Discrete Algebraic Structures

WiSe 2025/2026

Prof. Dr. Antoine Wiehe
Research Group for Theoretical Computer Science

Overview of the course

Part 5: Algebra
- Cryptography
- Error correction
- Quantum Computing

Part 3: Combinatorics and counting
- Runtime analysis of algorithms
- Basics of probabilities

next 2 weeks

Part 4: basic number theory

Part 2: Graphs and relations
- Representing data and relationships between things
- Important basis for the rest of your studies (in particular "Algorithms and Data Structures")

Part 1: introduction to mathematics
- Sets, functions
- Basics in logic and proofs

- Runtime of algorithms:

```
n = 10
S = { (a,b,c) for a in range(n)
              for b in range(n)
              for c in range(n)
              if a!=b and b!=c }
for (a,b,c) in S:
  # do something
  foo(a,b,c)
```

- Runtime of algorithms:

```
n = 10
S = { (a,b,c) for a in range(n)
              for b in range(n)
              for c in range(n)
              if a!=b and b!=c }
for (a,b,c) in S:
  # do something
  foo(a,b,c)
```

$$S = \{(a, b, c) \in \{0, \ldots, n-1\}^3 \mid a \neq b \text{ and } b \neq c\}$$

- Runtime of algorithms:

```
n = 10
S = { (a,b,c) for a in range(n)
              for b in range(n)
              for c in range(n)
              if a!=b and b!=c }
for (a,b,c) in S:
  # do something
  foo(a,b,c)
```

$$S = \{(a, b, c) \in \{0, \dots, n-1\}^3 \mid a \neq b \text{ and } b \neq c\}$$

- Runtime depends on the size of S
- For which values of n would this code run fast?

- Runtime of algorithms:

```
n = 10
S = { (a,b,c) for a in range(n)
              for b in range(n)
              for c in range(n)
              if a!=b and b!=c }
for (a,b,c) in S:
  # do something
  foo(a,b,c)
```

$$S = \{(a, b, c) \in \{0, \ldots, n-1\}^3 \mid a \neq b \text{ and } b \neq c\}$$

- Runtime depends on the size of S
- For which values of n would this code run fast?

$n = 10 \Rightarrow S$ has size 810

$n = 1000 \Rightarrow S$ has size 998001000

- Runtime of algorithms:

```
n = 10
S = { (a,b,c) for a in range(n)
              for b in range(n)
              for c in range(n)
              if a!=b and b!=c }
for (a,b,c) in S:
  # do something
  foo(a,b,c)
```

$$S = \{(a,b,c) \in \{0,\ldots,n-1\}^3 \mid a \neq b \text{ and } b \neq c\}$$

- Runtime depends on the size of S
- For which values of n would this code run fast?

$$n = 10 \Rightarrow S \text{ has size } 810$$
$$n = 1000 \Rightarrow S \text{ has size } 998001000$$

- Computations with probabilities:

$$\Pr[\text{rolling a 7 with two dice}] = ?$$

- Runtime of algorithms:

```
n = 10
S = { (a,b,c) for a in range(n)
              for b in range(n)
              for c in range(n)
              if a!=b and b!=c }
for (a,b,c) in S:
  # do something
  foo(a,b,c)
```

$$S = \{(a,b,c) \in \{0,\dots,n-1\}^3 \mid a \neq b \text{ and } b \neq c\}$$

- Runtime depends on the size of S
- For which values of n would this code run fast?

$$n = 10 \Rightarrow S \text{ has size } 810$$
$$n = 1000 \Rightarrow S \text{ has size } 998001000$$

- Computations with probabilities:

$$\Pr[\text{rolling a 7 with two dice}] = ? = \frac{\text{number of ways to get a 7}}{\text{total number of ways of rolling two dice}}$$

- Runtime of algorithms:

```
n = 10
S = { (a,b,c) for a in range(n)
              for b in range(n)
              for c in range(n)
              if a!=b and b!=c }
for (a,b,c) in S:
  # do something
  foo(a,b,c)
```

$$S = \{(a,b,c) \in \{0,\ldots,n-1\}^3 \mid a \neq b \text{ and } b \neq c\}$$

- Runtime depends on the size of S
- For which values of n would this code run fast?

$$n = 10 \Rightarrow S \text{ has size } 810$$
$$n = 1000 \Rightarrow S \text{ has size } 998001000$$

- Computations with probabilities:

$$\Pr[\text{rolling a 7 with two dice}] = ? = \frac{\text{number of ways to get a 7}}{\text{total number of ways of rolling two dice}}$$

- Many algorithms use randomness:

$$\Pr[\text{algorithm outputs correct result}] = \frac{\text{number of ways to arrive at correct result}}{\text{total number of ways the algorithm can run}}$$

- Runtime of algorithms:

```
n = 10
S = { (a,b,c) for a in range(n)
              for b in range(n)
              for c in range(n)
              if a!=b and b!=c }
for (a,b,c) in S:
  # do something
  foo(a,b,c)
```

$$S = \{(a,b,c) \in \{0,\ldots,n-1\}^3 \mid a \neq b \text{ and } b \neq c\}$$

- Runtime depends on the size of S
- For which values of n would this code run fast?

$$n = 10 \Rightarrow S \text{ has size } 810$$
$$n = 1000 \Rightarrow S \text{ has size } 998001000$$

- Computations with probabilities:

$$\Pr[\text{rolling a 7 with two dice}] = ? = \frac{\text{number of ways to get a 7}}{\text{total number of ways of rolling two dice}}$$

- Many algorithms use randomness:

$$\Pr[\text{algorithm outputs correct result}] = \frac{\text{number of ways to arrive at correct result}}{\text{total number of ways the algorithm can run}}$$

$$\text{average runtime of algorithm} = \sum_{t \geq 1} t \cdot \Pr[\text{algorithm finishes after } t \text{ seconds}]$$

- Runtime of algorithms:

```
n = 10
S = { (a,b,c) for a in range(n)
              for b in range(n)
              for c in range(n)
              if a!=b and b!=c }
for (a,b,c) in S:
  # do something
  foo(a,b,c)
```

$$S = \{(a, b, c) \in \{0, \ldots, n-1\}^3 \mid a \neq b \text{ and } b \neq c\}$$

- Runtime depends on the size of S
- For which values of n would this code run fast?

$n = 10 \Rightarrow S$ has size 810

$n = 1000 \Rightarrow S$ has size 998001000

- Computations with probabilities:

$$\Pr[\text{rolling a 7 with two dice}] = ? = \frac{\text{number of ways to get a 7}}{\text{total number of ways of rolling two dice}}$$

- Many algorithms use randomness:

$$\Pr[\text{algorithm outputs correct result}] = \frac{\text{number of ways to arrive at correct result}}{\text{total number of ways the algorithm can run}}$$

$$\text{average runtime of algorithm} = \sum_{t \geq 1} t \cdot \Pr[\text{algorithm finishes after } t \text{ seconds}]$$

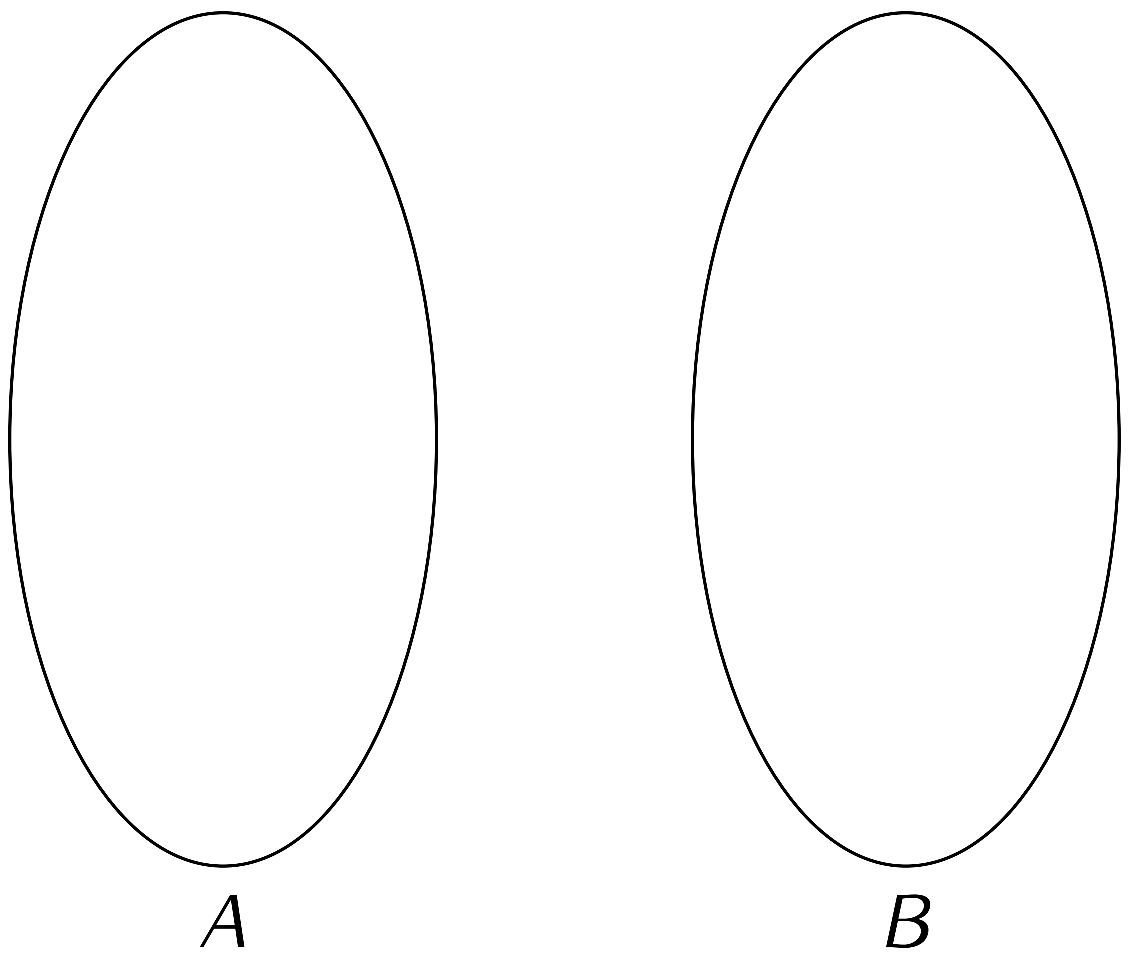
important for your Algorithms course in 2nd year

By comparison: combinatorial proofs

By computations

By comparison: combinatorial proofs

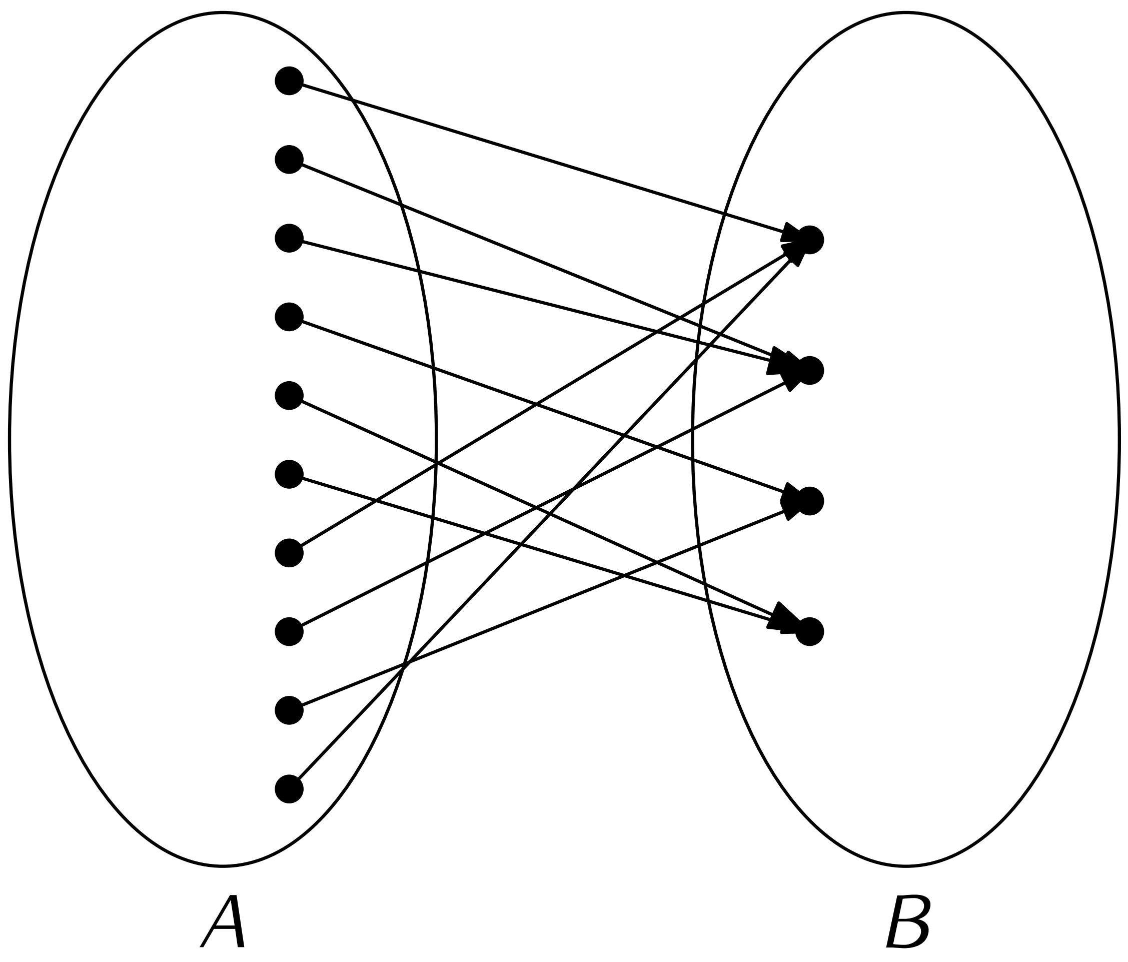
- Size of A already known
- Want to know the size of B



