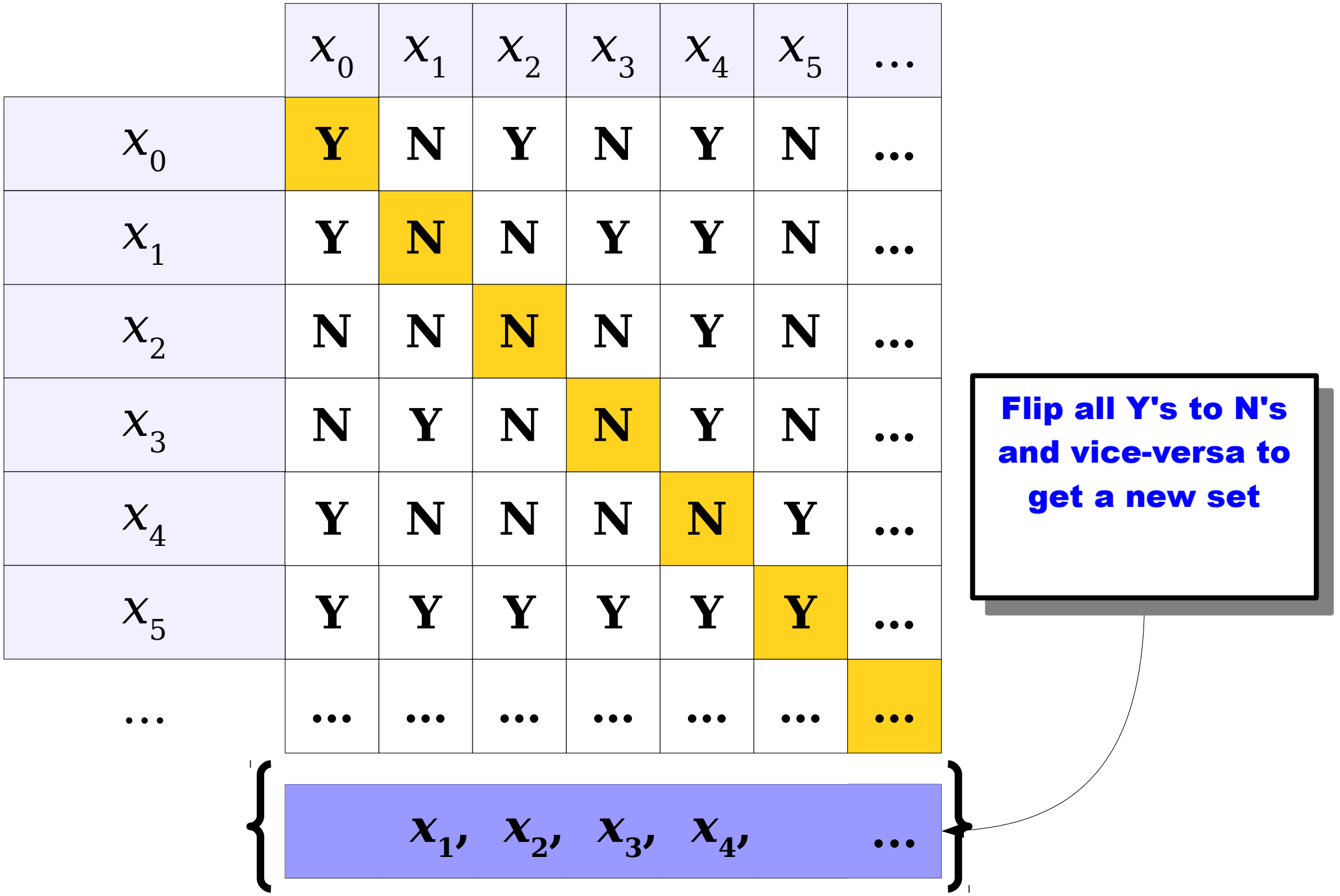

x_0 x_1 x_2 x_3 x_4 x_5 …
x_0 Y N Y N Y N …
x_1 Y N N Y Y N …
x_2 N N N N Y N …
x_3 N Y N N Y N …
x_4 Y N N N N Y …
x_5 Y Y Y Y Y Y …
… … … … … … … …
Y N N N N Y …
Which row in the table is paired with this set?

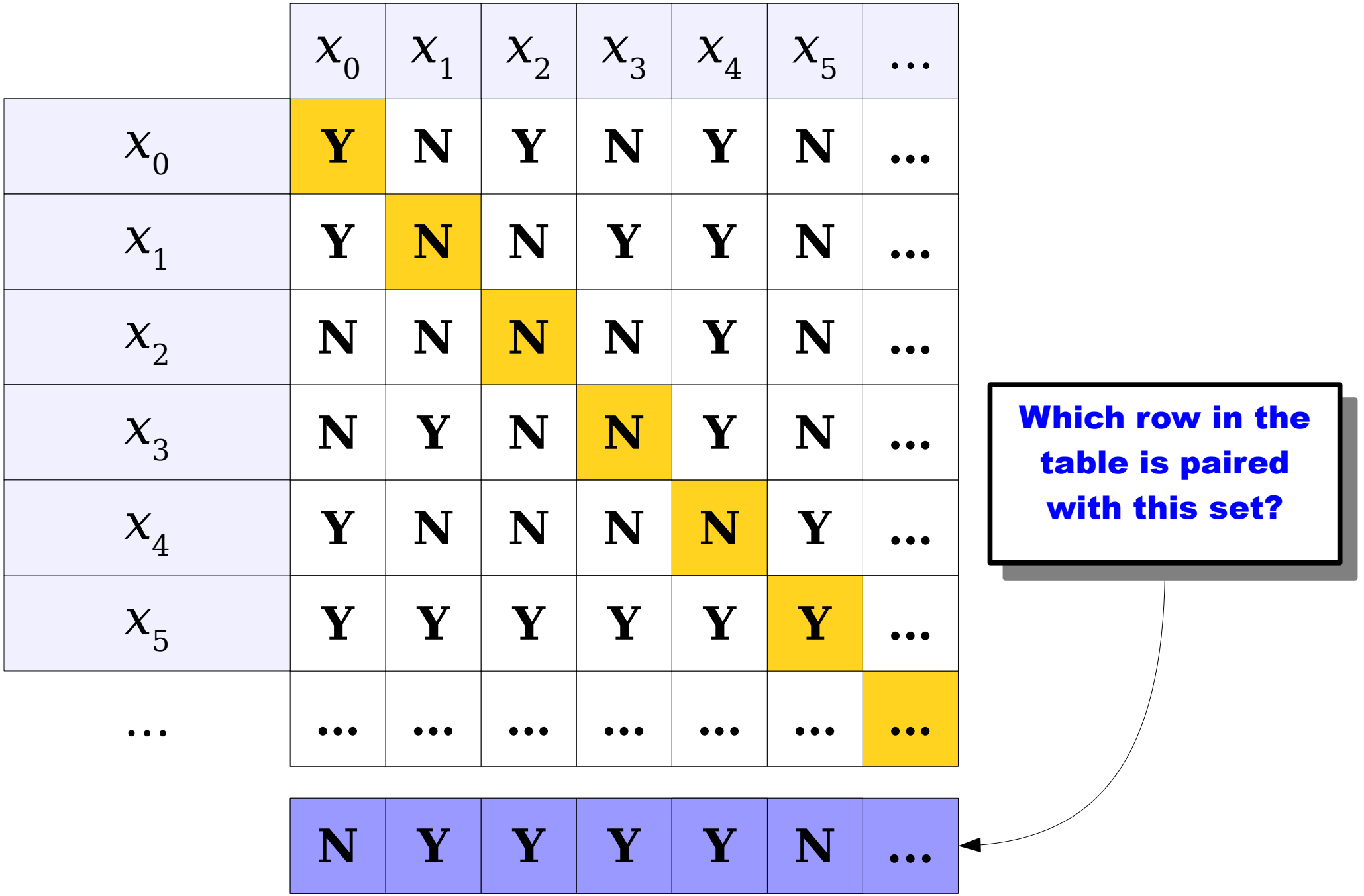

x_0 x_1 x_2 x_3 x_4 x_5 …
x_0 Y N Y N Y N …
x_1 Y N N Y Y N …
x_2 N N N N Y N …
x_3 N Y N N Y N …
x_4 Y N N N N Y …
x_5 Y Y Y Y Y Y …
… … … … … … … …
Y N N N N Y …
Which row in the table is paired with this set?