By computations

By comparison: combinatorial proofs

- Size of A already known
- Want to know the size of B

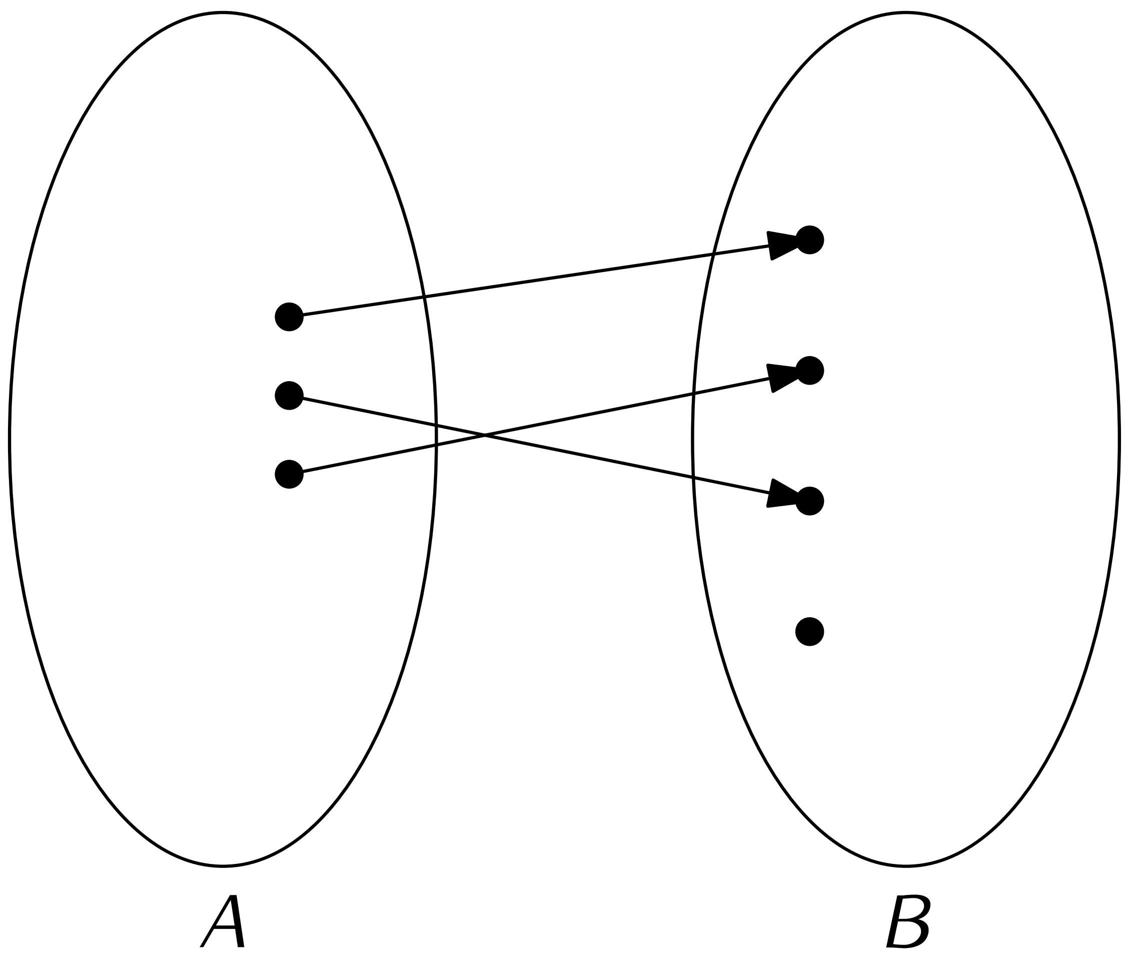


surjective function $\Rightarrow$ B is not bigger

By computations

By comparison: combinatorial proofs

- Size of A already known
- Want to know the size of B

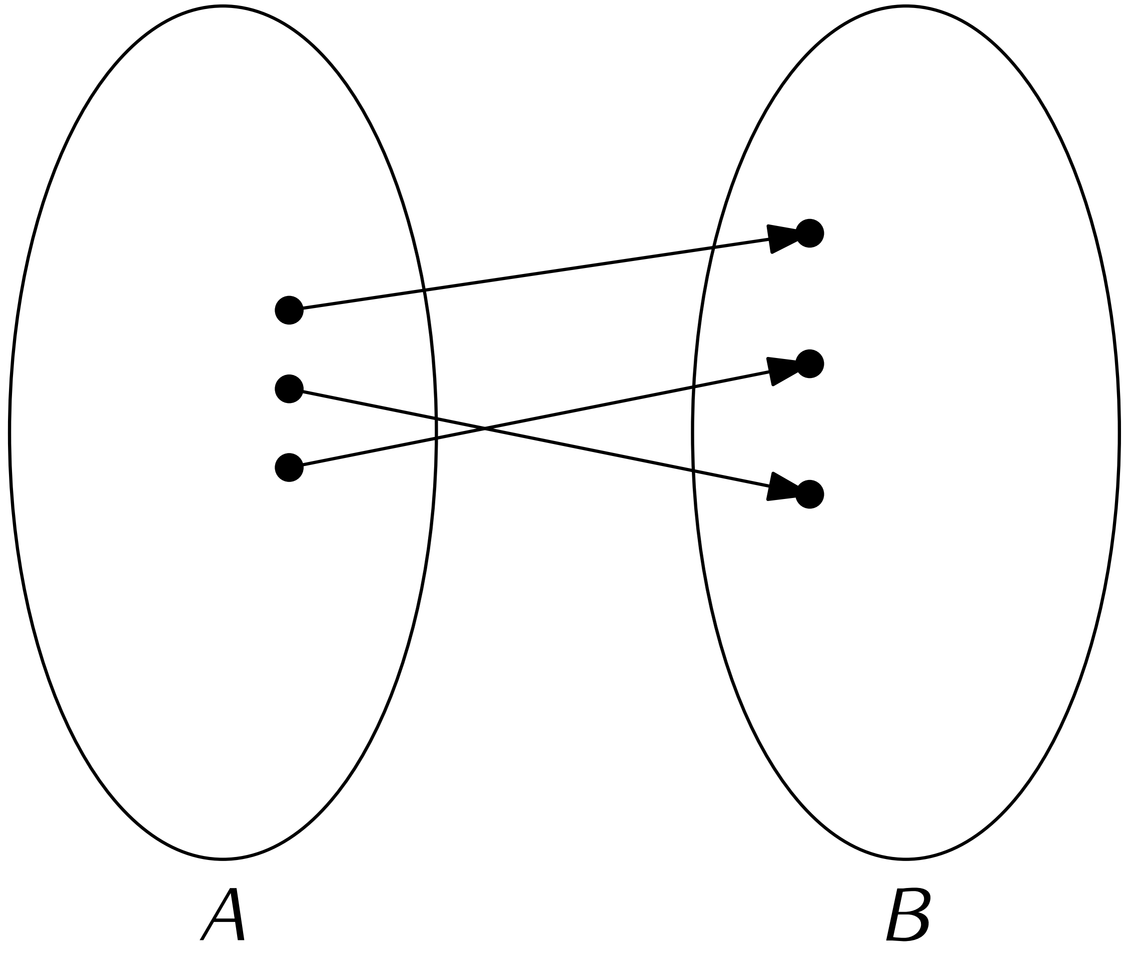


injective function $\Rightarrow$ B is not smaller

By computations

By comparison: combinatorial proofs

- Size of A already known
- Want to know the size of B

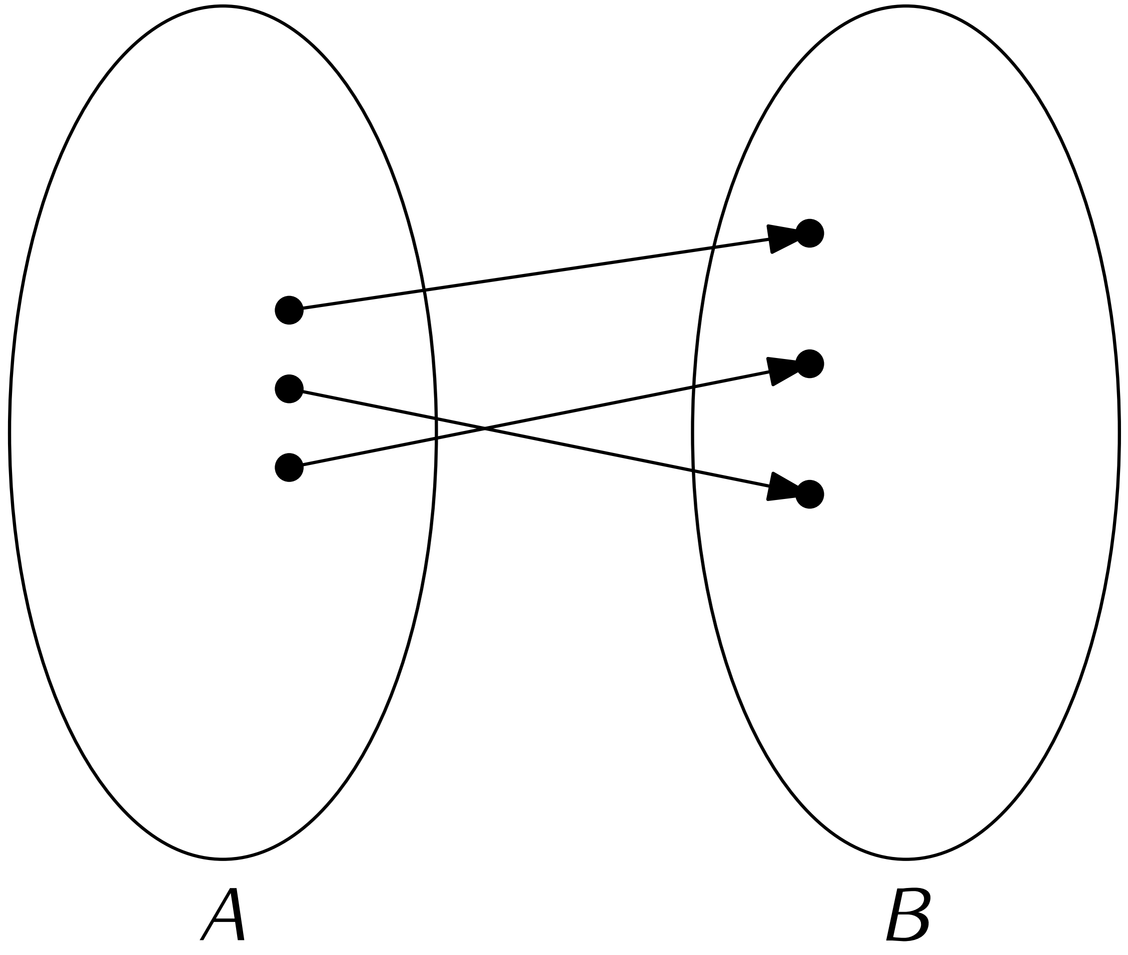


bijective function $\Rightarrow$ B has the same size

By computations

By comparison: combinatorial proofs

- Size of A already known
- Want to know the size of B



bijective function $\Rightarrow$ B has the same size

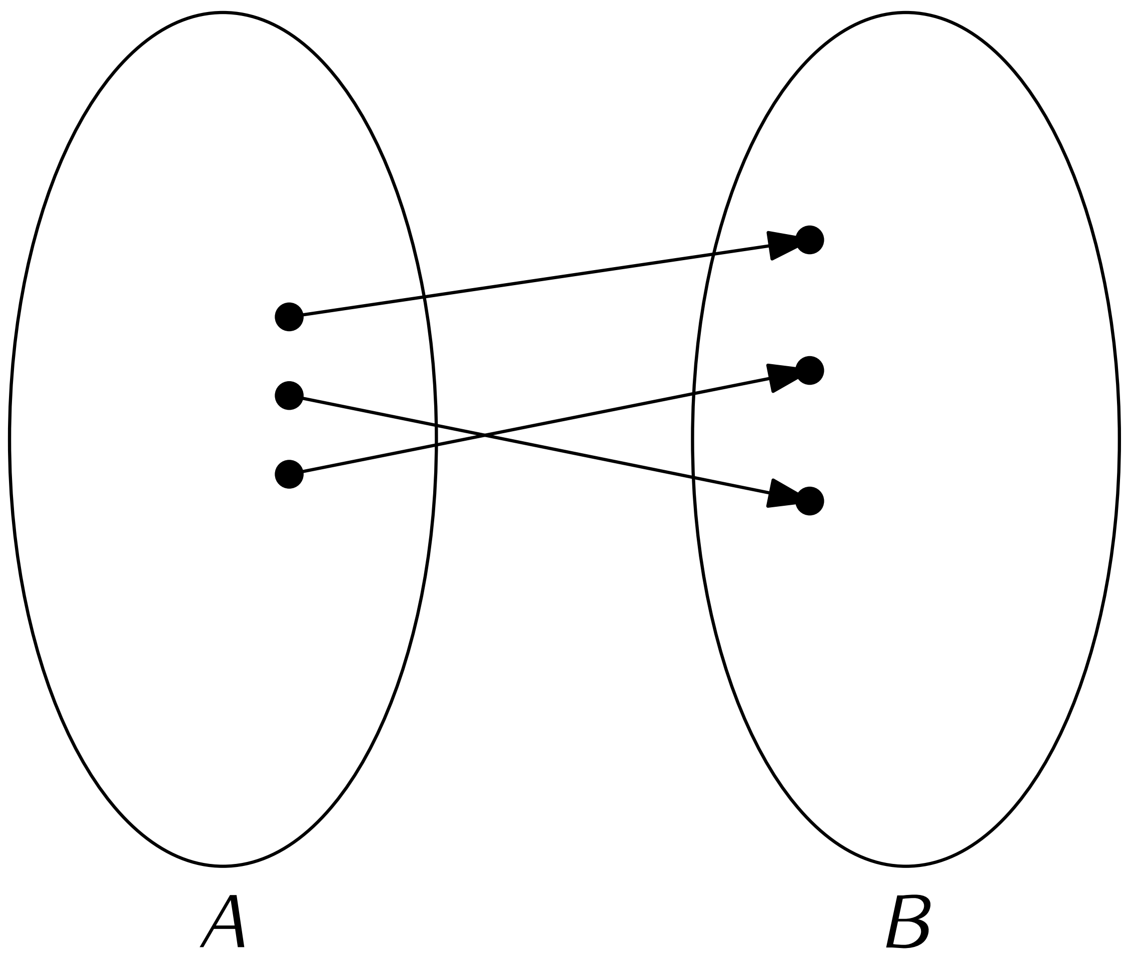
By computations

Sets are usually built by combining certain basic operations:

- unions, products, differences
- drawing things from a set of elements of known size

By comparison: combinatorial proofs

- Size of A already known
- Want to know the size of B



bijective function $\Rightarrow$ B has the same size

By computations

Sets are usually built by combining certain basic operations:

- unions, products, differences
- drawing things from a set of elements of known size

	Order matters	Order does not matter
Replacement	n^k	$\frac{n!}{k!(n-k)!}$
No replacement	$n^{\underline{k}}$	$\binom{n+k-1}{n-1}$

Cardinalities

Notation. We write $|A|$ for the size of the set A.

Notation. We write $|A|$ for the size of the set A.

- If A is finite, easy

Notation. We write $|A|$ for the size of the set A.

- If A is finite, easy
- If A infinite, can we just say $|A| = \infty$?

$$\{2, 4, 6, \ldots\} \quad \subseteq \quad \mathbb{N} \quad \subseteq \quad \mathbb{Z} \quad \subseteq \quad \mathbb{Q} \quad \subseteq \quad \mathbb{R}$$

Notation. We write $|A|$ for the size of the set A.

- If A is finite, easy
- If A infinite, can we just say $|A| = \infty$?

$$\{2, 4, 6, \dots\} \quad \subseteq \quad \mathbb{N} \quad \subseteq \quad \mathbb{Z} \quad \subseteq \quad \mathbb{Q} \quad \subseteq \quad \mathbb{R}$$

Definition. A set A is countable if there exists an injective function $A \to \mathbb{N}$, and it is uncountable otherwise.

Notation. We write $|A|$ for the size of the set A.

- If A is finite, easy
- If A infinite, can we just say $|A| = \infty$?

$$\{2, 4, 6, \dots\} \quad \subseteq \quad \mathbb{N} \quad \subseteq \quad \mathbb{Z} \quad \subseteq \quad \mathbb{Q} \quad \subseteq \quad \mathbb{R}$$

Definition. A set A is countable if there exists an injective function $A \to \mathbb{N}$, and it is uncountable otherwise.

exists injective function $A \to \mathbb{N} \iff$ exists surjective function $f : \mathbb{N} \to A$

Notation. We write $|A|$ for the size of the set A.

- If A is finite, easy
- If A infinite, can we just say $|A| = \infty$?

$$\{2, 4, 6, \dots\} \quad \subseteq \quad \mathbb{N} \quad \subseteq \quad \mathbb{Z} \quad \subseteq \quad \mathbb{Q} \quad \subseteq \quad \mathbb{R}$$

Definition. A set A is countable if there exists an injective function $A \to \mathbb{N}$, and it is uncountable otherwise.

$$\text{exists injective function } A \to \mathbb{N} \overset{*}{\iff} \text{exists surjective function } f\colon \mathbb{N} \to A$$

(*) Only if A is not empty

Notation. We write $|A|$ for the size of the set A.

- If A is finite, easy
- If A infinite, can we just say $|A| = \infty$?

$$\{2, 4, 6, \dots\} \quad \subseteq \quad \mathbb{N} \quad \subseteq \quad \mathbb{Z} \quad \subseteq \quad \mathbb{Q} \quad \subseteq \quad \mathbb{R}$$

Definition. A set A is countable if there exists an injective function $A \to \mathbb{N}$, and it is uncountable otherwise.

$$\text{exists injective function } A \to \mathbb{N} \quad \overset{*}{\Longleftrightarrow} \quad \text{exists surjective function } f : \mathbb{N} \to A$$

$$\{f(1), f(2), f(3), \dots\}$$

(*) Only if A is not empty

Notation. We write $|A|$ for the size of the set A.

- If A is finite, easy
- If A infinite, can we just say $|A| = \infty$?

$$\{2, 4, 6, \dots\} \quad \subseteq \quad \mathbb{N} \quad \subseteq \quad \mathbb{Z} \quad \subseteq \quad \mathbb{Q} \quad \subseteq \quad \mathbb{R}$$

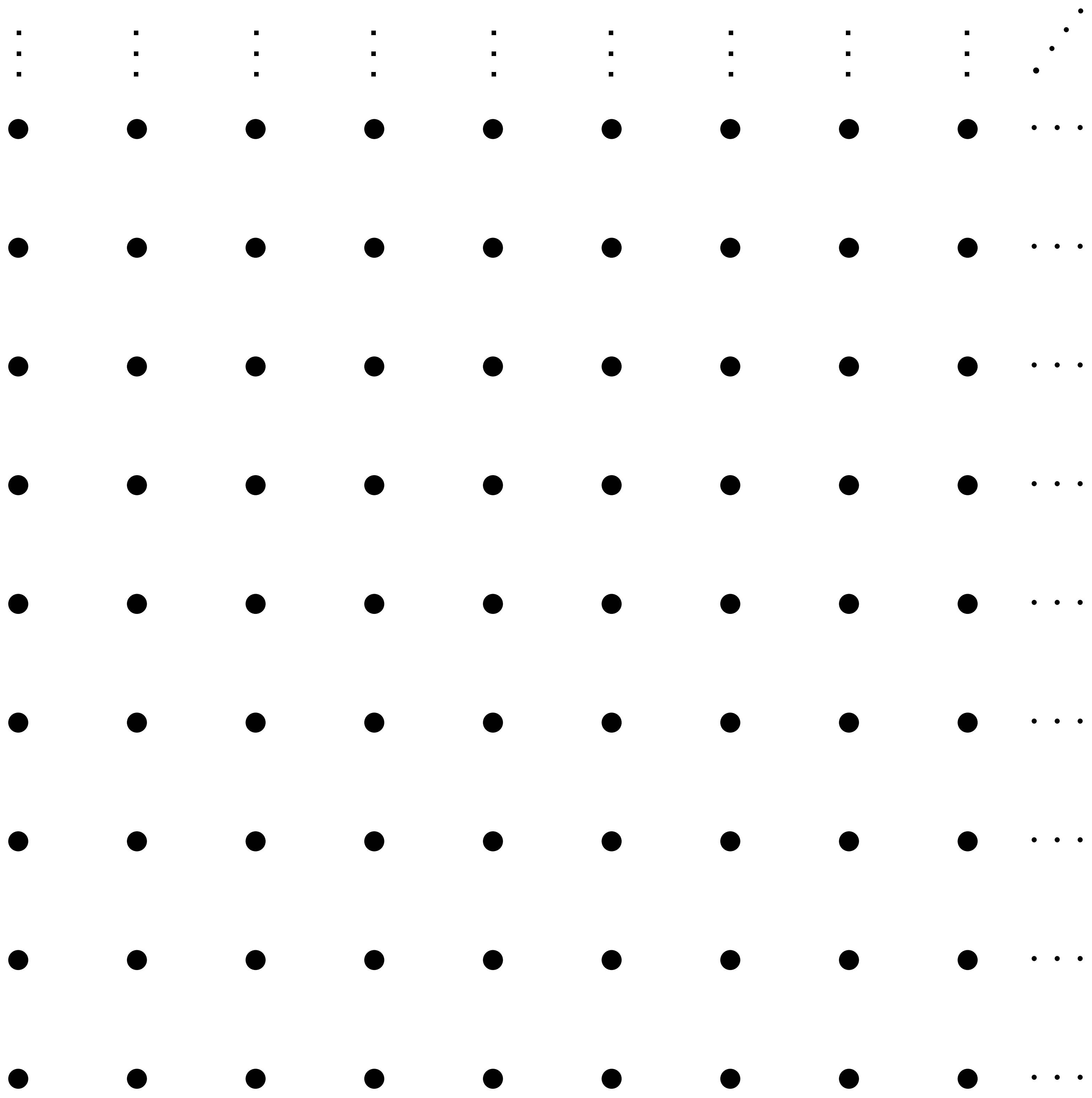
Definition. A set A is countable if there exists an injective function $A \to \mathbb{N}$, and it is uncountable otherwise.