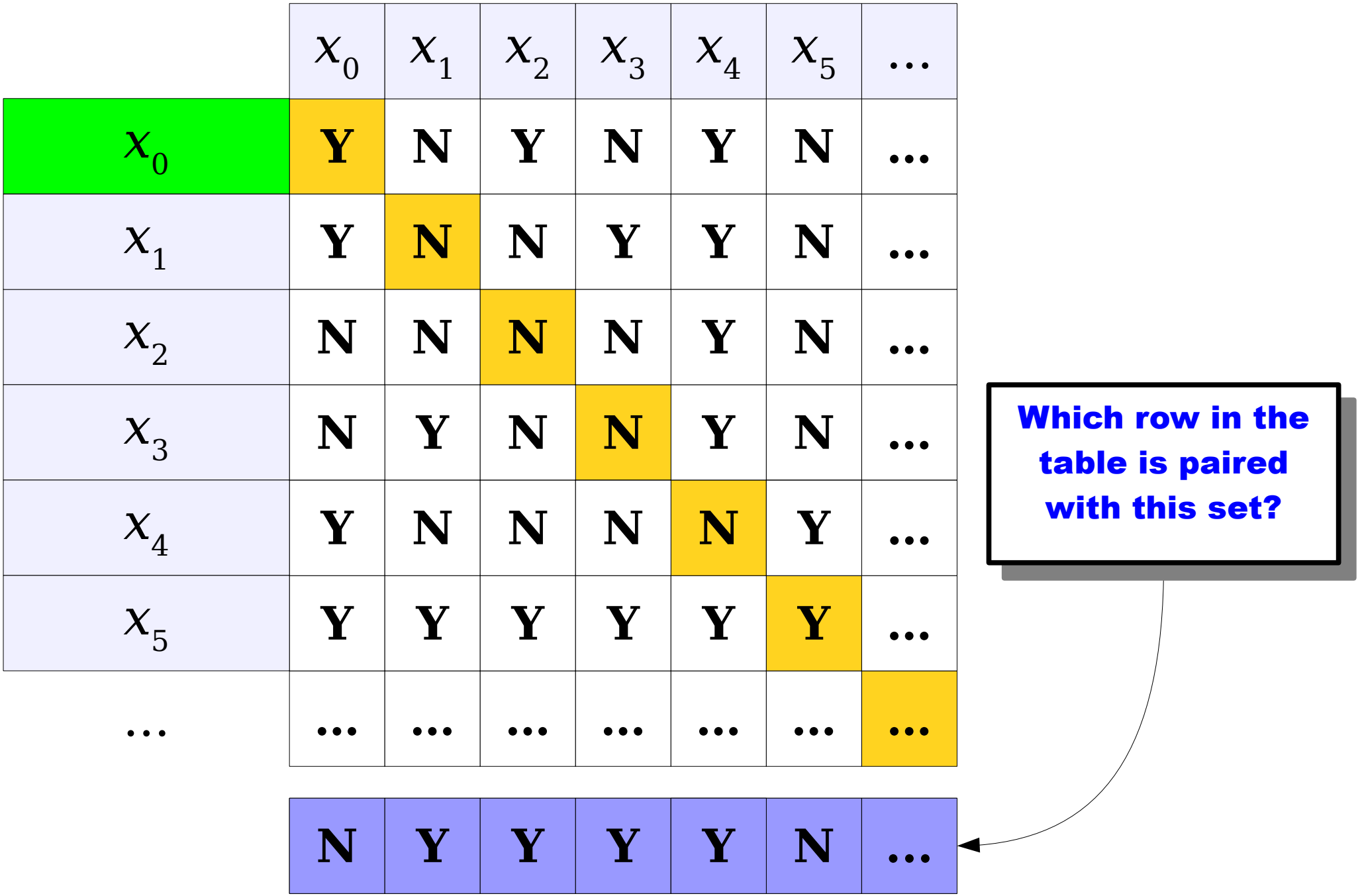
	x_0	x_1	x_2	x_3	x_4	x_5	…
x_0	**Y**	**N**	**Y**	**N**	**Y**	**N**	…
x_1	**Y**	**N**	**N**	**Y**	**Y**	**N**	…
x_2	**N**	**N**	**N**	**N**	**Y**	**N**	…
x_3	**N**	**Y**	**N**	**N**	**Y**	**N**	…
x_4	**Y**	**N**	**N**	**N**	**N**	**Y**	…
x_5	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**	…
…	…	…	…	…	…	…	…