exists injective function $A \to \mathbb{N}$ $\overset{*}{\iff}$ exists surjective function $f : \mathbb{N} \to A$

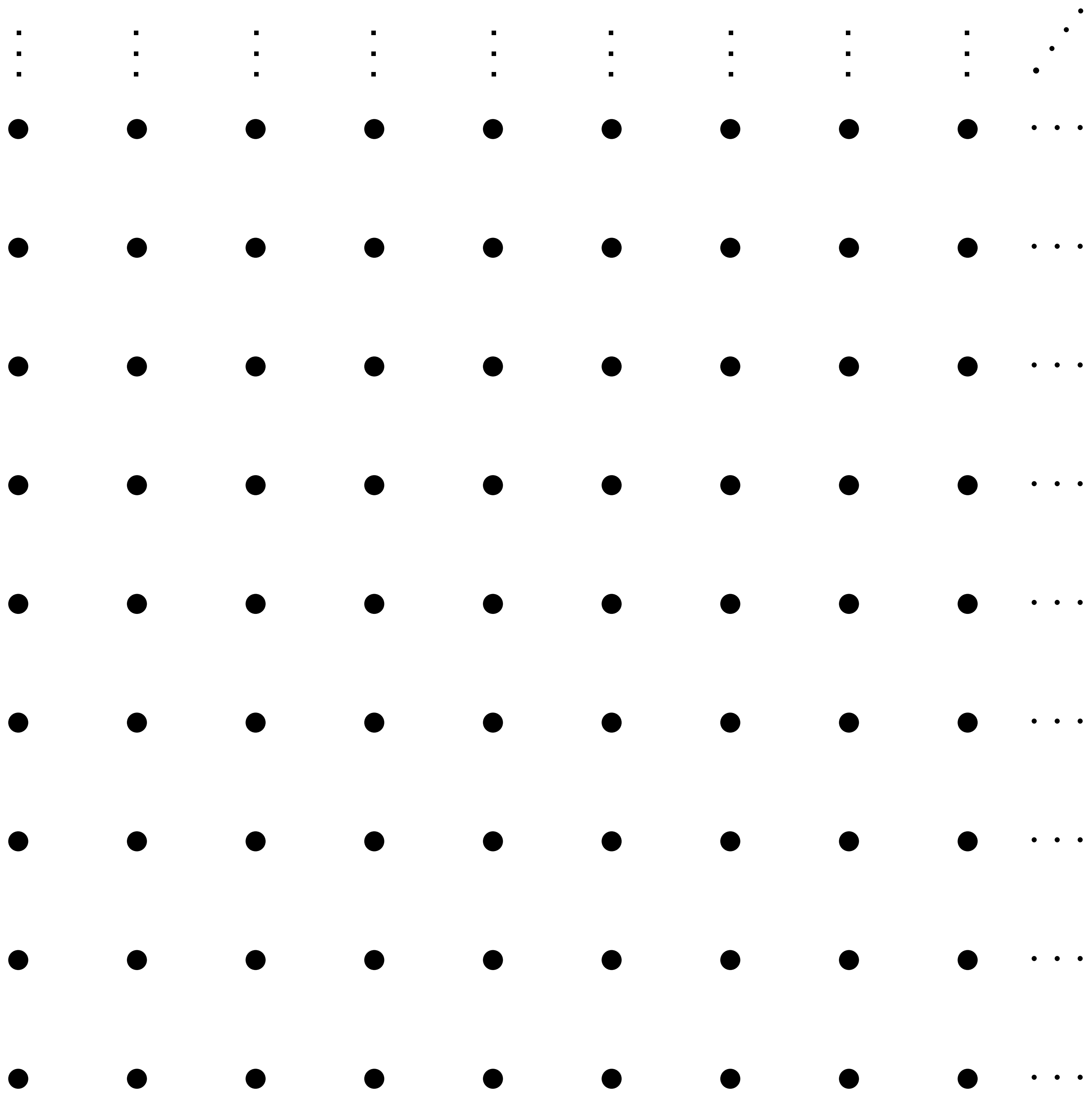
$$A = \{f(1), f(2), f(3), \dots\}$$

$$\boxed{\text{Countable set} = \text{can be represented on a computer}}$$

(*) Only if A is not empty

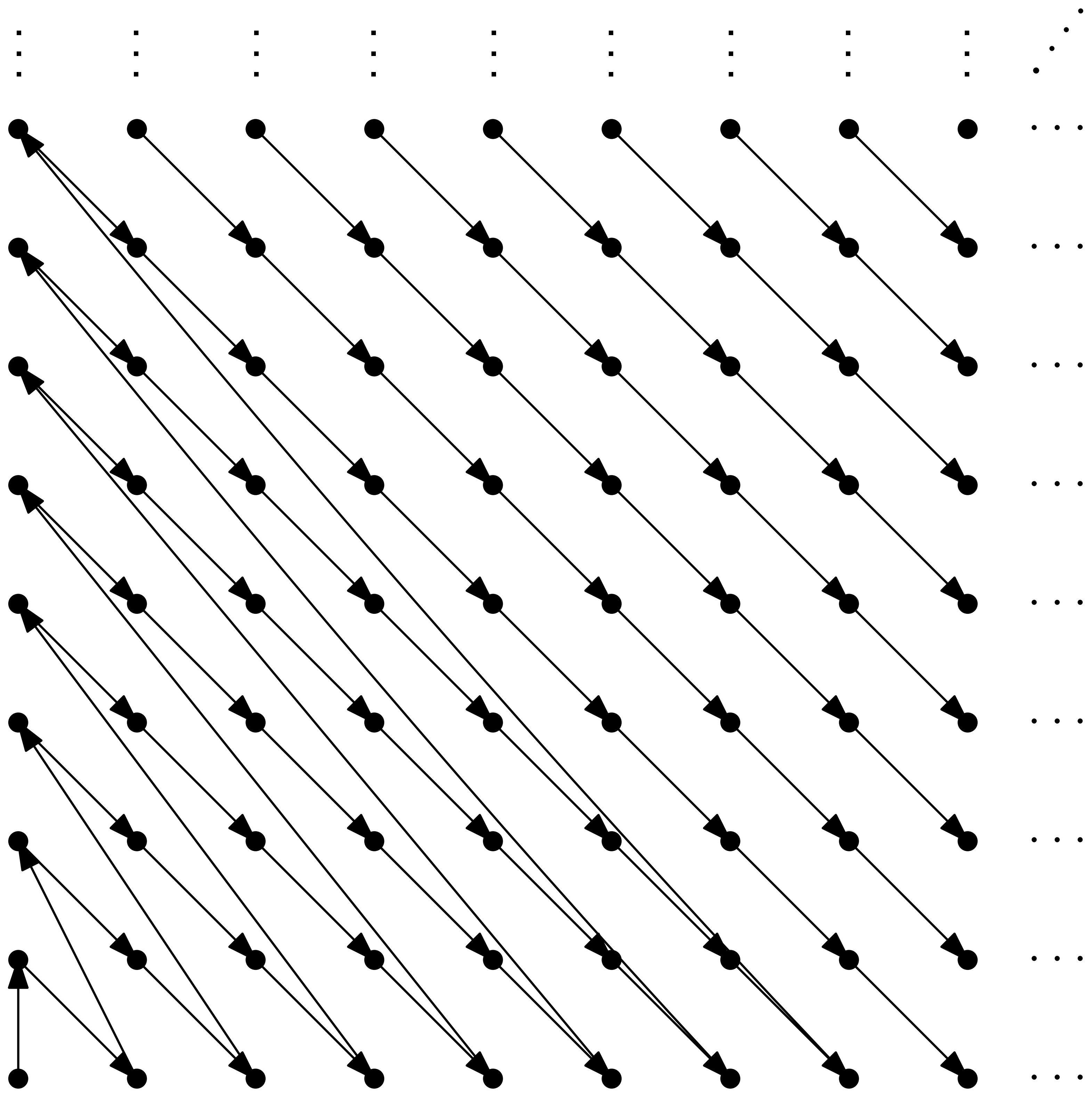


$\mathbb{N}^2$ is countable



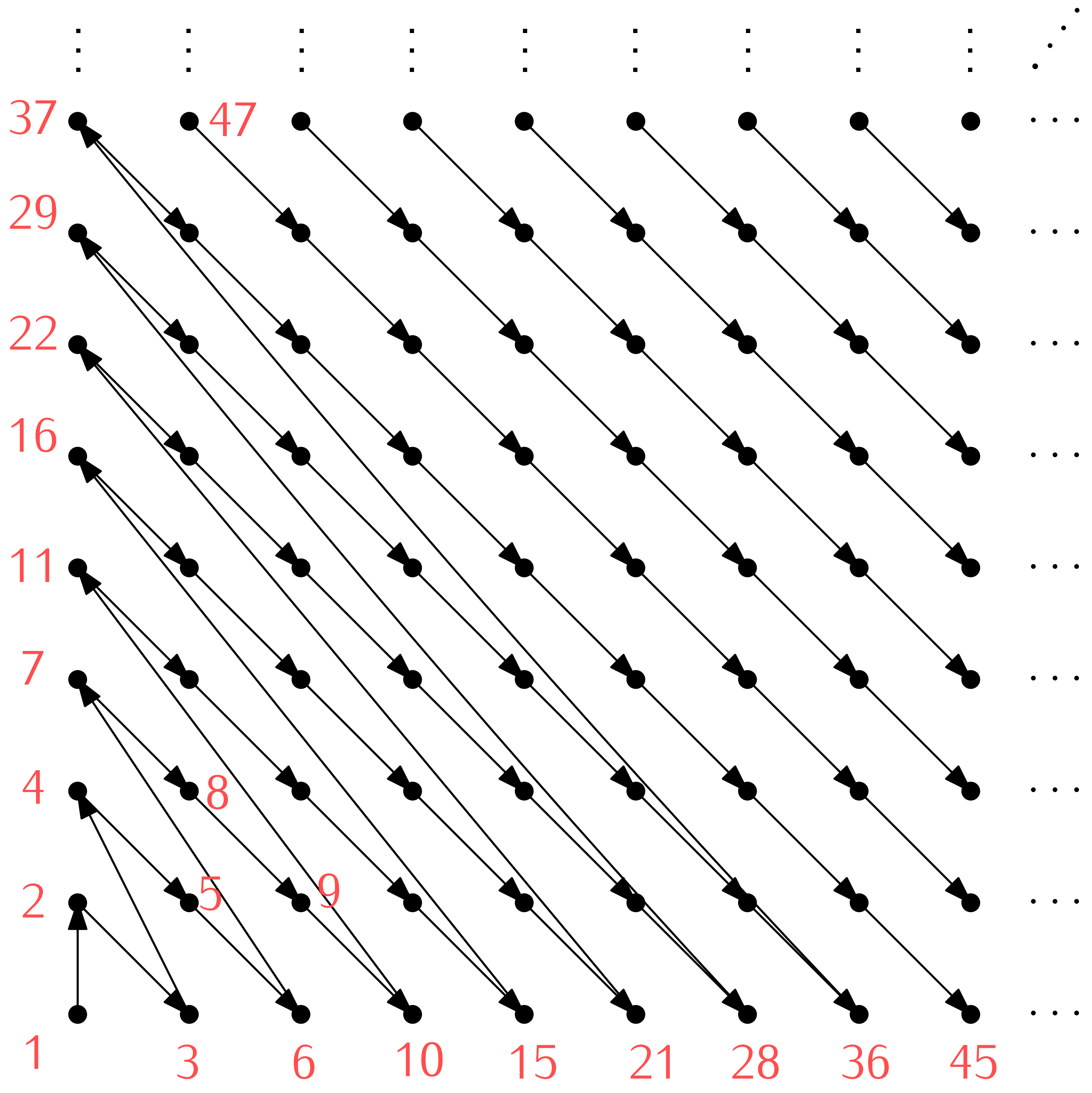
$$f : \mathbb{N}^2 \to \mathbb{N}$$

$$f(n, m) =$$



$f : \mathbb{N}^2 \to \mathbb{N}$

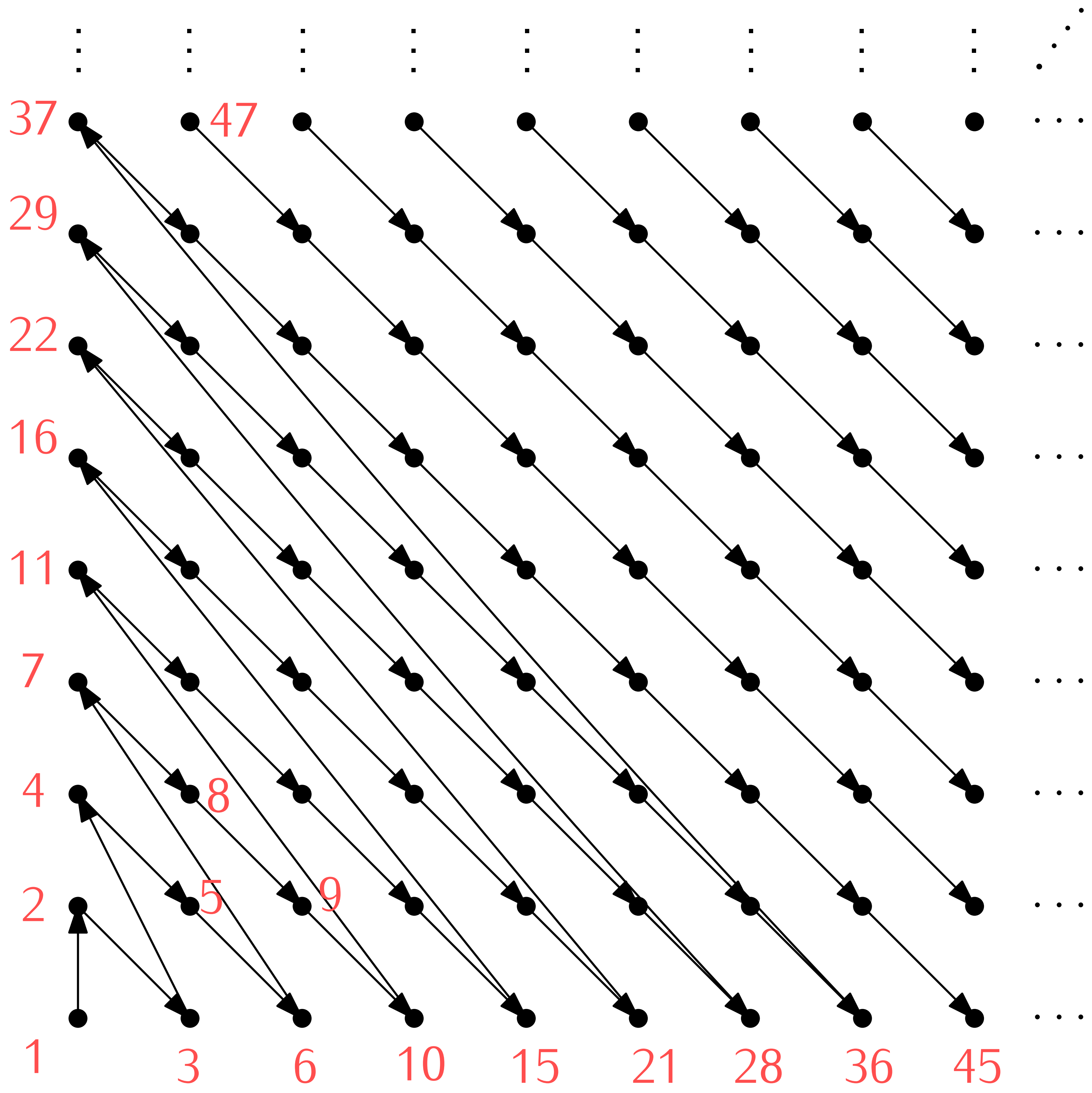
$f(n, m) = ((n + m)^2 - n - 3m + 2)/2$



$f : \mathbb{N}^2 \to \mathbb{N}$

$f(n, m) = ((n + m)^2 - n - 3m + 2)/2$

n	m	$f(n,m)$
1	1	1
1	2	2
1	3	4
1	4	7
2	1	3
3	3	13
$\vdots$	$\vdots$	$\vdots$

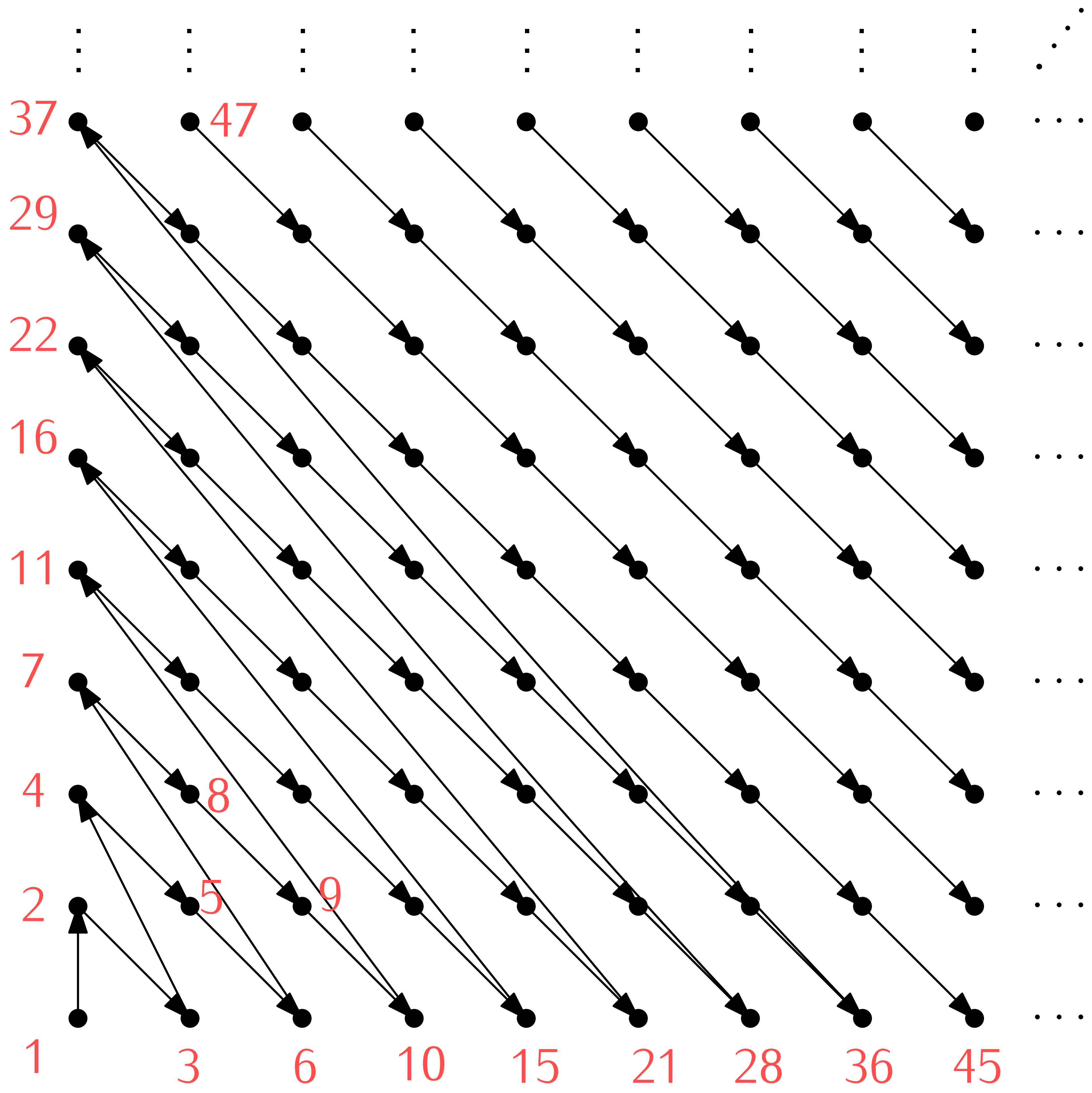


$f : \mathbb{N}^2 \to \mathbb{N}$

$f(n, m) = ((n + m)^2 - n - 3m + 2)/2$

n	m	$f(n, m)$
1	1	1
1	2	2
1	3	4
1	4	7
2	1	3
3	3	13
$\vdots$	$\vdots$	$\vdots$

Exercise: a) Prove that f is injective

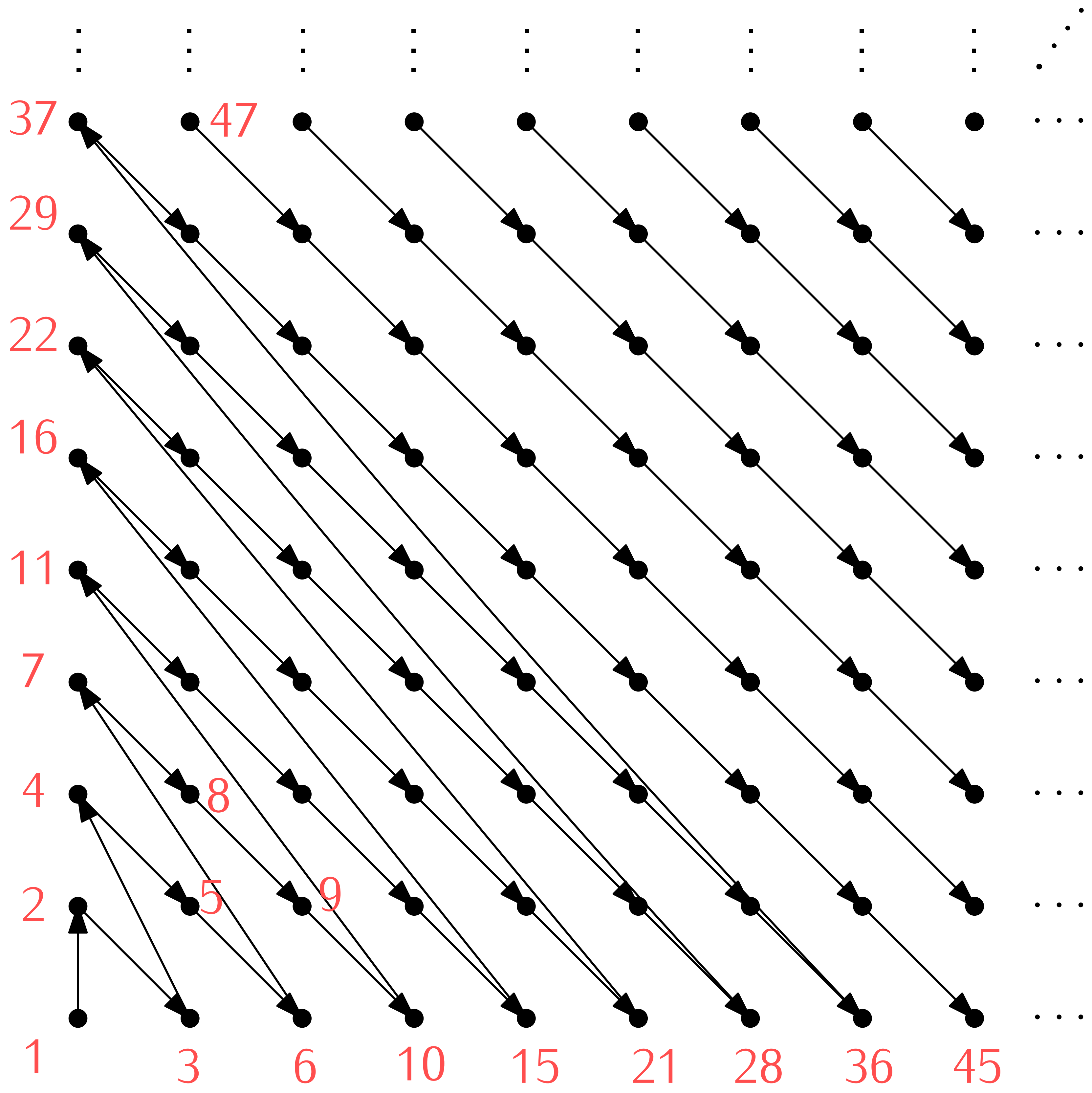


$f : \mathbb{N}^2 \to \mathbb{N}$

$f(n, m) = ((n + m)^2 - n - 3m + 2)/2$

n	m	$f(n,m)$
1	1	1
1	2	2
1	3	4
1	4	7
2	1	3
3	3	13
⋮	⋮	⋮

Exercise: a) Prove that f is injective
b) Prove that f is surjective



$f : \mathbb{N}^2 \to \mathbb{N}$

$f(n,m) = ((n+m)^2 - n - 3m + 2)/2$

n	m	$f(n,m)$
1	1	1
1	2	2
1	3	4
1	4	7
2	1	3
3	3	13
$\vdots$	$\vdots$	$\vdots$

Exercise: a) Prove that f is injective
b) Prove that f is surjective
c) Find $g : \mathbb{N} \to \mathbb{N}^2$ such that
- $g(f(n,m)) = (n,m)$ for all $n, m \in \mathbb{N}$
- $f(g(k)) = k$ for all $k \in \mathbb{N}$

Theorem. There is no surjective function $f\colon \mathbb{N} \to \mathbb{R}$.

Theorem. There is no surjective function $f\colon \mathbb{N} \to \mathbb{R}$.

Proof. By contradiction: suppose $f\colon \mathbb{N} \to \mathbb{R}$ is surjective.

$$
\begin{array}{rrccccccccc}
f(1) & 0. & 2 & 3 & 1 & 5 & 8 & 8 & 1 & 3 & \cdots \\
f(2) & 1. & 3 & 0 & 7 & 2 & 6 & 3 & 2 & 5 & \cdots \\
f(3) & -10. & 2 & 3 & 1 & 5 & 8 & 8 & 1 & 3 & \cdots \\
f(4) & 0. & 2 & 3 & 1 & 5 & 8 & 8 & 1 & 3 & \cdots \\
f(5) & 0. & 2 & 3 & 1 & 5 & 0 & 8 & 1 & 3 & \cdots \\
f(6) & 0. & 2 & 3 & 1 & 5 & 1 & 0 & 1 & 3 & \cdots \\
f(7) & 0. & 2 & 3 & 1 & 5 & 0 & 8 & 1 & 3 & \cdots \\
f(8) & 0. & 2 & 3 & 1 & 5 & 0 & 8 & 1 & 2 & \cdots \\
\vdots & & & & & & & & & &
\end{array}
$$

Theorem. There is no surjective function $f : \mathbb{N} \to \mathbb{R}$.

Proof. By contradiction: suppose $f : \mathbb{N} \to \mathbb{R}$ is surjective.

$$
\begin{array}{lrccccccccc}
f(1) & 0. & 2 & 3 & 1 & 5 & 8 & 8 & 1 & 3 & \cdots \\
f(2) & 1. & 3 & 0 & 7 & 2 & 6 & 3 & 2 & 5 & \cdots \\
f(3) & -10. & 2 & 3 & 1 & 5 & 8 & 8 & 1 & 3 & \cdots \\
f(4) & 0. & 2 & 3 & 1 & 5 & 8 & 8 & 1 & 3 & \cdots \\
f(5) & 0. & 2 & 3 & 1 & 5 & 0 & 8 & 1 & 3 & \cdots \\
f(6) & 0. & 2 & 3 & 1 & 5 & 1 & 0 & 1 & 3 & \cdots \\
f(7) & 0. & 2 & 3 & 1 & 5 & 0 & 8 & 1 & 3 & \cdots \\
f(8) & 0. & 2 & 3 & 1 & 5 & 0 & 8 & 1 & 2 & \cdots \\
\vdots & & & & & & & & & & \\
\text{Define } x = & 0. & & & & & & & & & \cdots
\end{array}
$$

Theorem. There is no surjective function $f\colon \mathbb{N} \to \mathbb{R}$.

Proof. By contradiction: suppose $f\colon \mathbb{N} \to \mathbb{R}$ is surjective.

$$
\begin{array}{rrccccccccc}