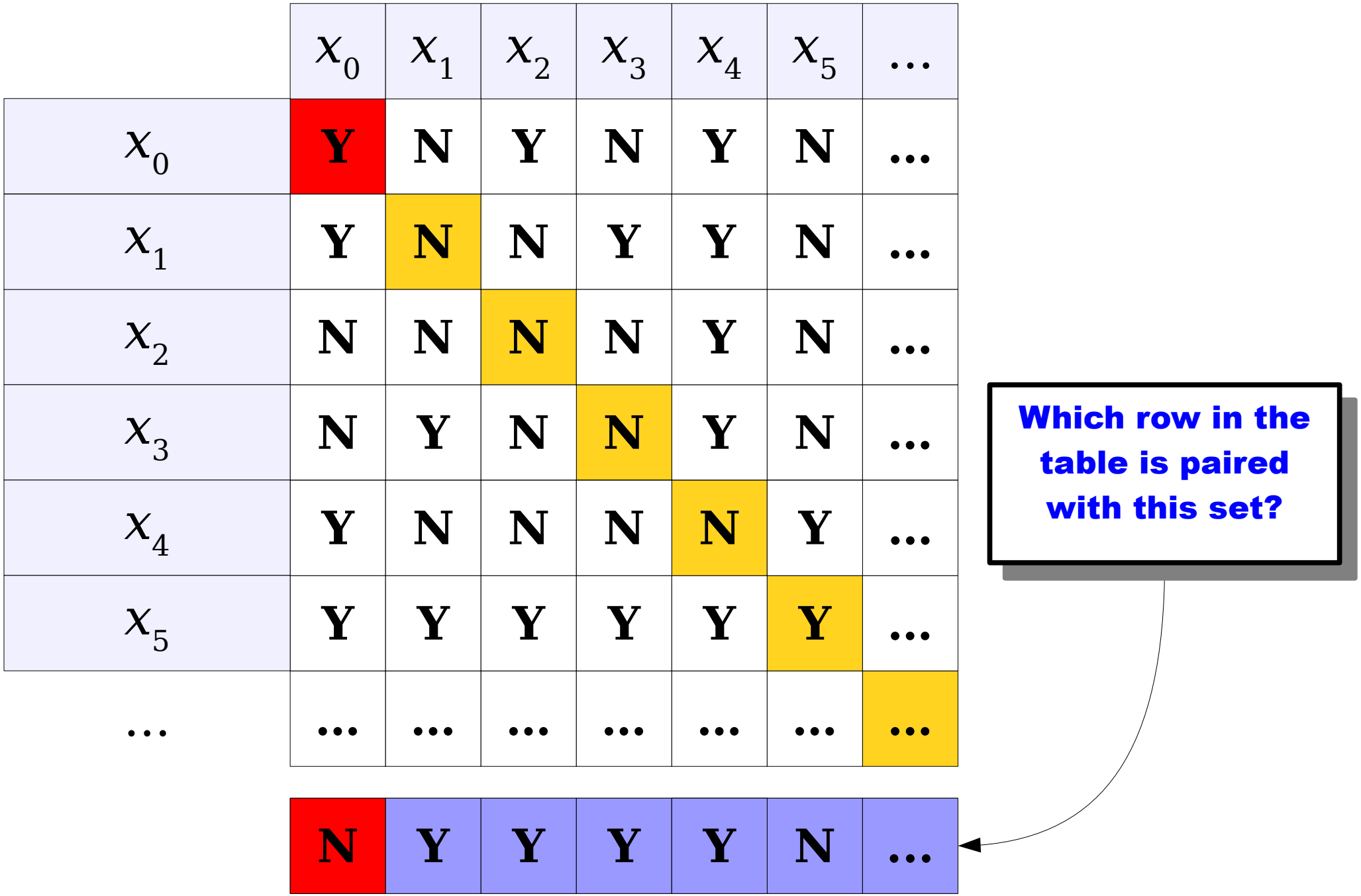
Y	**N**	**N**	**N**	**N**	**Y**	…

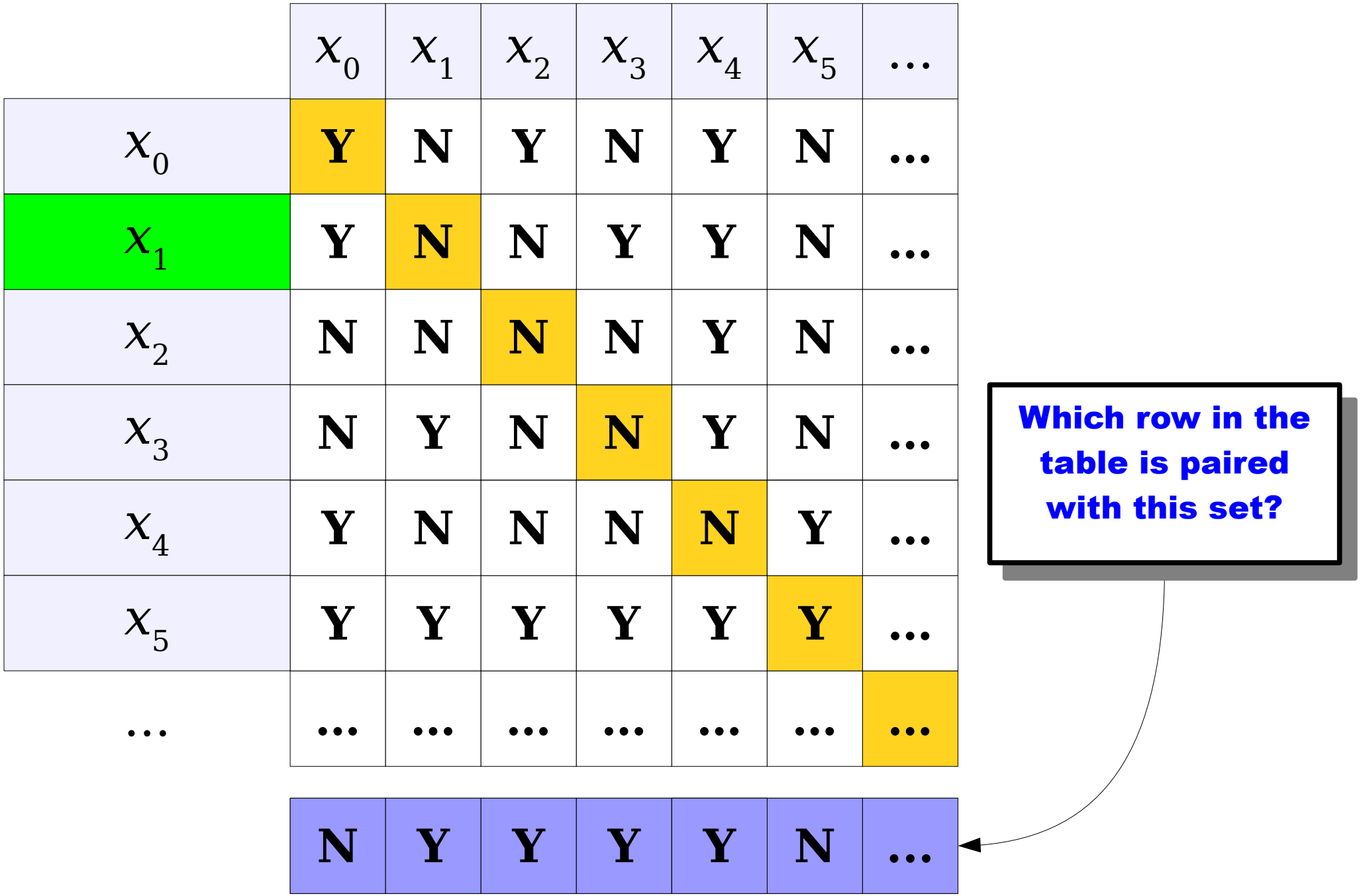

x_0 x_1 x_2 x_3 x_4 x_5 …
x_0 Y N Y N Y N …
x_1 Y N N Y Y N …
x_2 N N N N Y N …
x_3 N Y N N Y N …
x_4 Y N N N N Y …
x_5 Y Y Y Y Y Y …
… … … … … … … …
N Y Y Y Y N …
Flip all Y's to N's and vice-versa to get a new set