f(1) & 0. & \boxed{2} & 3 & 1 & 5 & 8 & 8 & 1 & 3 & \cdots \\
f(2) & 1. & 3 & \boxed{0} & 7 & 2 & 6 & 3 & 2 & 5 & \cdots \\
f(3) & -10. & 2 & 3 & \boxed{1} & 5 & 8 & 8 & 1 & 3 & \cdots \\
f(4) & 0. & 2 & 3 & 1 & \boxed{5} & 8 & 8 & 1 & 3 & \cdots \\
f(5) & 0. & 2 & 3 & 1 & 5 & \boxed{0} & 8 & 1 & 3 & \cdots \\
f(6) & 0. & 2 & 3 & 1 & 5 & 1 & \boxed{0} & 1 & 3 & \cdots \\
f(7) & 0. & 2 & 3 & 1 & 5 & 0 & 8 & \boxed{1} & 3 & \cdots \\
f(8) & 0. & 2 & 3 & 1 & 5 & 0 & 8 & 1 & \boxed{2} & \cdots \\
\vdots & & & & & & & & & & \\
\text{Define } x = & 0. & & & & & & & & & \cdots
\end{array}
$$

Theorem. There is no surjective function $f\colon \mathbb{N} \to \mathbb{R}$.

Proof. By contradiction: suppose $f\colon \mathbb{N} \to \mathbb{R}$ is surjective.

$$
\begin{array}{rrccccccccc}
f(1) & 0. & \boxed{2} & 3 & 1 & 5 & 8 & 8 & 1 & 3 & \cdots \\
f(2) & 1. & 3 & \boxed{0} & 7 & 2 & 6 & 3 & 2 & 5 & \cdots \\
f(3) & -10. & 2 & 3 & \boxed{1} & 5 & 8 & 8 & 1 & 3 & \cdots \\
f(4) & 0. & 2 & 3 & 1 & \boxed{5} & 8 & 8 & 1 & 3 & \cdots \\
f(5) & 0. & 2 & 3 & 1 & 5 & \boxed{0} & 8 & 1 & 3 & \cdots \\
f(6) & 0. & 2 & 3 & 1 & 5 & 1 & \boxed{0} & 1 & 3 & \cdots \\
f(7) & 0. & 2 & 3 & 1 & 5 & 0 & 8 & \boxed{1} & 3 & \cdots \\
f(8) & 0. & 2 & 3 & 1 & 5 & 0 & 8 & 1 & \boxed{2} & \cdots \\
\vdots & & & & & & & & & & \\
\text{Define } x = & 0. & \boxed{1} & \boxed{1} & \boxed{2} & \boxed{1} & \boxed{1} & \boxed{1} & \boxed{2} & \boxed{1} & \cdots
\end{array}
$$

Theorem. There is no surjective function $f\colon \mathbb{N} \to \mathbb{R}$.

Proof. By contradiction: suppose $f\colon \mathbb{N} \to \mathbb{R}$ is surjective.

$$
\begin{array}{rrccccccccc}
f(1) & 0. & \boxed{2} & 3 & 1 & 5 & 8 & 8 & 1 & 3 & \cdots \\
f(2) & 1. & 3 & \boxed{0} & 7 & 2 & 6 & 3 & 2 & 5 & \cdots \\
f(3) & -10. & 2 & 3 & \boxed{1} & 5 & 8 & 8 & 1 & 3 & \cdots \\
f(4) & 0. & 2 & 3 & 1 & \boxed{5} & 8 & 8 & 1 & 3 & \cdots \\
f(5) & 0. & 2 & 3 & 1 & 5 & \boxed{0} & 8 & 1 & 3 & \cdots \\
f(6) & 0. & 2 & 3 & 1 & 5 & 1 & \boxed{0} & 1 & 3 & \cdots \\
f(7) & 0. & 2 & 3 & 1 & 5 & 0 & 8 & \boxed{1} & 3 & \cdots \\
f(8) & 0. & 2 & 3 & 1 & 5 & 0 & 8 & 1 & \boxed{2} & \cdots \\
\vdots & & & & & & & & & & \\
\text{Define } x = & 0. & \boxed{1} & \boxed{1} & \boxed{2} & \boxed{1} & \boxed{1} & \boxed{1} & \boxed{2} & \boxed{1} & \cdots
\end{array}
$$

- Since f is surjective, we must have $x = f(n)$ for some n.

Theorem. There is no surjective function $f: \mathbb{N} \to \mathbb{R}$.

Proof. By contradiction: suppose $f: \mathbb{N} \to \mathbb{R}$ is surjective.

$$
\begin{array}{lrcccccccccc}
f(1) & 0. & \boxed{2} & 3 & 1 & 5 & 8 & 8 & 1 & 3 & \cdots \\
f(2) & 1. & 3 & \boxed{0} & 7 & 2 & 6 & 3 & 2 & 5 & \cdots \\
f(3) & -10. & 2 & 3 & \boxed{1} & 5 & 8 & 8 & 1 & 3 & \cdots \\
f(4) & 0. & 2 & 3 & 1 & \boxed{5} & 8 & 8 & 1 & 3 & \cdots \\
f(5) & 0. & 2 & 3 & 1 & 5 & \boxed{0} & 8 & 1 & 3 & \cdots \\
f(6) & 0. & 2 & 3 & 1 & 5 & 1 & \boxed{0} & 1 & 3 & \cdots \\
f(7) & 0. & 2 & 3 & 1 & 5 & 0 & 8 & \boxed{1} & 3 & \cdots \\
f(8) & 0. & 2 & 3 & 1 & 5 & 0 & 8 & 1 & \boxed{2} & \cdots \\
\vdots & & & & & & & & & & \\
\text{Define } x = & 0. & \boxed{1} & \boxed{1} & \boxed{2} & \boxed{1} & \boxed{1} & \boxed{1} & \boxed{2} & \boxed{1} & \cdots
\end{array}
$$

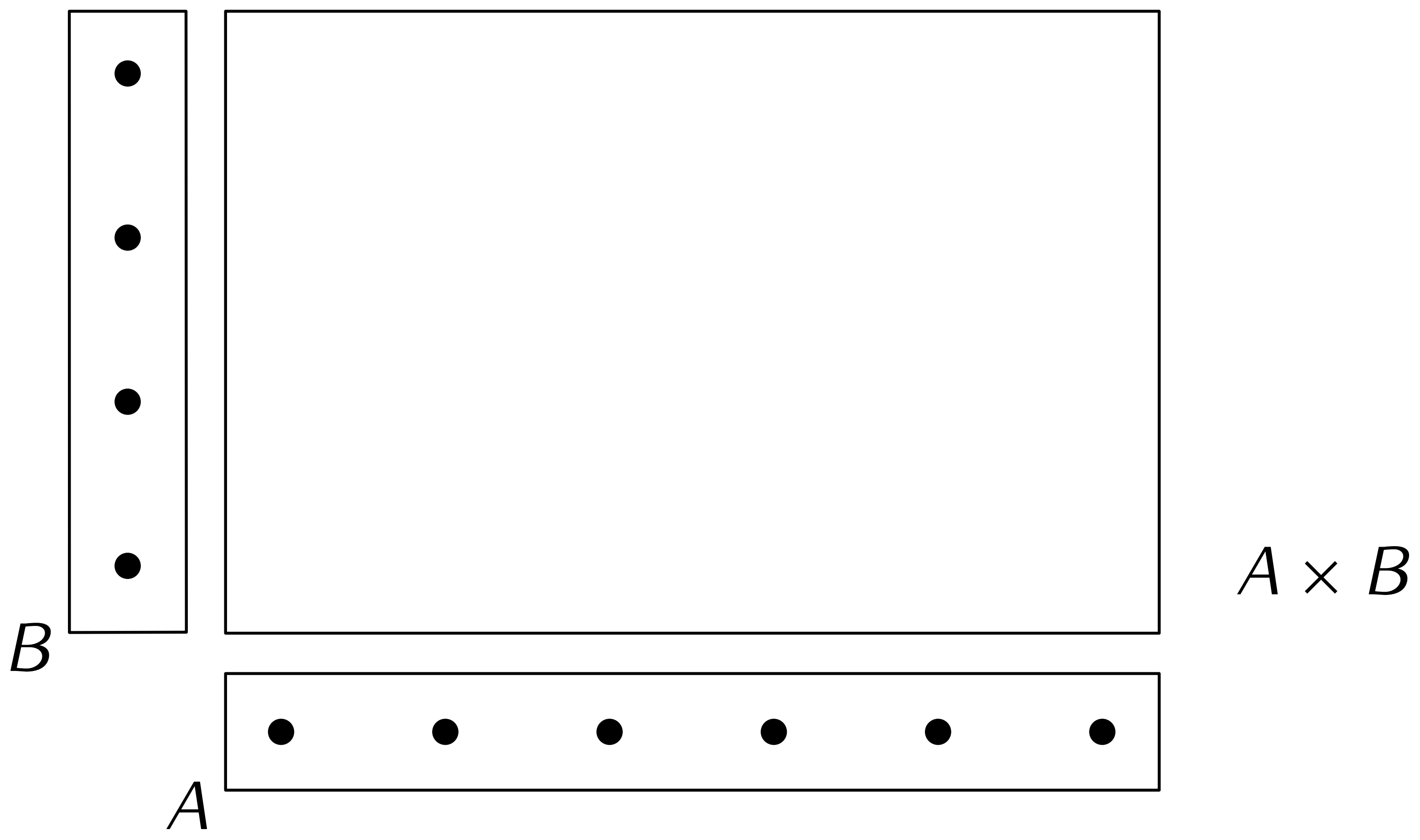
- Since f is surjective, we must have $x = f(n)$ for some n.
- But x and $f(n)$ differ on the nth digit after decimal point. $\square$

Basic counting rules and combinatorial proofs

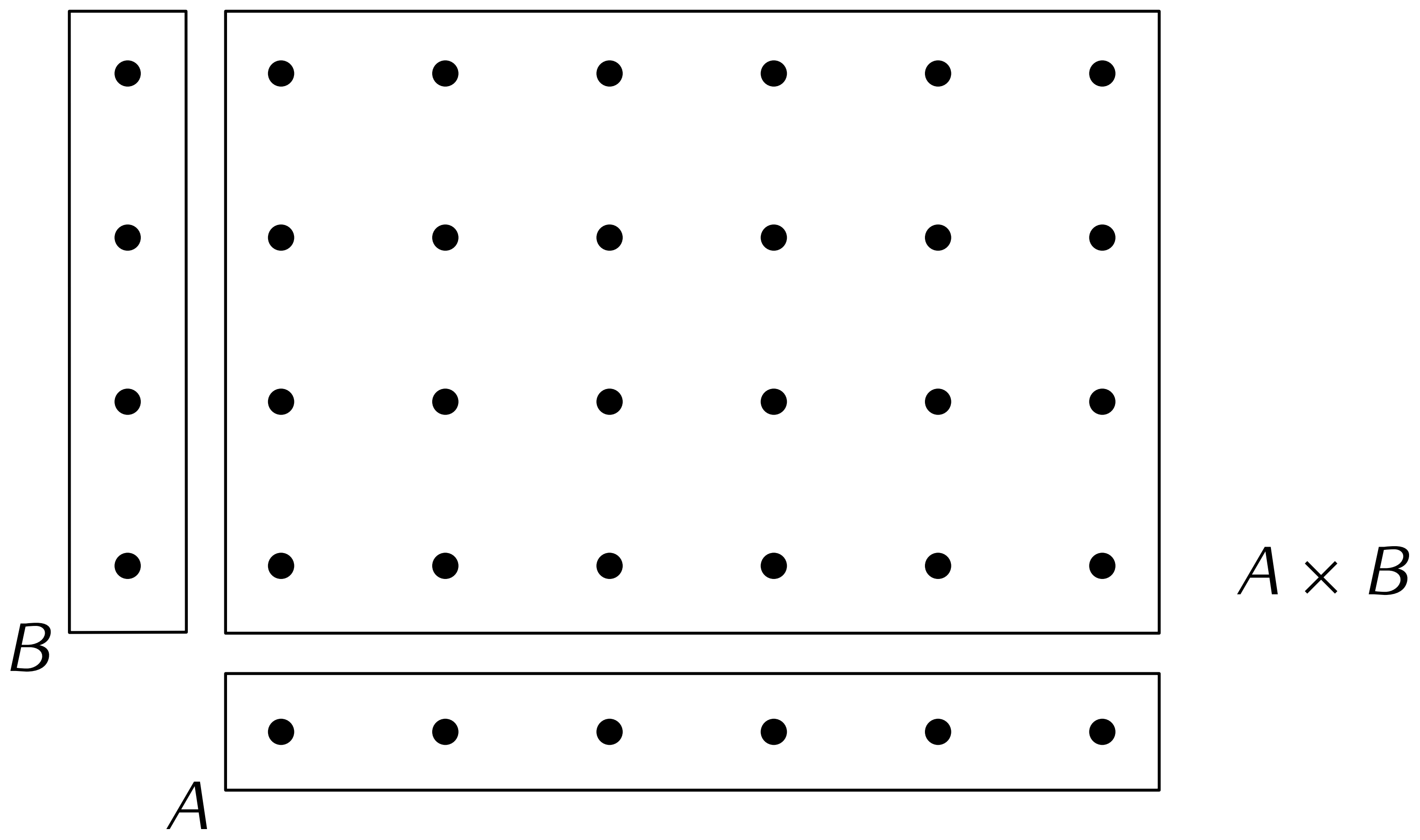
(Back to finite sets)

Notation. We write $|A|$ for the size of the set A.

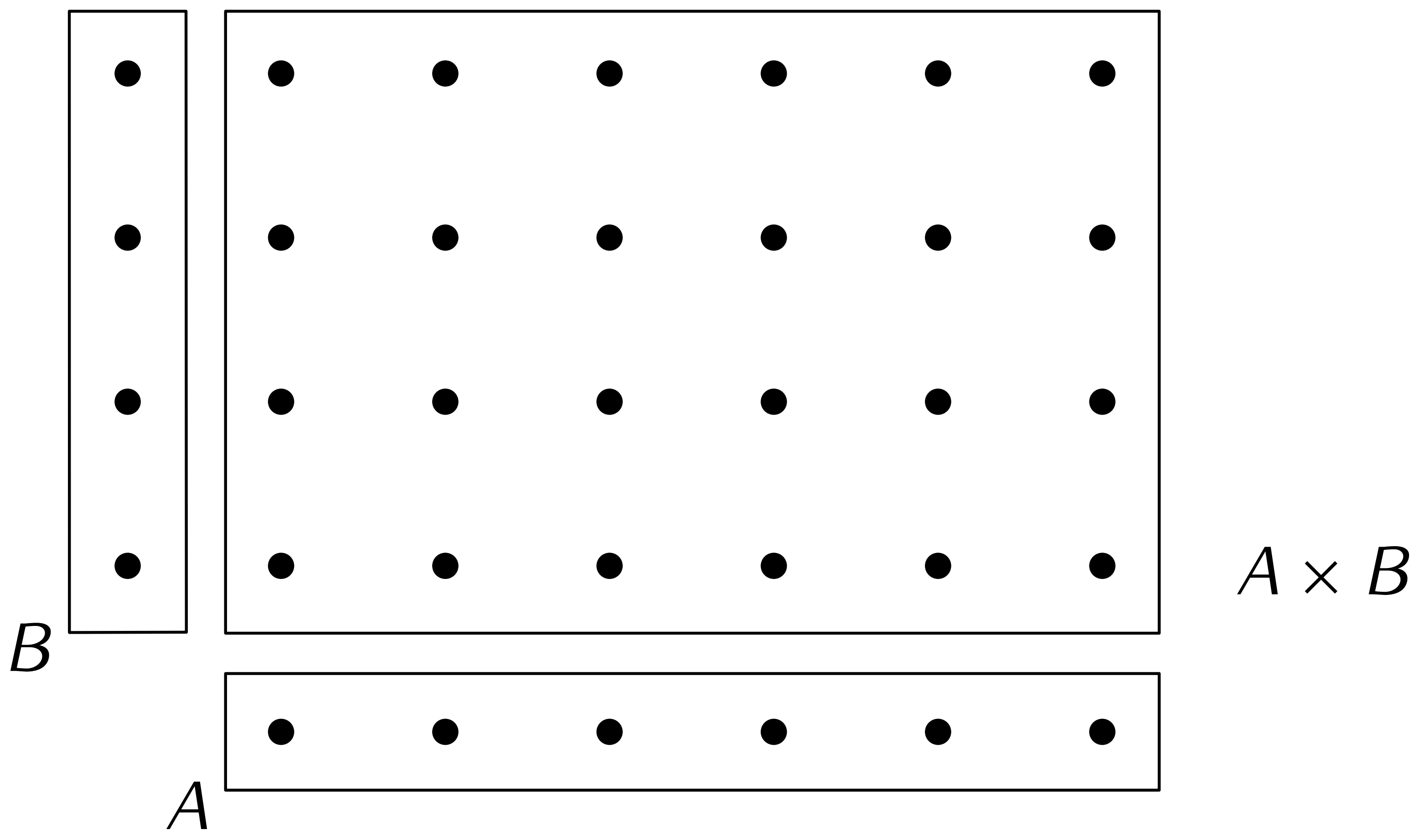
Notation. We write $|A|$ for the size of the set A.



Notation. We write $|A|$ for the size of the set A.

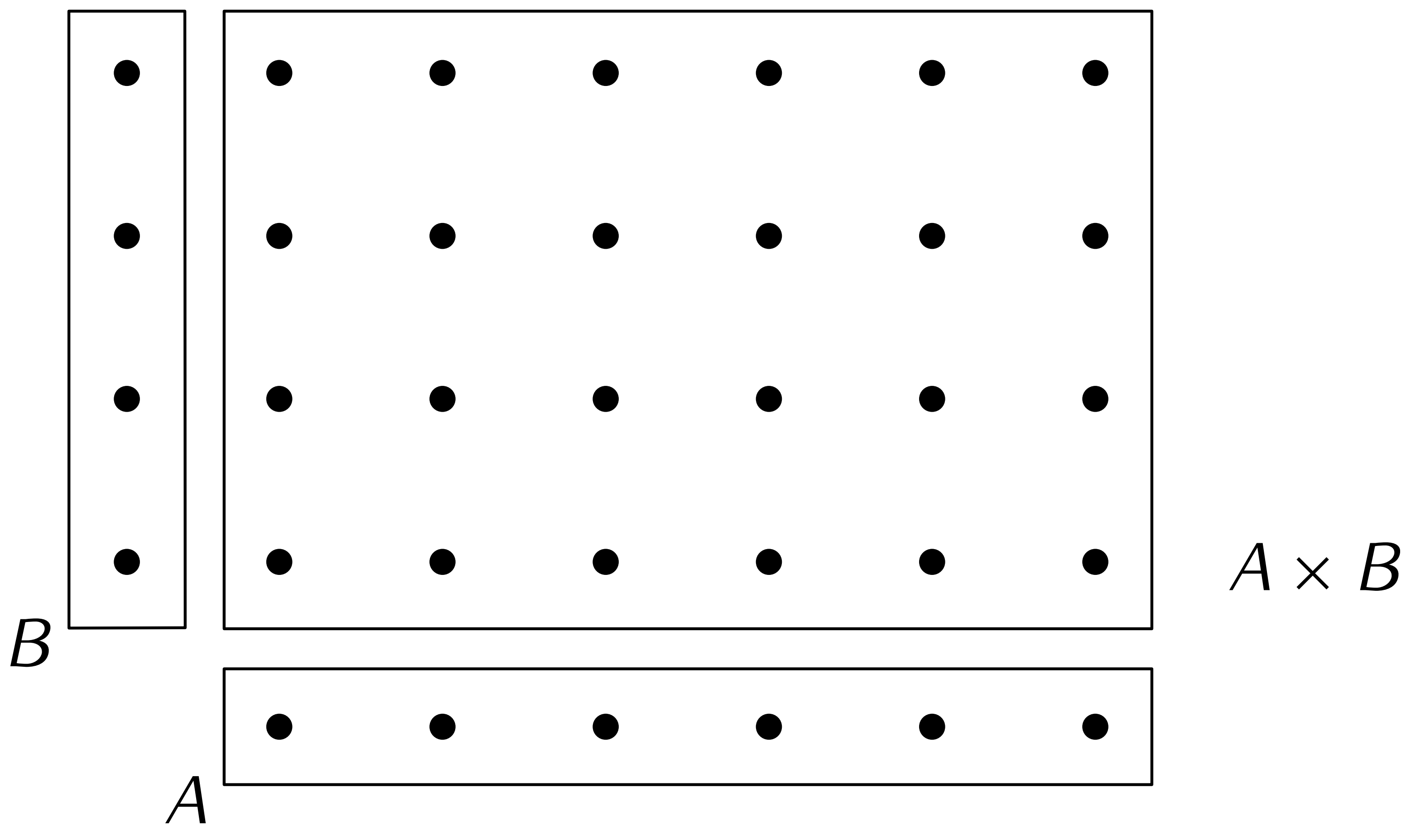


Notation. We write $|A|$ for the size of the set A.

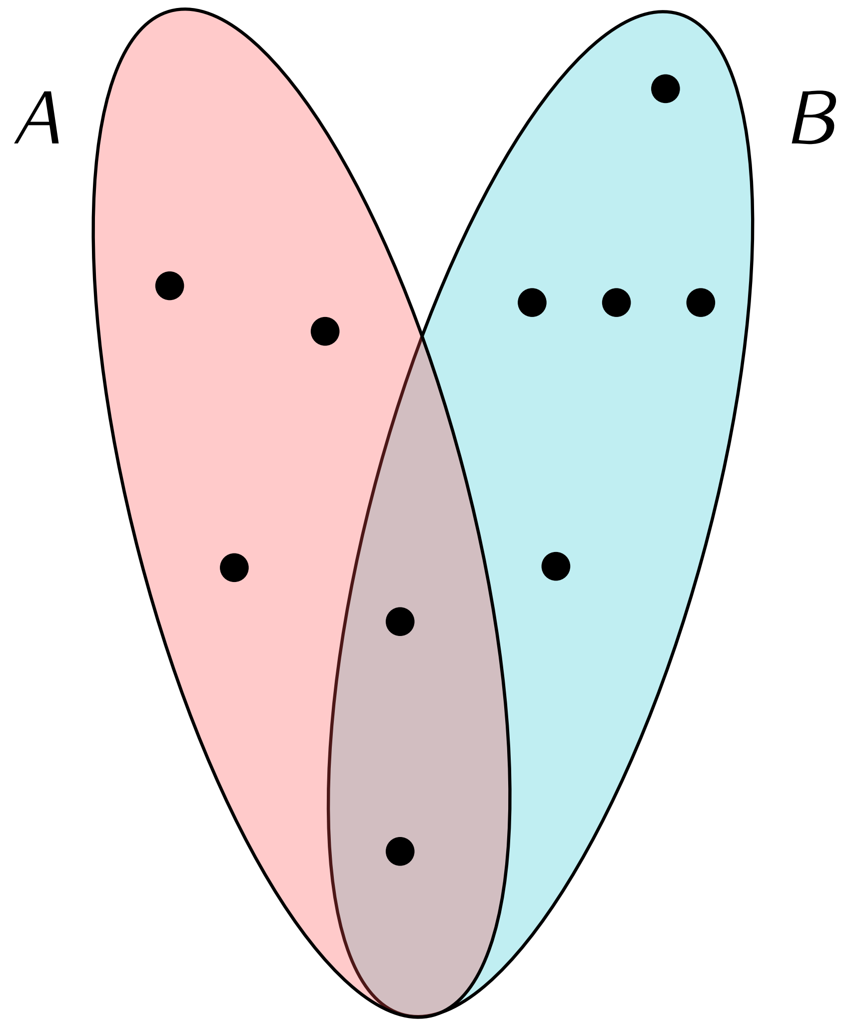


$$\Rightarrow |A \times B| = |A| \times |B|$$

Notation. We write $|A|$ for the size of the set A.



$$\Rightarrow |A \times B| = |A| \times |B|$$



$$|A| =$$
$$|B| =$$
$$|A \cup B| =$$

Notation. We write $|A|$ for the size of the set A.

Theorem. Let A, B be finite sets. Then:

- $|A \times B| = |A| \times |B|$
- $|A \cup B| = |A| + |B| - |A \cap B|$.

In particular $|A \cup B| \leq |A| + |B|$ and $|A \cup B| = |A| + |B|$ if $A \cap B = \emptyset$.

Notation. We write $|A|$ for the size of the set A.

Theorem. Let A, B be finite sets. Then:

- $|A \times B| = |A| \times |B|$
- $|A \cup B| = |A| + |B| - |A \cap B|$.

In particular $|A \cup B| \leq |A| + |B|$ and $|A \cup B| = |A| + |B|$ if $A \cap B = \emptyset$.

These are the product rule and the union rule.

Notation. We write $|A|$ for the size of the set A.

Theorem. Let A, B be finite sets. Then:
- $|A \times B| = |A| \times |B|$
- $|A \cup B| = |A| + |B| - |A \cap B|$.

In particular $|A \cup B| \leq |A| + |B|$ and $|A \cup B| = |A| + |B|$ if $A \cap B = \emptyset$.

more generally:
$|A| = |A_1| + |A_2| + \cdots + |A_n|$
if $\{A_1, \ldots, A_n\}$ is a
partition of A

These are the product rule and the union rule.

Notation. We write $|A|$ for the size of the set A.