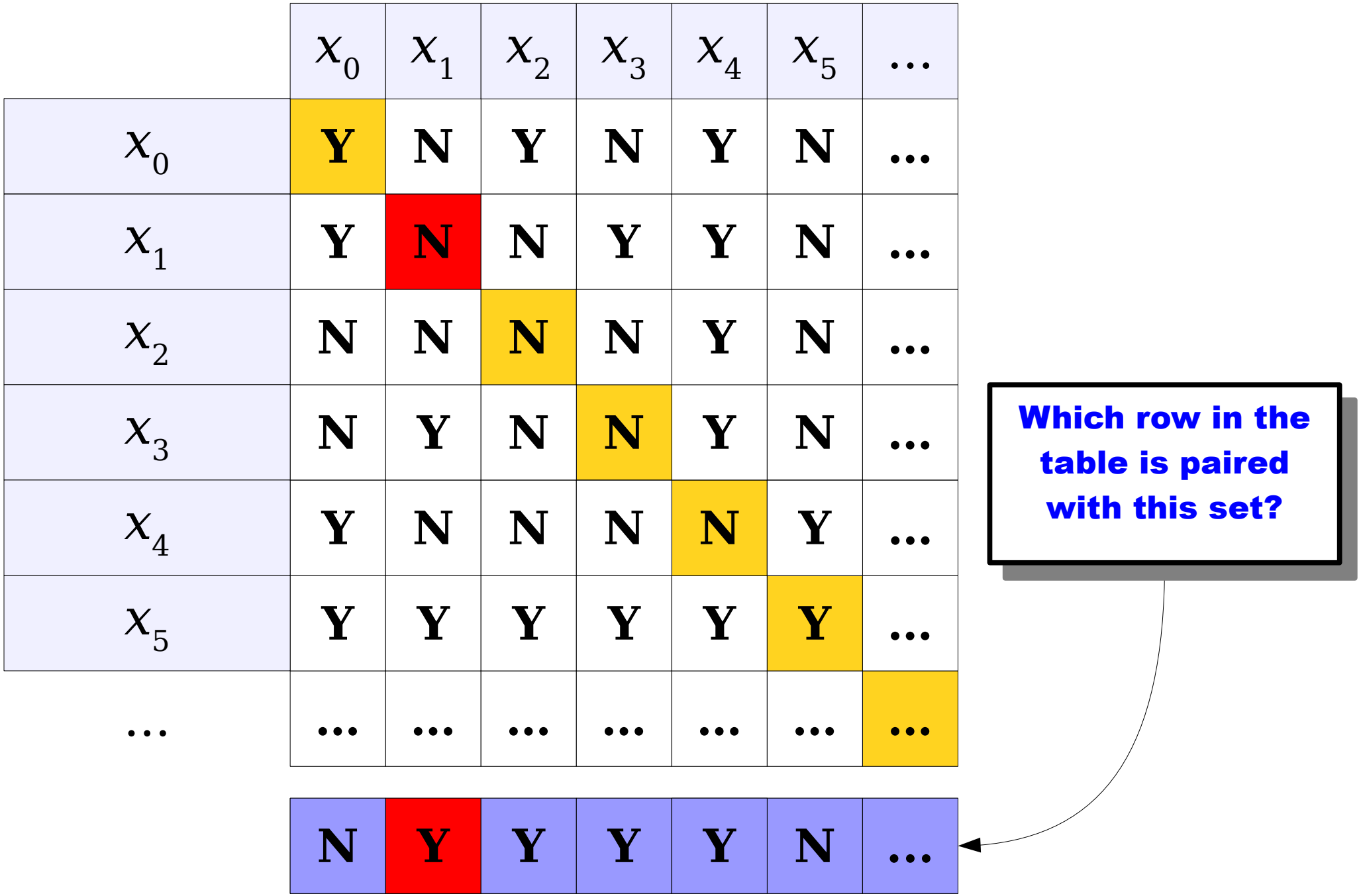

x_0 x_1 x_2 x_3 x_4 x_5 …
x_0 Y N Y N Y N …
x_1 Y N N Y Y N …
x_2 N N N N Y N …
x_3 N Y N N Y N …
x_4 Y N N N N Y …
x_5 Y Y Y Y Y Y …
… … … … … … … …
x_1, x_2, x_3, x_4, …
Flip all Y's to N's and vice-versa to get a new set

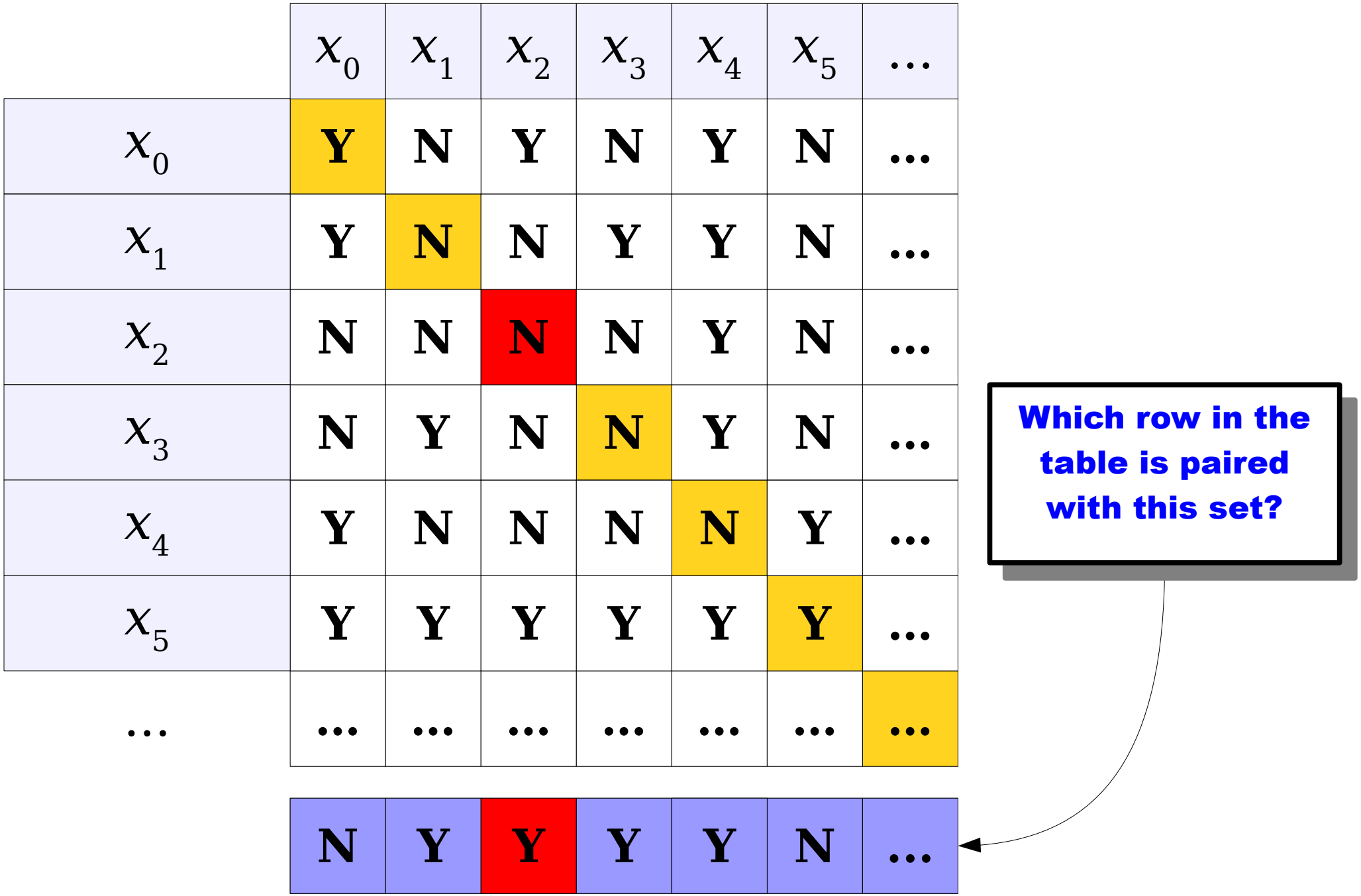

x_0 x_1 x_2 x_3 x_4 x_5 …
x_0 Y N Y N Y N …
x_1 Y N N Y Y N …
x_2 N N N N Y N …
x_3 N Y N N Y N …
x_4 Y N N N N Y …
x_5 Y Y Y Y Y Y …
… … … … … … … …
N Y Y Y Y N …
Which row in the table is paired with this set?