Theorem. Let A, B be finite sets. Then:

- $|A \times B| = |A| \times |B|$
- $|A \cup B| = |A| + |B| - |A \cap B|$.

In particular $|A \cup B| \leq |A| + |B|$ and $|A \cup B| = |A| + |B|$ if $A \cap B = \emptyset$.

more generally:
$|A| = |A_1| + |A_2| + \cdots + |A_n|$
if $\{A_1, \ldots, A_n\}$ is a partition of A

These are the product rule and the union rule.

Examples. We obtain from the basic rules:

- Number of binary strings of length 2:
- Number of binary strings of length 3:
- Number of binary strings of length n:
- Number of binary strings of length $\leq n$:

Definition. Let $p \leq q$ and $a_p, \ldots, a_q$ in a set where $+$ makes sense. Then $\sum_{i=p}^{q} a_i = a_p + a_{p+1} + \cdots + a_q$.

Definition. Let $p \leq q$ and $a_p, \ldots, a_q$ in a set where $+$ makes sense. Then $\sum_{i=p}^{q} a_i = a_p + a_{p+1} + \cdots + a_q$.

```
def sum_symbol(A,p,q):
  result = 0
  for i in range(p,q+1):
    result = result + A[i]
```

Definition. Let $p \leq q$ and $a_p, \ldots, a_q$ in a set where $+$ makes sense. Then $\sum_{i=p}^{q} a_i = a_p + a_{p+1} + \cdots + a_q$.

Empty sum (when $q < p$) is 0.

```
def sum_symbol(A,p,q):
  result = 0
  for i in range(p,q+1):
    result = result + A[i]
```

Definition. Let $p \leq q$ and $a_p, \ldots, a_q$ in a set where $+$ makes sense. Then $\sum_{i=p}^{q} a_i = a_p + a_{p+1} + \cdots + a_q$.

Empty sum (when $q < p$) is 0.

```
def sum_symbol(A,p,q):
  result = 0
  for i in range(p,q+1):
    result = result + A[i]
```

Similar notation exists for many other operators:

- $\bigcup_{i=p}^{q} A_i = A_p \cup A_{p+1} \cup \cdots \cup A_q$

```
def union_symbol(A,p,q):
  result = set() # initialize to be the empty set
  for i in range(p,q+1):
    result = result.update(A[i]) # take the union
```

Definition. Let $p \leq q$ and $a_p, \ldots, a_q$ in a set where $+$ makes sense. Then $\sum_{i=p}^{q} a_i = a_p + a_{p+1} + \cdots + a_q$.

Empty sum (when $q < p$) is 0.

```
def sum_symbol(A,p,q):
  result = 0
  for i in range(p,q+1):
    result = result + A[i]
```

Similar notation exists for many other operators:

- $\bigcup_{i=p}^{q} A_i = A_p \cup A_{p+1} \cup \cdots \cup A_q$

  Empty union (when $q < p$) is $\emptyset$.

```
def union_symbol(A,p,q):
  result = set() # initialize to be the empty set
  for i in range(p,q+1):
    result = result.update(A[i]) # take the union
```

Definition. Let $p \leq q$ and $a_p, \ldots, a_q$ in a set where $+$ makes sense. Then $\sum_{i=p}^{q} a_i = a_p + a_{p+1} + \cdots + a_q$.

Empty sum (when $q < p$) is 0.

```
def sum_symbol(A,p,q):
  result = 0
  for i in range(p,q+1):
    result = result + A[i]
```

Similar notation exists for many other operators:

- $\bigcup_{i=p}^{q} A_i = A_p \cup A_{p+1} \cup \cdots \cup A_q$

  Empty union (when $q < p$) is $\emptyset$.

```
def union_symbol(A,p,q):
  result = set() # initialize to be the empty set
  for i in range(p,q+1):
    result = result.update(A[i]) # take the union
```

- $\prod_{i=p}^{q} a_i = a_p \times a_{p+1} \times \cdots \times a_q$

```
def product_symbol(A,p,q):
  result = ?   # initialize
  for i in range(p,q+1):
    result = result * A[i]
```

Definition. Let $p \leq q$ and $a_p, \ldots, a_q$ in a set where $+$ makes sense. Then $\sum_{i=p}^{q} a_i = a_p + a_{p+1} + \cdots + a_q$.

Empty sum (when $q < p$) is 0.

```
def sum_symbol(A,p,q):
  result = 0
  for i in range(p,q+1):
    result = result + A[i]
```

Similar notation exists for many other operators:

- $\bigcup_{i=p}^{q} A_i = A_p \cup A_{p+1} \cup \cdots \cup A_q$

  Empty union (when $q < p$) is $\emptyset$.

```
def union_symbol(A,p,q):
  result = set() # initialize to be the empty set
  for i in range(p,q+1):
    result = result.update(A[i]) # take the union
```

- $\prod_{i=p}^{q} a_i = a_p \times a_{p+1} \times \cdots \times a_q$

  Empty product (when $q < p$) is 1.

```
def product_symbol(A,p,q):
  result = 1   # initialize
  for i in range(p,q+1):
    result = result * A[i]
```

Theorem. Let A, B be finite sets. Let $f\colon A \to B$.

- If f is injective, then $|A| \leq |B|$.
- If f is surjective, then $|A| \geq |B|$.
- If f is bijective, then $|A| = |B|$.

Theorem. Let A, B be finite sets. Let $f\colon A \to B$.

- If f is injective, then $|A| \leq |B|$.
- If f is surjective, then $|A| \geq |B|$.
- If f is bijective, then $|A| = |B|$.

Careful! Even if $|A| \leq |B|$, it does not mean that f must be injective!

Theorem. Let A, B be finite sets. Let $f\colon A \to B$.

- If f is injective, then $|A| \leq |B|$.
- If f is surjective, then $|A| \geq |B|$.
- If f is bijective, then $|A| = |B|$.

Careful! Even if $|A| \leq |B|$, it does not mean that f must be injective!

A combinatorial proof of an equality between numbers $n = m$ consists of the following steps:

1. We find A such that $|A| = n$.
2. We find B such that $|B| = m$.
3. We find a function $f\colon A \to B$.
4. We prove that f is a bijection.

Theorem. Let A, B be finite sets. Let $f\colon A \to B$.

- If f is injective, then $|A| \leq |B|$.
- If f is surjective, then $|A| \geq |B|$.
- If f is bijective, then $|A| = |B|$.

Careful! Even if $|A| \leq |B|$, it does not mean that f must be injective!

A combinatorial proof of an equality between numbers $n = m$ consists of the following steps:

1. We find A such that $|A| = n$.
2. We find B such that $|B| = m$.
3. We find a function $f\colon A \to B$.
4. We prove that f is a bijection.

How do we "find a set"? Use the basic rules!

- symbol $+$ or $\sum$: union of disjoint sets
- symbol $\times$ or $\prod$ (or power): product of sets
- a number (like 5): any set of that size, for example $\{0, \ldots, 4\}$

Theorem. Let A, B be finite sets. Let $f\colon A \to B$.
- If f is injective, then $|A| \leq |B|$.
- If f is surjective, then $|A| \geq |B|$.
- If f is bijective, then $|A| = |B|$.

Careful! Even if $|A| \leq |B|$, it does not mean that f must be injective!

A combinatorial proof of an equality between numbers $n = m$ consists of the following steps:
1. We find A such that $|A| = n$.
2. We find B such that $|B| = m$.
3. We find a function $f\colon A \to B$.
4. We prove that f is a bijection.

How do we "find a set"? Use the basic rules!
- symbol $+$ or $\sum$: union of disjoint sets
- symbol $\times$ or $\prod$ (or power): product of sets
- a number (like 5): any set of that size, for example $\{0, \dots, 4\}$

How do we find a function $f\colon A \to B$?
Draw a picture and work from there.

Theorem. For all $n \geq 1$, $\sum_{k=0}^{n-1}(2k+1) = n^2$.

Theorem. For all $n \geq 1$, $\sum_{k=0}^{n-1}(2k+1) = n^2$.

$\sum$ corresponds to union of disjoint sets

$A = A_1 \cup \cdots \cup A_{n-1}$

Theorem. For all $n \geq 1$, $\sum_{k=0}^{n-1}(2k+1) = n^2$.

$\sum$ corresponds to union of disjoint sets

$A = A_1 \cup \cdots \cup A_{n-1}$

$\times$ corresponds to product of sets

$B = C^2$

Theorem. For all $n \geq 1$, $\sum_{k=0}^{n-1} (2k+1) = n^2$.

$\times$ corresponds to product of sets
$B = C^2$

$\sum$ corresponds to union of disjoint sets
$A = A_1 \cup \cdots \cup A_{n-1}$

A_k has size $2k+1$ and C has size n

Theorem. For all $n \geq 1$, $\sum_{k=0}^{n-1}(2k+1) = n^2$.

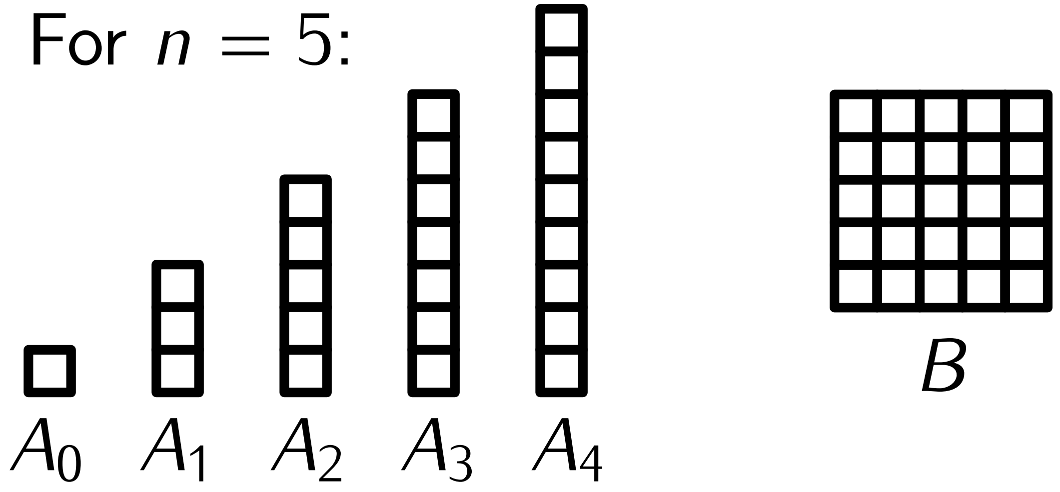
$\times$ corresponds to product of sets
$B = C^2$

$\sum$ corresponds to union of disjoint sets
$A = A_1 \cup \cdots \cup A_{n-1}$

A_k has size $2k+1$ and C has size n

direct translation of the equality!

For $n = 5$:



Theorem. For all $n \geq 1$, $\sum_{k=0}^{n-1}(2k+1) = n^2$.

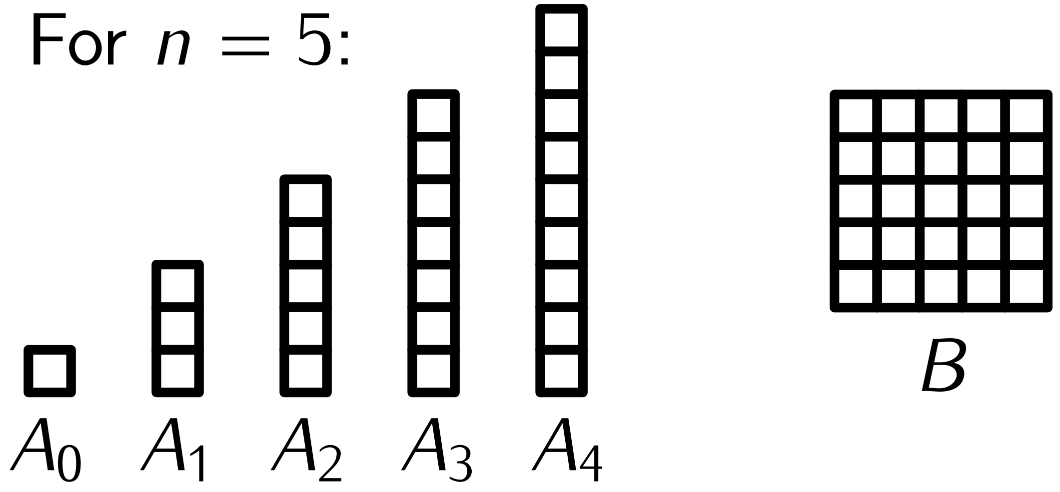
$\times$ corresponds to product of sets
$B = C^2$

$\sum$ corresponds to union of disjoint sets
$A = A_1 \cup \cdots \cup A_{n-1}$

A_k has size $2k+1$ and C has size n

direct translation of the equality!

For $n = 5$:



The theorem says: there are as many squares on the left as there are on the right

Example of a combinatorial proof

Antoine Wiehe

Theorem. For all $n \geq 1$, $\sum_{k=0}^{n-1} (2k+1) = n^2$.

$\sum$ corresponds to union of disjoint sets

$A = A_1 \cup \cdots \cup A_{n-1}$

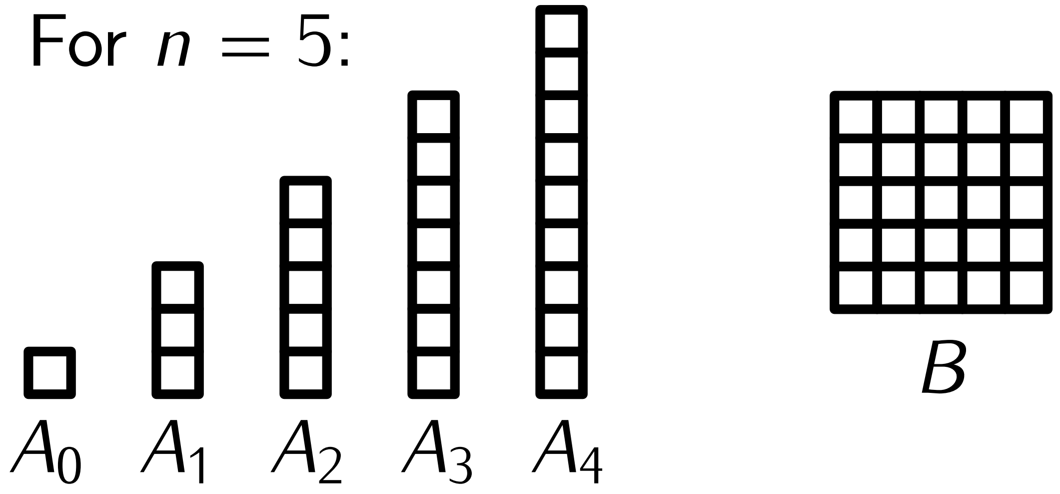
$A_k = \{0, \ldots, 2k\}$?

A_k has size $2k+1$ and C has size n

$\times$ corresponds to product of sets $B = C^2$

direct translation of the equality!

For $n = 5$:



The theorem says: there are as many squares on the left as there are on the right

Theorem. For all $n \geq 1$, $\sum_{k=0}^{n-1}(2k+1) = n^2$.

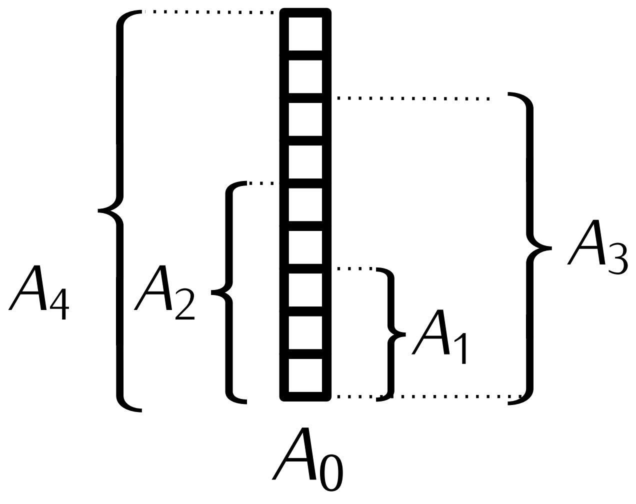
$\sum$ corresponds to union of disjoint sets

$A = A_1 \cup \cdots \cup A_{n-1}$

$A_k = \{0, \ldots, 2k\}$?

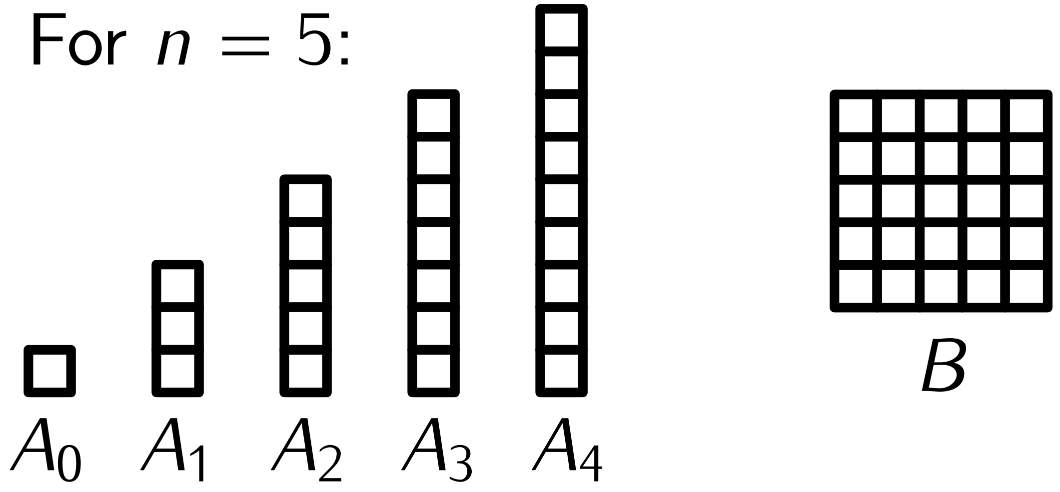
A_k has size $2k+1$ and C has size n

$\times$ corresponds to product of sets $B = C^2$



direct translation of the equality!