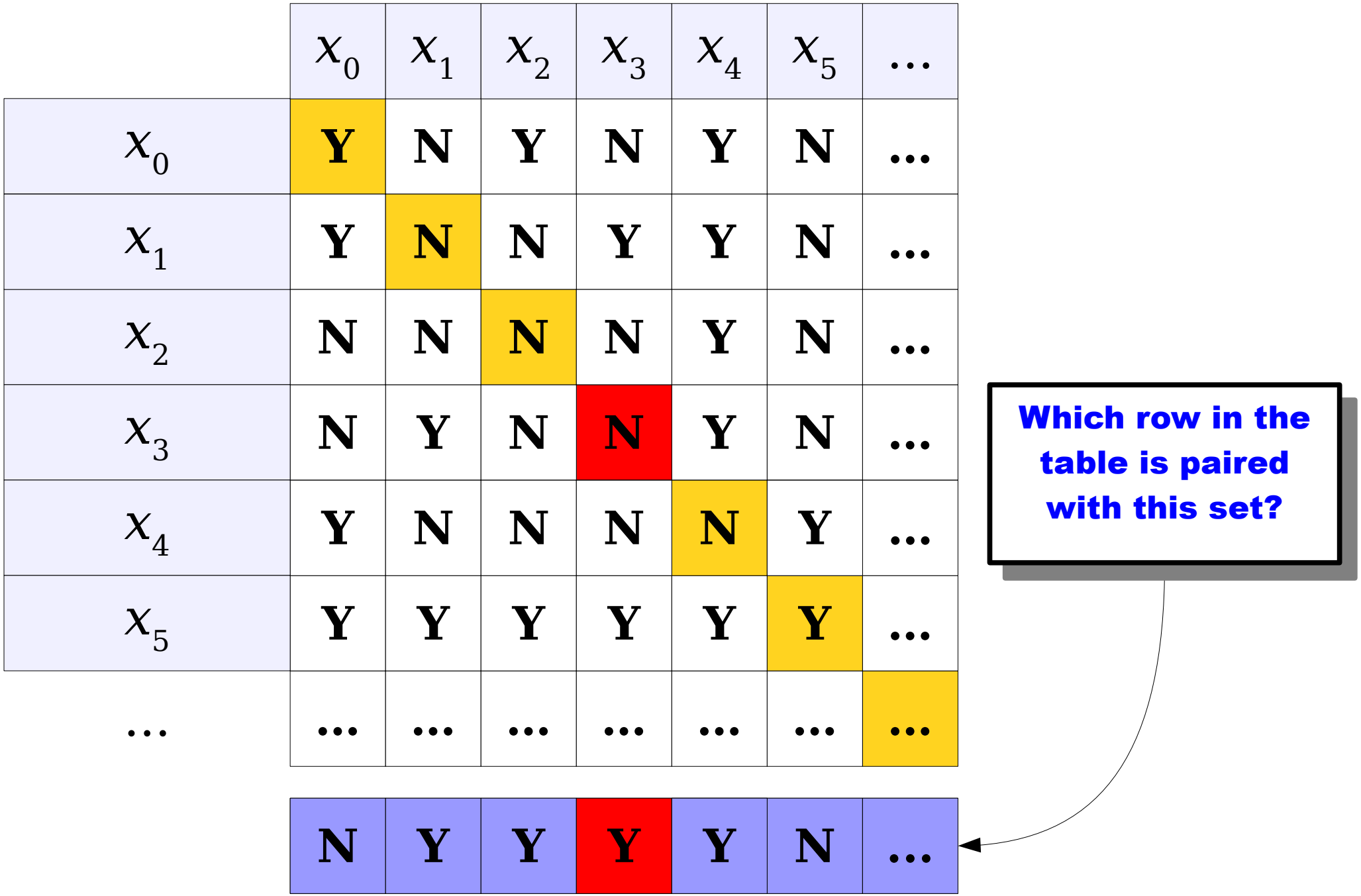

x_0 x_1 x_2 x_3 x_4 x_5 …
x_0 Y N Y N Y N …
x_1 Y N N Y Y N …
x_2 N N N N Y N …
x_3 N Y N N Y N …
x_4 Y N N N N Y …
x_5 Y Y Y Y Y Y …
… … … … … … … …
N Y Y Y Y N …
Which row in the table is paired with this set?

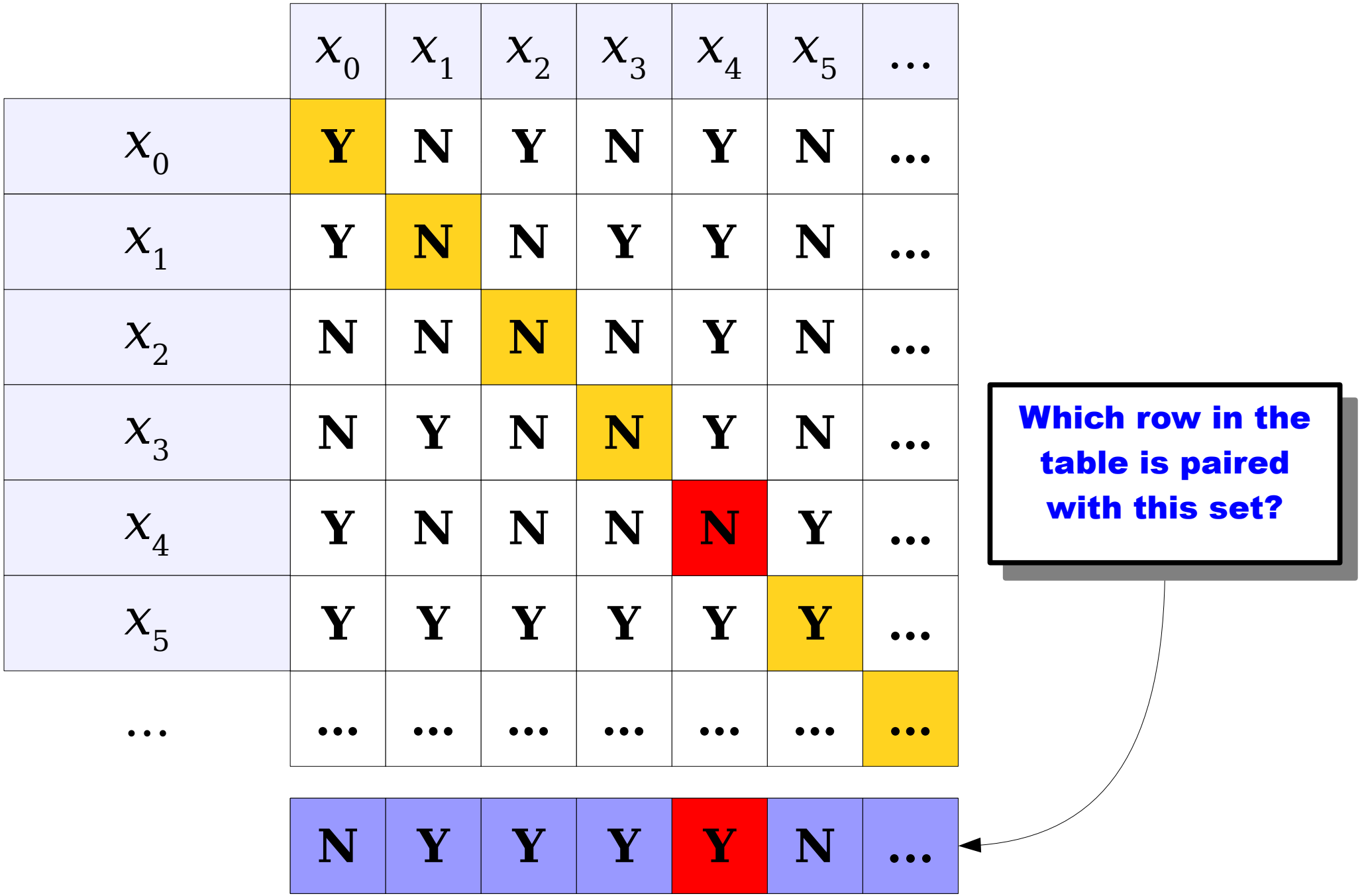

x_0 x_1 x_2 x_3 x_4 x_5 …
x_0 Y N Y N Y N …
x_1 Y N N Y Y N …
x_2 N N N N Y N …
x_3 N Y N N Y N …
x_4 Y N N N N Y …
x_5 Y Y Y Y Y Y …
… … … … … … … …
N Y Y Y Y N …
Which row in the table is paired with this set?

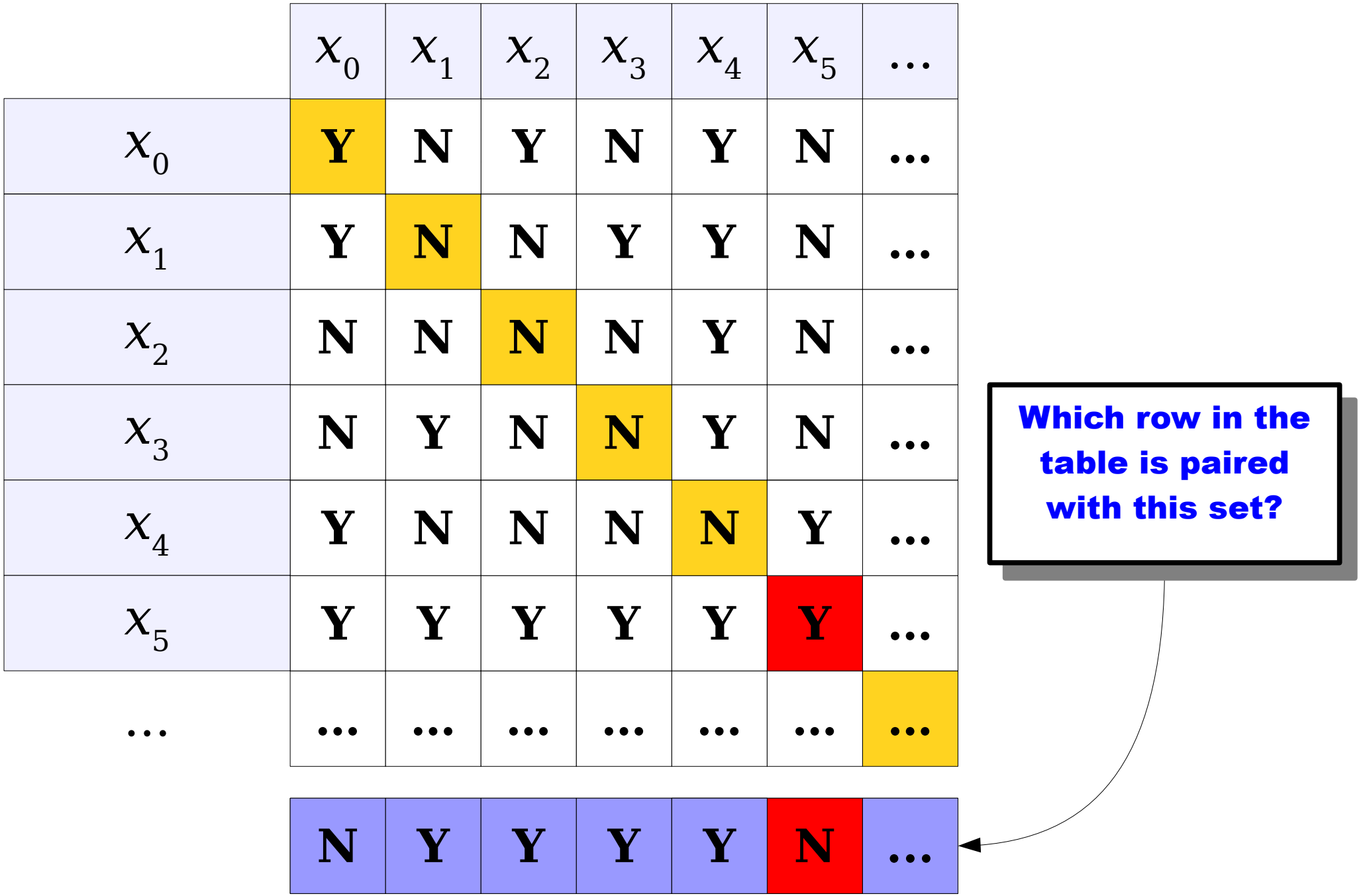

x_0 x_1 x_2 x_3 x_4 x_5 …
x_0 Y N Y N Y N …
x_1 Y N N Y Y N …
x_2 N N N N Y N …
x_3 N Y N N Y N …
x_4 Y N N N N Y …
x_5 Y Y Y Y Y Y …
… … … … … … … …
N Y Y Y Y N …
Which row in the table is paired with this set?

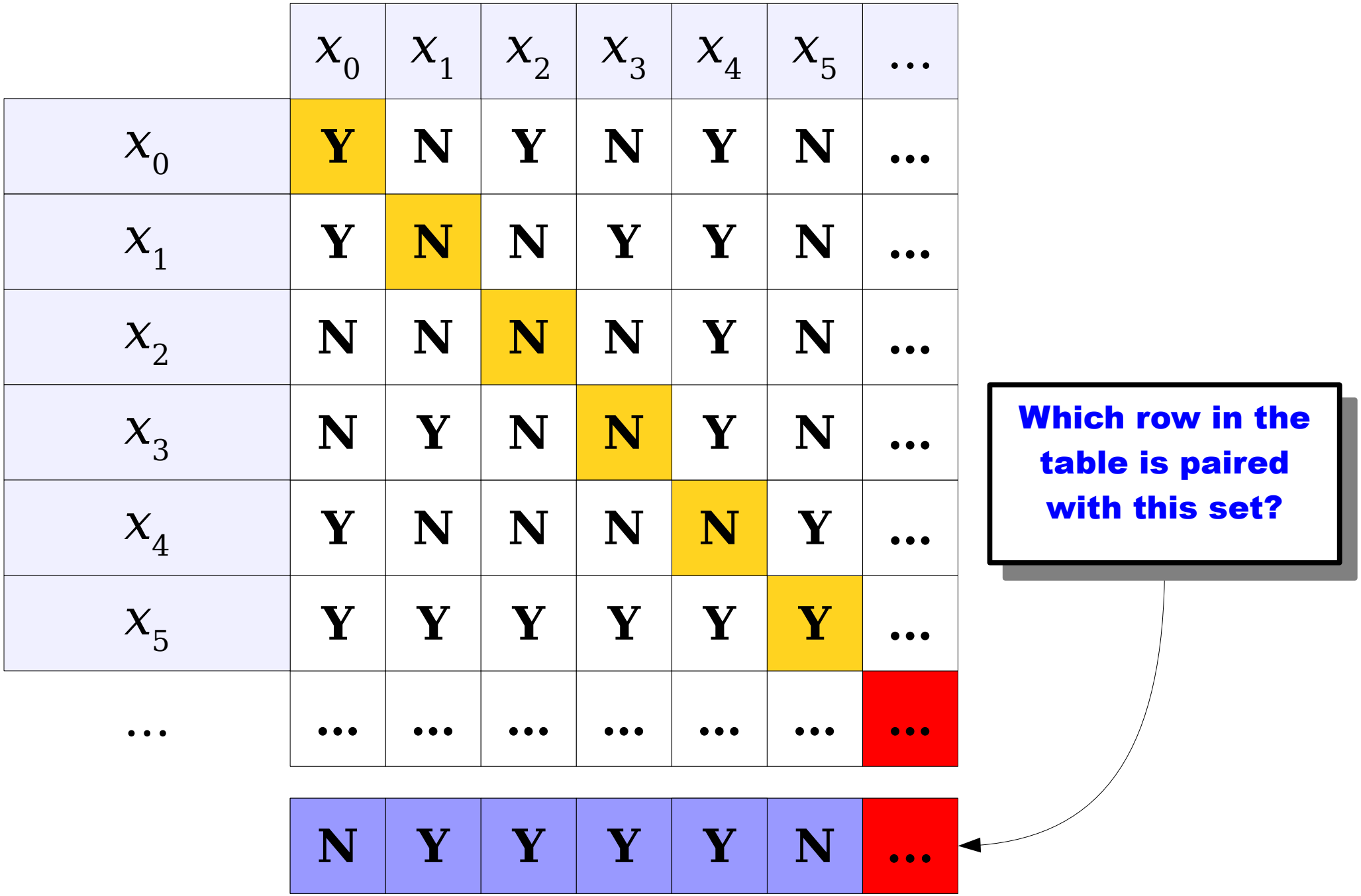

x_0 x_1 x_2 x_3 x_4 x_5 …
x_0 Y N Y N Y N …
x_1 Y N N Y Y N …
x_2 N N N N Y N …
x_3 N Y N N Y N …
x_4 Y N N N N Y …
x_5 Y Y Y Y Y Y …
… … … … … … … …
N Y Y Y Y N …
Which row in the table is paired with this set?


x_0 x_1 x_2 x_3 x_4 x_5 …
x_0 Y N Y N Y N …
x_1 Y N N Y Y N …
x_2 N N N N Y N …
x_3 N Y N N Y N …
x_4 Y N N N N Y …
x_5 Y Y Y Y Y Y …
… … … … … … … …
N Y Y Y Y N …
Which row in the table is paired with this set?


x_0 x_1 x_2 x_3 x_4 x_5 …
x_0 Y N Y N Y N …
x_1 Y N N Y Y N …
x_2 N N N N Y N …
x_3 N Y N N Y N …
x_4 Y N N N N Y …
x_5 Y Y Y Y Y Y …
… … … … … … … …
N Y Y Y Y N …
Which row in the table is paired with this set?


x_0 x_1 x_2 x_3 x_4 x_5 …
x_0 Y N Y N Y N …
x_1 Y N N Y Y N …
x_2 N N N N Y N …
x_3 N Y N N Y N …
x_4 Y N N N N Y …
x_5 Y Y Y Y Y Y …
… … … … … … … …
N Y Y Y Y N …
Which row in the table is paired with this set?


x_0 x_1 x_2 x_3 x_4 x_5 …
x_0 Y N Y N Y N …
x_1 Y N N Y Y N …
x_2 N N N N Y N …
x_3 N Y N N Y N …
x_4 Y N N N N Y …
x_5 Y Y Y Y Y Y …
… … … … … … … …
N Y Y Y Y N …
Which row in the table is paired with this set?