For $n = 5$:



The theorem says: there are as many squares on the left as there are on the right

Theorem. For all $n \geq 1$, $\sum_{k=0}^{n-1}(2k+1) = n^2$.

$\sum$ corresponds to union of disjoint sets

$A = A_1 \cup \cdots \cup A_{n-1}$

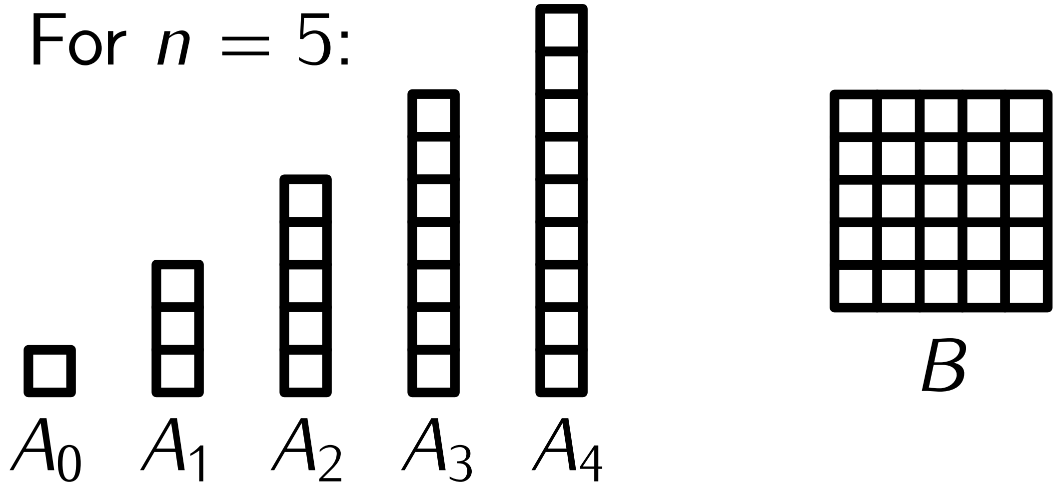
$A_k = \{(k,0), \ldots, (k,2k)\}$

A_k has size $2k+1$ and C has size n

$\times$ corresponds to product of sets $B = C^2$

direct translation of the equality!

For $n = 5$:



The theorem says: there are as many squares on the left as there are on the right

Theorem. For all $n \geq 1$, $\sum_{k=0}^{n-1}(2k+1) = n^2$.

$\times$ corresponds to product of sets
$B = C^2$

$\sum$ corresponds to union of disjoint sets

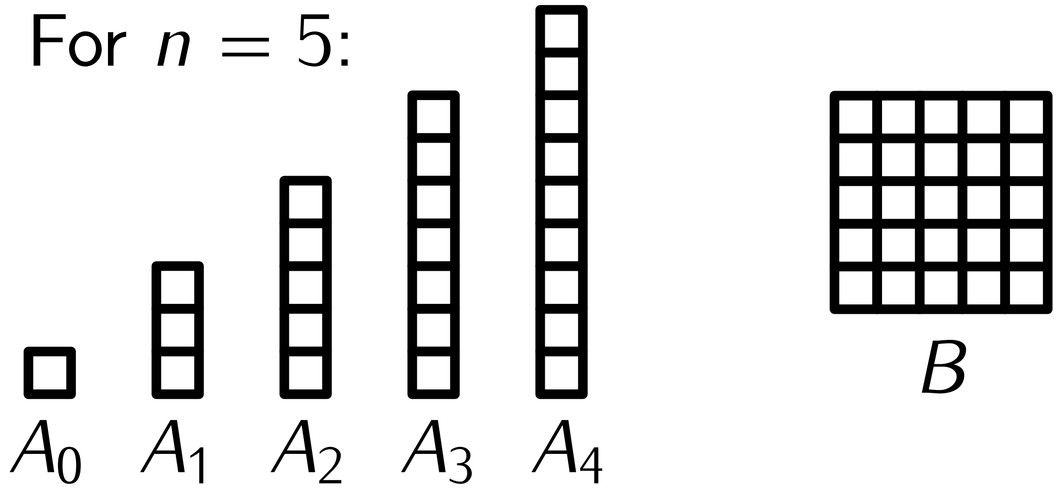
$A = A_1 \cup \cdots \cup A_{n-1}$

$A_k = \{(k,0), \ldots, (k,2k)\}$

A_k has size $2k+1$ and C has size n

direct translation of the equality!

For $n = 5$:



It remains to find the bijection $f : A \to B$!

The theorem says: there are as many squares on the left as there are on the right

Theorem. For all $n \geq 1$, $\sum_{k=0}^{n-1}(2k+1) = n^2$.

$\sum$ corresponds to union of disjoint sets

$$A = A_1 \cup \dots \cup A_{n-1}$$

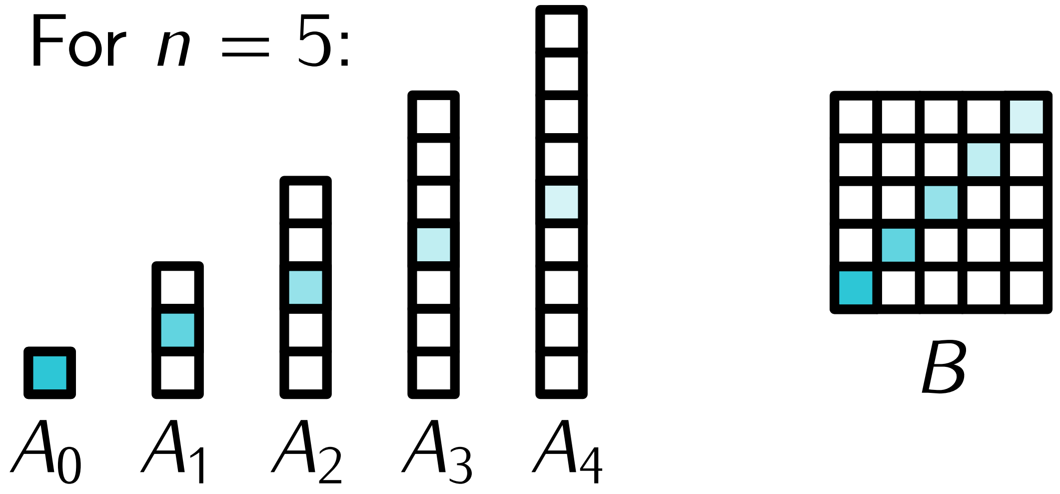
$$A_k = \{(k,0), \dots, (k,2k)\}$$

A_k has size $2k+1$ and C has size n

$\times$ corresponds to product of sets $B = C^2$

direct translation of the equality!

For $n = 5$:



It remains to find the bijection $f\colon A \to B$!

$$f(k,i) = \begin{cases} (k,k) & i = k \\ & i < k \\ & i > k \end{cases}$$

The theorem says: there are as many squares on the left as there are on the right

Theorem. For all $n \geq 1$, $\sum_{k=0}^{n-1}(2k+1) = n^2$.

$\sum$ corresponds to union of disjoint sets

$A = A_1 \cup \cdots \cup A_{n-1}$

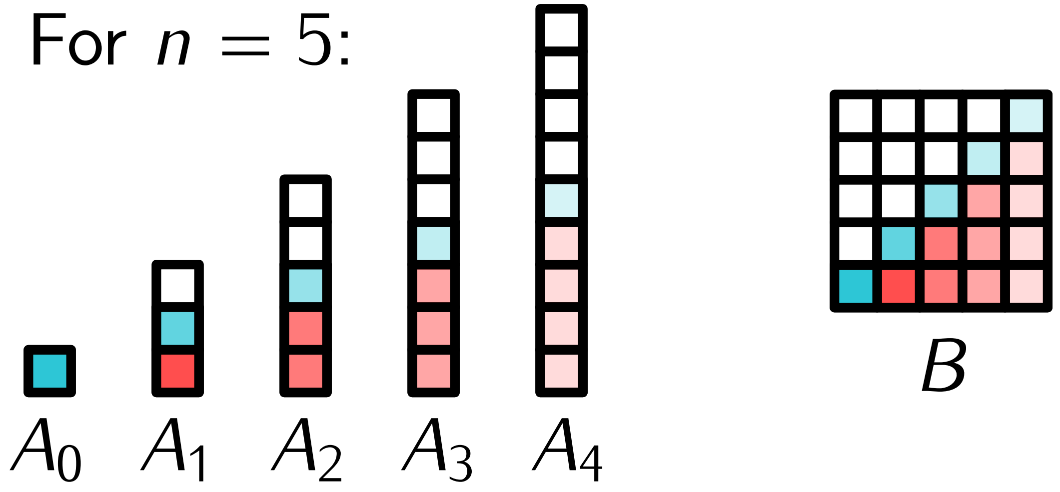
$A_k = \{(k,0), \ldots, (k,2k)\}$

A_k has size $2k+1$ and C has size n

$\times$ corresponds to product of sets $B = C^2$

direct translation of the equality!

For $n = 5$:



It remains to find the bijection $f\colon A \to B$!

$$f(k,i) = \begin{cases} (k,k) & i = k \\ (k,i) & i < k \\ & i > k \end{cases}$$

The theorem says: there are as many squares on the left as there are on the right

Theorem. For all $n \geq 1$, $\sum_{k=0}^{n-1}(2k+1) = n^2$.

$\times$ corresponds to product of sets
$B = C^2$

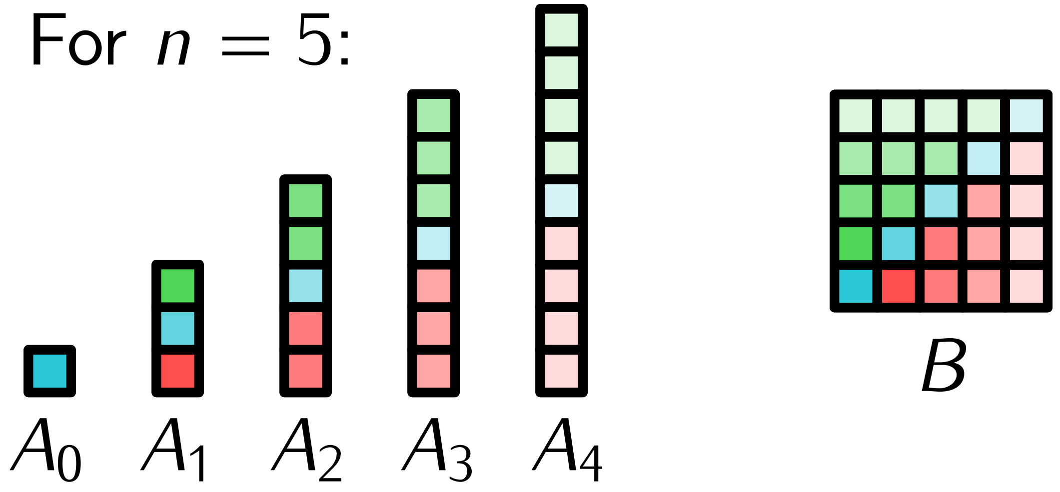
$\sum$ corresponds to union
of disjoint sets

$$A = A_1 \cup \cdots \cup A_{n-1}$$
$$A_k = \{(k,0), \ldots, (k,2k)\}$$

A_k has size $2k+1$ and
C has size n

direct translation of the equality!

For $n = 5$:



It remains to find the bijection $f\colon A \to B$!

$$f(k,i) = \begin{cases} (k,k) & i = k \\ (k,i) & i < k \\ (i,k) & i > k \end{cases}$$

The theorem says: there are as many squares
on the left as there are on the right

Theorem. For all $n \geq 1$, $\sum_{k=0}^{n-1}(2k+1) = n^2$.

$\times$ corresponds to product of sets
$B = C^2$

$\sum$ corresponds to union of disjoint sets

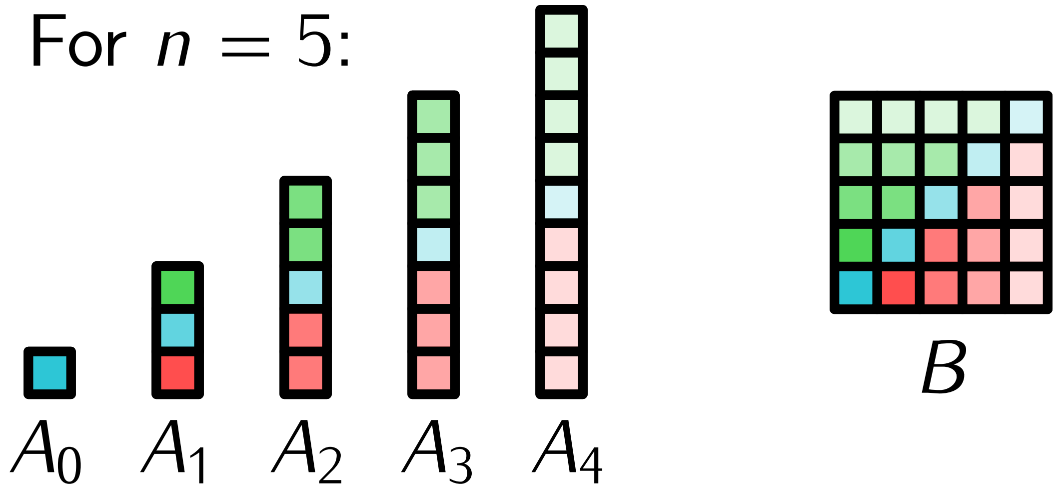
$A = A_1 \cup \cdots \cup A_{n-1}$

$A_k = \{(k,0), \ldots, (k,2k)\}$

A_k has size $2k+1$ and C has size n

direct translation of the equality!

For $n = 5$:



It remains to find the bijection $f\colon A \to B$!

$$f(k,i) = \begin{cases} (k,k) & i = k \\ (k,i) & i < k \\ (i,k) & i > k \end{cases}$$

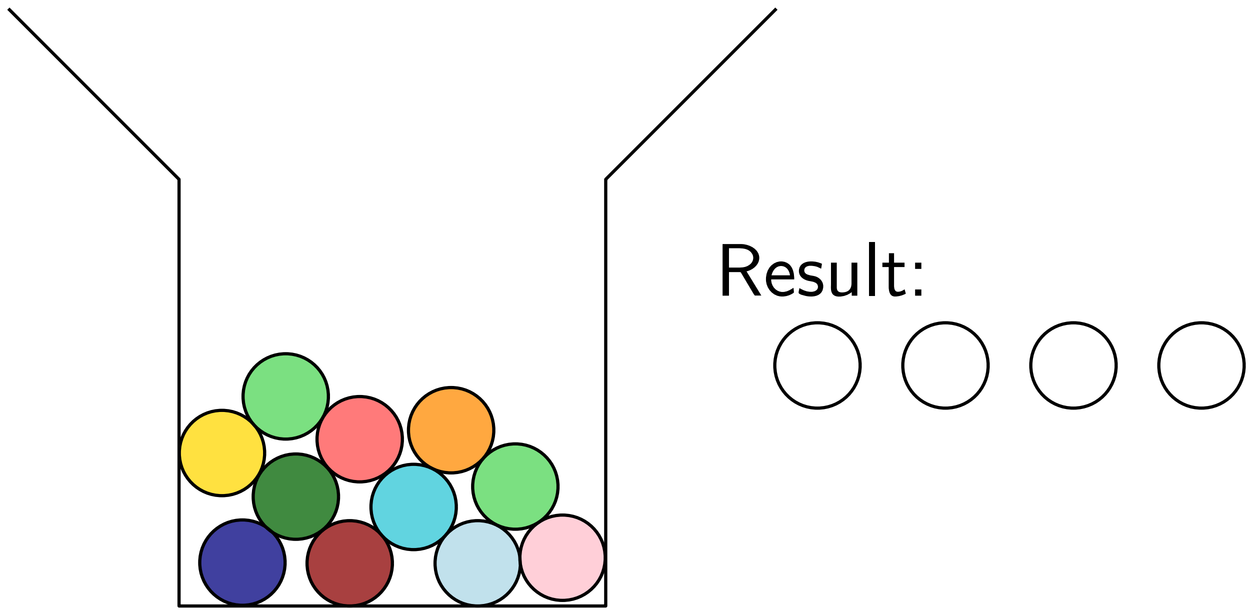
Exercise: prove that this function is a bijection.