x_0 x_1 x_2 x_3 x_4 x_5 …
x_0 Y N Y N Y N …
x_1 Y N N Y Y N …
x_2 N N N N Y N …
x_3 N Y N N Y N …
x_4 Y N N N N Y …
x_5 Y Y Y Y Y Y …
… … … … … … … …
N Y Y Y Y N …
Which row in the table is paired with this set?

	x_0	x_1	x_2	x_3	x_4	x_5	…
x_0	**Y**	**N**	**Y**	**N**	**Y**	**N**	…
x_1	**Y**	**N**	**N**	**Y**	**Y**	**N**	…
x_2	**N**	**N**	**N**	**N**	**Y**	**N**	…
x_3	**N**	**Y**	**N**	**N**	**Y**	**N**	…
x_4	**Y**	**N**	**N**	**N**	**N**	**Y**	…
x_5	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**	…
…	…	…	…	…	…	…	…

	x_0	x_1	x_2	x_3	x_4	x_5	…
x_0	**N**	**N**	**N**	**N**	**N**	**Y**	…
x_1	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**	…
x_2	**N**	**Y**	**N**	**Y**	**Y**	**N**	…
x_3	**Y**	**N**	**Y**	**N**	**Y**	**N**	…
x_4	**N**	**N**	**Y**	**Y**	**N**	**Y**	…
x_5	**Y**	**N**	**N**	**N**	**N**	**Y**	…
…	…	…	…	…	…	…	…

	x_0	x_1	x_2	x_3	x_4	x_5	…
x_0	**N**	**N**	**N**	**N**	**N**	**Y**	…
x_1	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**	…
x_2	**N**	**Y**	**N**	**Y**	**Y**	**N**	…
x_3	**Y**	**N**	**Y**	**N**	**Y**	**N**	…
x_4	**N**	**N**	**Y**	**Y**	**N**	**Y**	…
x_5	**Y**	**N**	**N**	**N**	**N**	**Y**	…
…	…	…	…	…	…	…	…

	x_0	x_1	x_2	x_3	x_4	x_5	…
x_0	**N**	**N**	**N**	**N**	**N**	**Y**	…
x_1	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**	…
x_2	**N**	**Y**	**N**	**Y**	**Y**	**N**	…
x_3	**Y**	**N**	**Y**	**N**	**Y**	**N**	…
x_4	**N**	**N**	**Y**	**Y**	**N**	**Y**	…
x_5	**Y**	**N**	**N**	**N**	**N**	**Y**	…
…	…	…	…	…	…	…	…

N	**Y**	**N**	**N**	**N**	**Y**	…

	x_0	x_1	x_2	x_3	x_4	x_5	…
x_0	**N**	**N**	**N**	**N**	**N**	**Y**	…
x_1	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**	…
x_2	**N**	**Y**	**N**	**Y**	**Y**	**N**	…
x_3	**Y**	**N**	**Y**	**N**	**Y**	**N**	…
x_4	**N**	**N**	**Y**	**Y**	**N**	**Y**	…
x_5	**Y**	**N**	**N**	**N**	**N**	**Y**	…
…	…	…	…	…	…	…	…

Y	**N**	**Y**	**Y**	**Y**	**N**	…

	x_0	x_1	x_2	x_3	x_4	x_5	…
x_0	**N**	**N**	**N**	**N**	**N**	**Y**	…
x_1	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**	…
x_2	**N**	**Y**	**N**	**Y**	**Y**	**N**	…
x_3	**Y**	**N**	**Y**	**N**	**Y**	**N**	…
x_4	**N**	**N**	**Y**	**Y**	**N**	**Y**	…
x_5	**Y**	**N**	**N**	**N**	**N**	**Y**	…
…	…	…	…	…	…	…	…

Y	**N**	**Y**	**Y**	**Y**	**N**	…

	x_0	x_1	x_2	x_3	x_4	x_5	…
x_0	**N**	**N**	**N**	**N**	**N**	**Y**	…
x_1	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**	…
x_2	**N**	**Y**	**N**	**Y**	**Y**	**N**	…
x_3	**Y**	**N**	**Y**	**N**	**Y**	**N**	…
x_4	**N**	**N**	**Y**	**Y**	**N**	**Y**	…
x_5	**Y**	**N**	**N**	**N**	**N**	**Y**	…
…	…	…	…	…	…	…	…

Y	**N**	**Y**	**Y**	**Y**	**N**	…

	x_0	x_1	x_2	x_3	x_4	x_5	…
x_0	**N**	**N**	**N**	**N**	**N**	**Y**	…
x_1	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**	…
x_2	**N**	**Y**	**N**	**Y**	**Y**	**N**	…
x_3	**Y**	**N**	**Y**	**N**	**Y**	**N**	…
x_4	**N**	**N**	**Y**	**Y**	**N**	**Y**	…
x_5	**Y**	**N**	**N**	**N**	**N**	**Y**	…
…	…	…	…	…	…	…	…

Y	**N**	**Y**	**Y**	**Y**	**N**	…

	x_0	x_1	x_2	x_3	x_4	x_5	…
x_0	**N**	**N**	**N**	**N**	**N**	**Y**	…
x_1	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**	…
x_2	**N**	**Y**	**N**	**Y**	**Y**	**N**	…
x_3	**Y**	**N**	**Y**	**N**	**Y**	**N**	…
x_4	**N**	**N**	**Y**	**Y**	**N**	**Y**	…
x_5	**Y**	**N**	**N**	**N**	**N**	**Y**	…
…	…	…	…	…	…	…	…

Y	**N**	**Y**	**Y**	**Y**	**N**	…

	x_0	x_1	x_2	x_3	x_4	x_5	…
x_0	**N**	**N**	**N**	**N**	**N**	**Y**	…
x_1	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**	…
x_2	**N**	**Y**	**N**	**Y**	**Y**	**N**	…
x_3	**Y**	**N**	**Y**	**N**	**Y**	**N**	…
x_4	**N**	**N**	**Y**	**Y**	**N**	**Y**	…
x_5	**Y**	**N**	**N**	**N**	**N**	**Y**	…
…	…	…	…	…	…	…	…

Y	**N**	**Y**	**Y**	**Y**	**N**	…

	x_0	x_1	x_2	x_3	x_4	x_5	…
x_0	**N**	**N**	**N**	**N**	**N**	**Y**	…
x_1	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**	…
x_2	**N**	**Y**	**N**	**Y**	**Y**	**N**	…
x_3	**Y**	**N**	**Y**	**N**	**Y**	**N**	…
x_4	**N**	**N**	**Y**	**Y**	**N**	**Y**	…
x_5	**Y**	**N**	**N**	**N**	**N**	**Y**	…
…	…	…	…	…	…	…	…

Y	**N**	**Y**	**Y**	**Y**	**N**	…

	x_0	x_1	x_2	x_3	x_4	x_5	…
x_0	**N**	**N**	**N**	**N**	**N**	**Y**	…
x_1	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**	…
x_2	**N**	**Y**	**N**	**Y**	**Y**	**N**	…
x_3	**Y**	**N**	**Y**	**N**	**Y**	**N**	…
x_4	**N**	**N**	**Y**	**Y**	**N**	**Y**	…
x_5	**Y**	**N**	**N**	**N**	**N**	**Y**	…
…	…	…	…	…	…	…	…

Y	**N**	**Y**	**Y**	**Y**	**N**	…

The Diagonalization Proof

- No matter how we pair up elements of S and subsets of S, the complemented diagonal won't appear in the table.
  - In row n, the nth element must be wrong.
- No matter how we pair up elements of S and subsets of S, there is *always* at least one subset left over.
- This result is ***Cantor's theorem***: Every set is strictly smaller than its power set:

If S is a set, then $|S| < |\wp(S)|$.

Infinite Cardinalities

- 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!***

What does this have to do with computation?

“The set of all computer programs”

“The set of all problems to solve”

Where We're Going

- A ***string*** is a sequence of characters.
- We're going to prove the following results:
  - There are ***at most*** as many programs as there are strings.
  - There are ***at least*** as many problems as there are sets of strings.
- This leads to some *incredible* results – we'll see why in a minute!

Where We're Going

A ***string*** is a sequence of characters.

We're going to prove the following results:

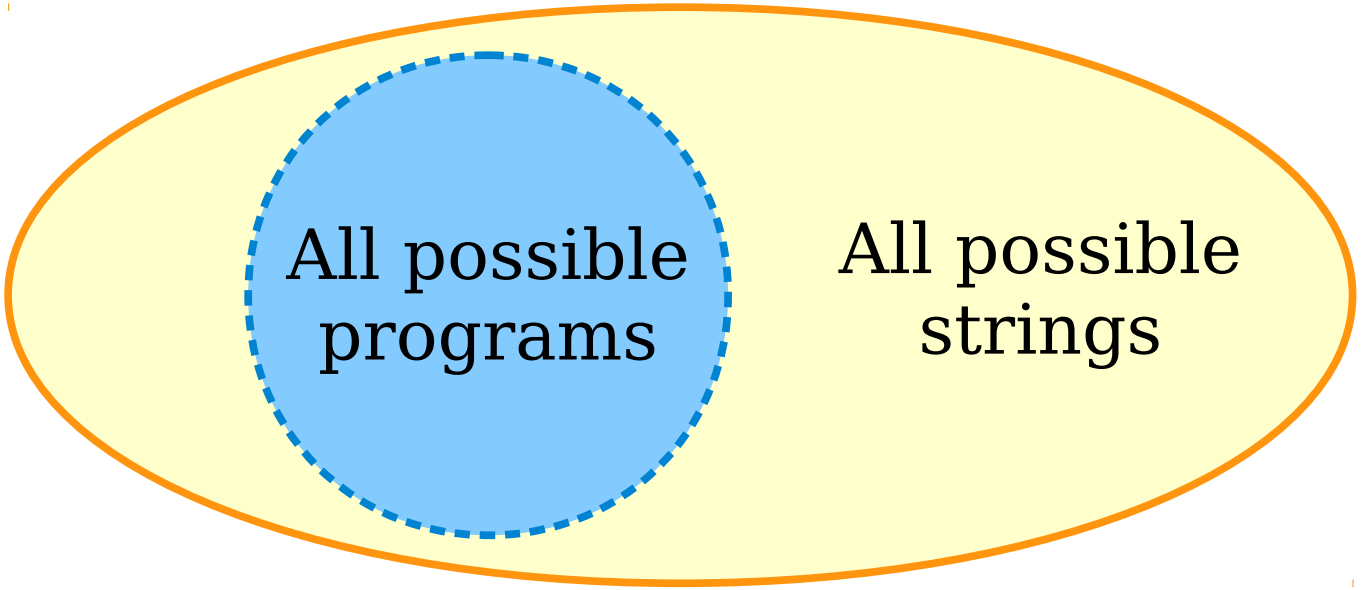
- There are ***at most*** as many programs as there are strings.

  There are ***at least*** as many problems as there are sets of strings.

This leads to some *incredible* results – we'll see why in a minute!

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|

Where We're Going

- A ***string*** is a sequence of characters.
- We're going to prove the following results:
  - There are ***at most*** as many programs as there are strings.
  - There are ***at least*** as many problems as there are sets of strings.
- This leads to some *incredible* results – we'll see why in a minute!

Where We're Going

- A ***string*** is a sequence of characters.
- We're going to prove the following results:
  - There are ***at most*** as many programs as there are strings. ✓
  - There are ***at least*** as many problems as there are sets of strings.
- This leads to some *incredible* results – we'll see why in a minute!

Where We're Going

A ***string*** is a sequence of characters.

We're going to prove the following results:

- There are ***at most*** as many programs as there are strings. ✓
- There are ***at least*** as many problems as there are sets of strings.

This leads to some *incredible* results – we'll see why in a minute!

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

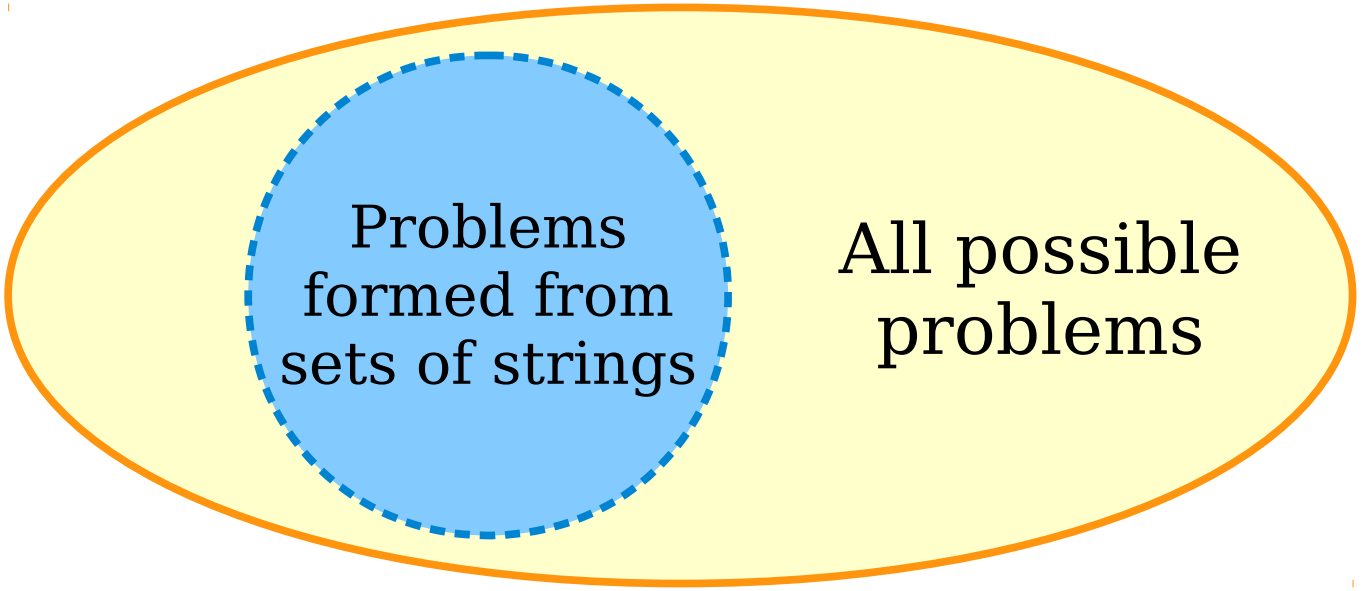
$$S = \{ p \mid p \text{ is a legal Java program} \}$$

- From this set S, we get this problem:

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

Strings and Problems

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



|Sets of Strings| ≤ |Problems|

Where We're Going

- A ***string*** is a sequence of characters.
- We're going to prove the following results:
  - There are ***at most*** as many programs as there are strings. ✓
  - There are ***at least*** as many problems as there are sets of strings.
- This leads to some *incredible* results – we'll see why in a minute!

Where We're Going

- A ***string*** is a sequence of characters.
- We're going to prove the following results:
  - There are ***at most*** as many programs as there are strings. ✓
  - There are ***at least*** as many problems as there are sets of strings. ✓
- This leads to some *incredible* results – we'll see why in a minute!

Where We're Going

A ***string*** is a sequence of characters.

We're going to prove the following results:

- There are ***at most*** as many programs as there are strings. ✓
- There are ***at least*** as many problems as there are sets of strings. ✓

- This leads to some *incredible* results – we'll see why in a minute!

Where We're Going

A ***string*** is a sequence of characters.

We're going to prove the following results:

- There are ***at most*** as many programs as there are strings. ✓
- There are ***at least*** as many problems as there are sets of strings. ✓

- This leads to some *incredible* results – we'll see why ~~in a minute!~~ *right now!*

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.

|Programs| ≤ |Strings| < |Sets of Strings| ≤ |Problems|

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.

|Programs| < |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?