The theorem says: there are as many squares on the left as there are on the right

Drawing from a set

Many objects can be seen as the result of the following procedure:

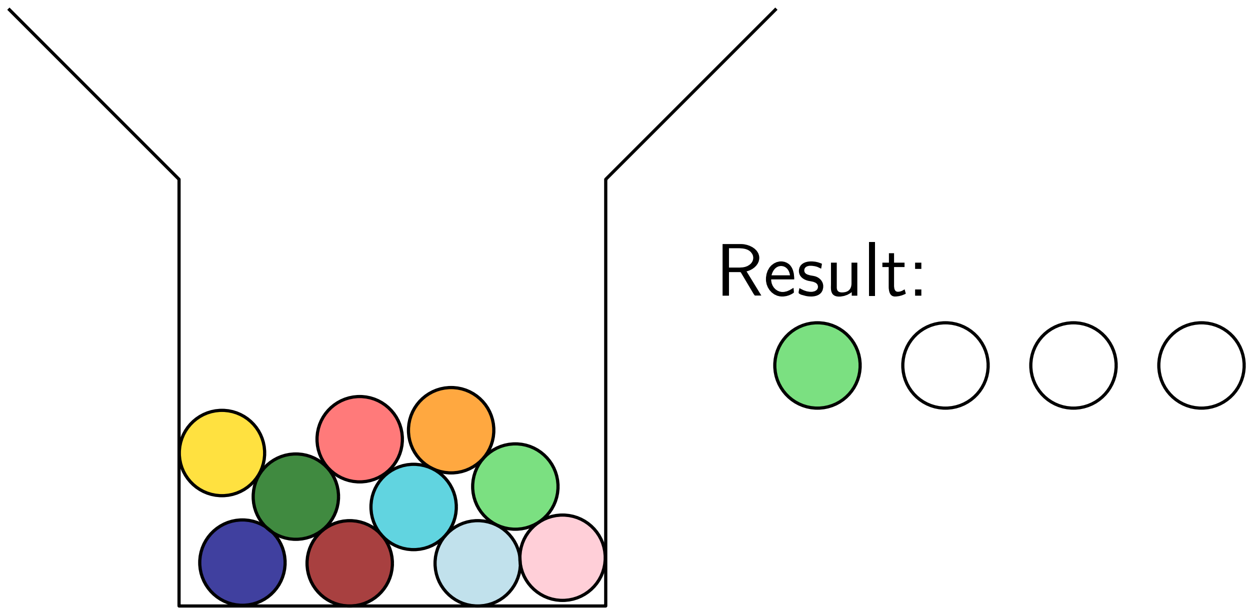
1. We start with a container A that has n elements
2. We extract (= draw) elements $a_1, \ldots, a_k$ from A



Drawing = taking objects from a box

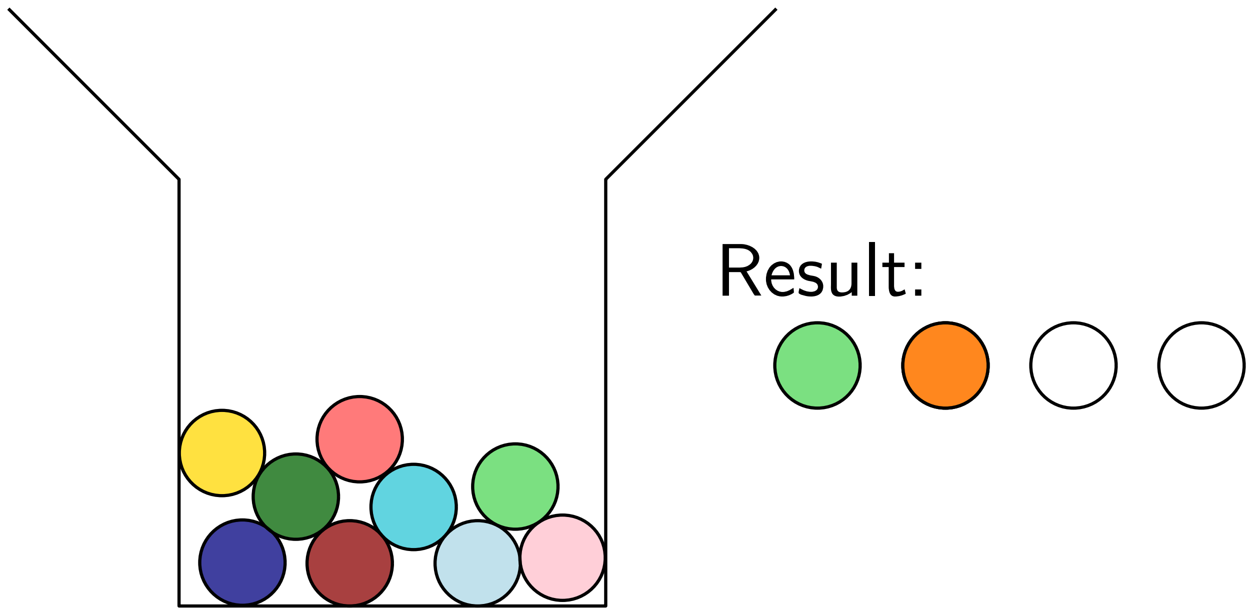
Many objects can be seen as the result of the following procedure:

1. We start with a container A that has n elements
2. We extract (= draw) elements $a_1, \ldots, a_k$ from A



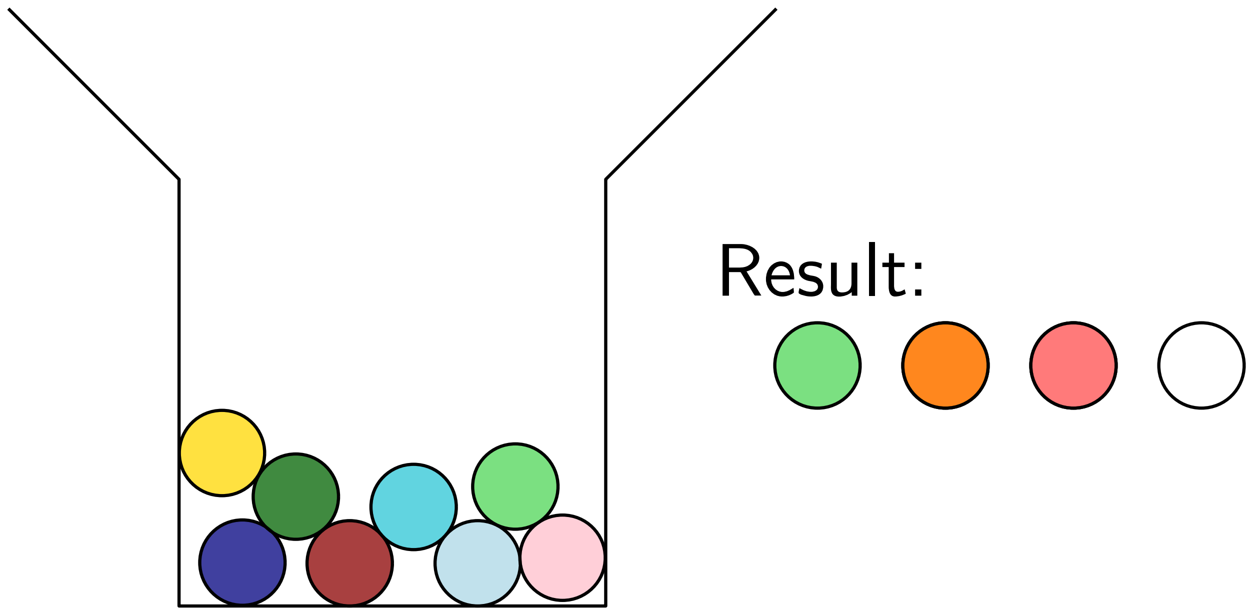
Many objects can be seen as the result of the following procedure:

1. We start with a container A that has n elements
2. We extract (= draw) elements $a_1, \ldots, a_k$ from A



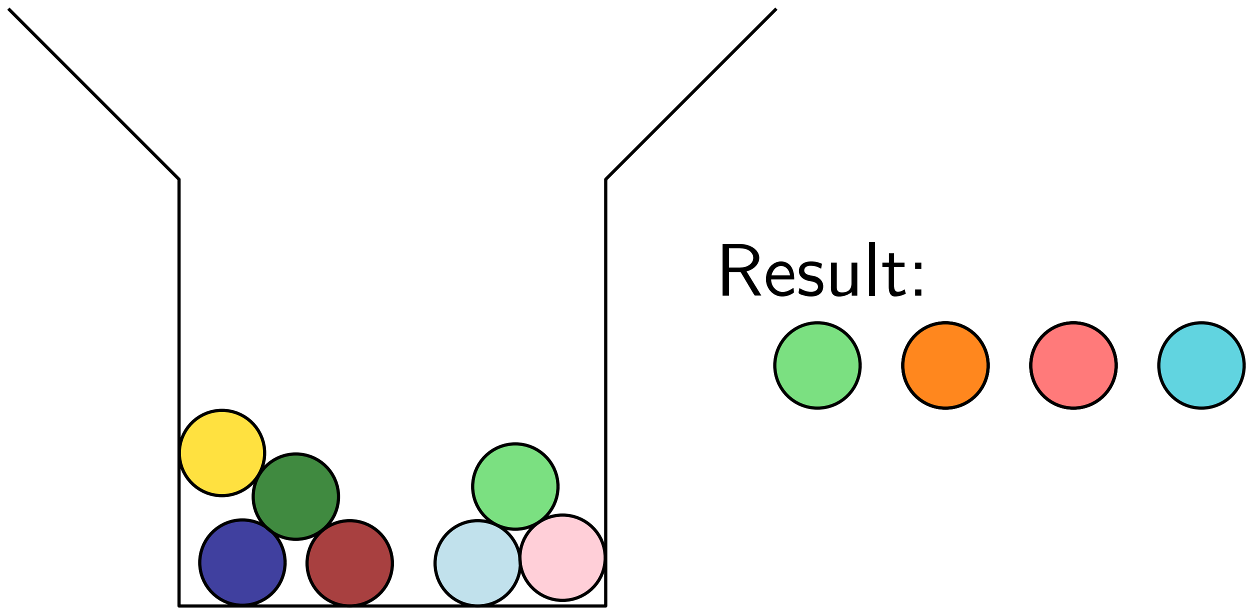
Many objects can be seen as the result of the following procedure:

1. We start with a container A that has n elements
2. We extract (= draw) elements $a_1, \ldots, a_k$ from A



Many objects can be seen as the result of the following procedure:

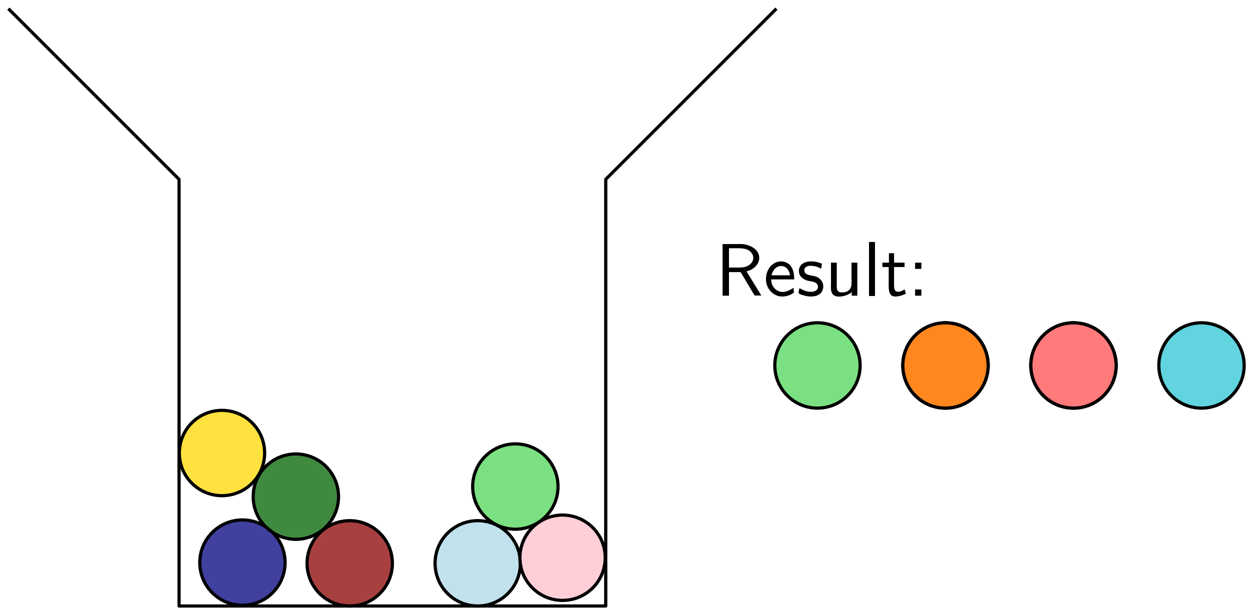
1. We start with a container A that has n elements
2. We extract (= draw) elements $a_1, \ldots, a_k$ from A



Many objects can be seen as the result of the following procedure:

1. We start with a container A that has n elements
2. We extract (= draw) elements $a_1, \ldots, a_k$ from A

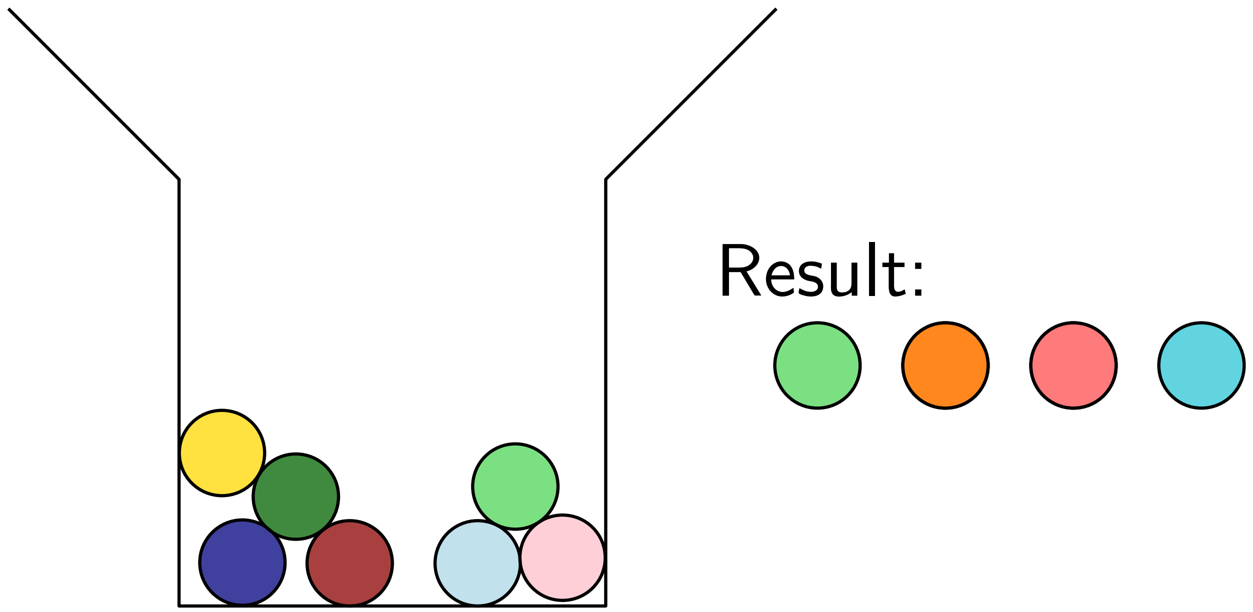
Goal: count how many different possible results there are



Many objects can be seen as the result of the following procedure:

1. We start with a container A that has n elements
2. We extract (= draw) elements $a_1, \ldots, a_k$ from A

Goal: count how many different possible results there are



4 main variations:

- Do we care about the order in which the balls were taken?
- Do we put the ball back in the box after taking it?
  Drawing with or without replacement

Definition. Drawing with order and with replacement:

- A is a set of size n
- Pick an item from A, remember the result and put it back
- Repeat k times
- The result is a tuple of length k

Definition. Drawing with order and with replacement:

- A is a set of size n
- Pick an item from A, remember the result and put it back
- Repeat k times
- The result is a tuple of length k

Set of possible results: A^k

Definition. Drawing with order and with replacement:

- A is a set of size n
- Pick an item from A, remember the result and put it back
- Repeat k times
- The result is a tuple of length k

Set of possible results: A^k, by the product rule: n^k possible results

Definition. Drawing with order and with replacement:

- A is a set of size n
- Pick an item from A, remember the result and put it back
- Repeat k times
- The result is a tuple of length k

```
from itertools import product

for a in product(A, repeat=k):
  # ...
```

Set of possible results: A^k, by the product rule: n^k possible results

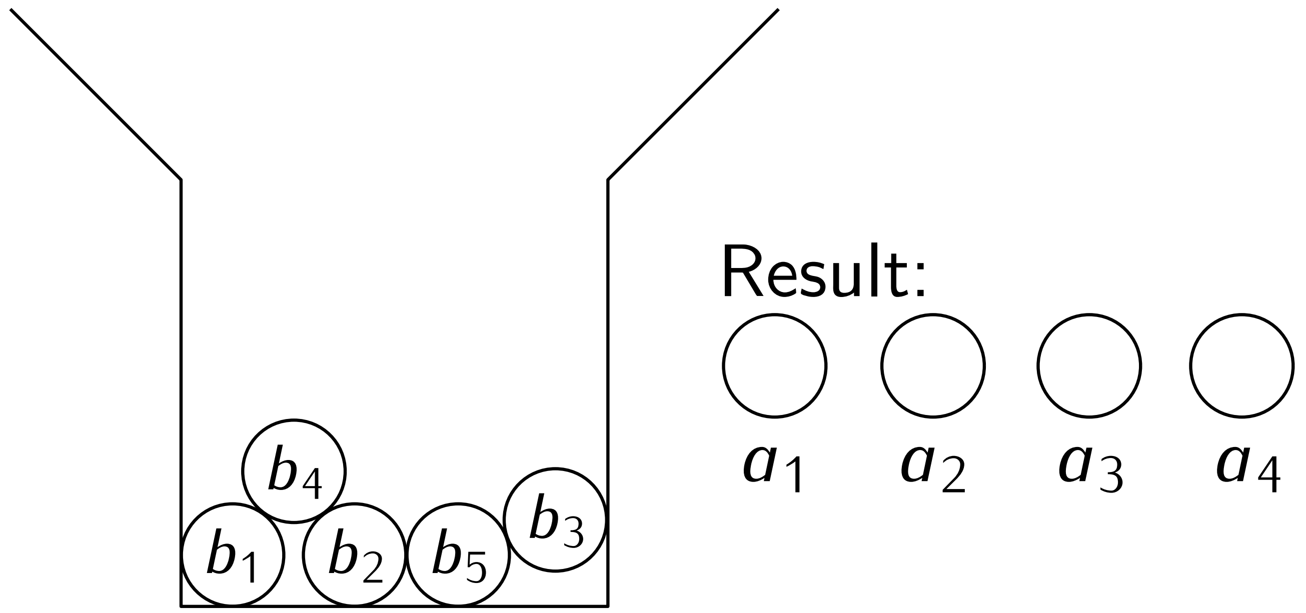
Definition. Drawing with order and with replacement:

- A is a set of size n
- Pick an item from A, remember the result and put it back
- Repeat k times
- The result is a tuple of length k

```
from itertools import product

for a in product(A, repeat=k):
  # ...
```

Set of possible results: A^k, by the product rule: n^k possible results

Application: counting functions $f : A \to B$



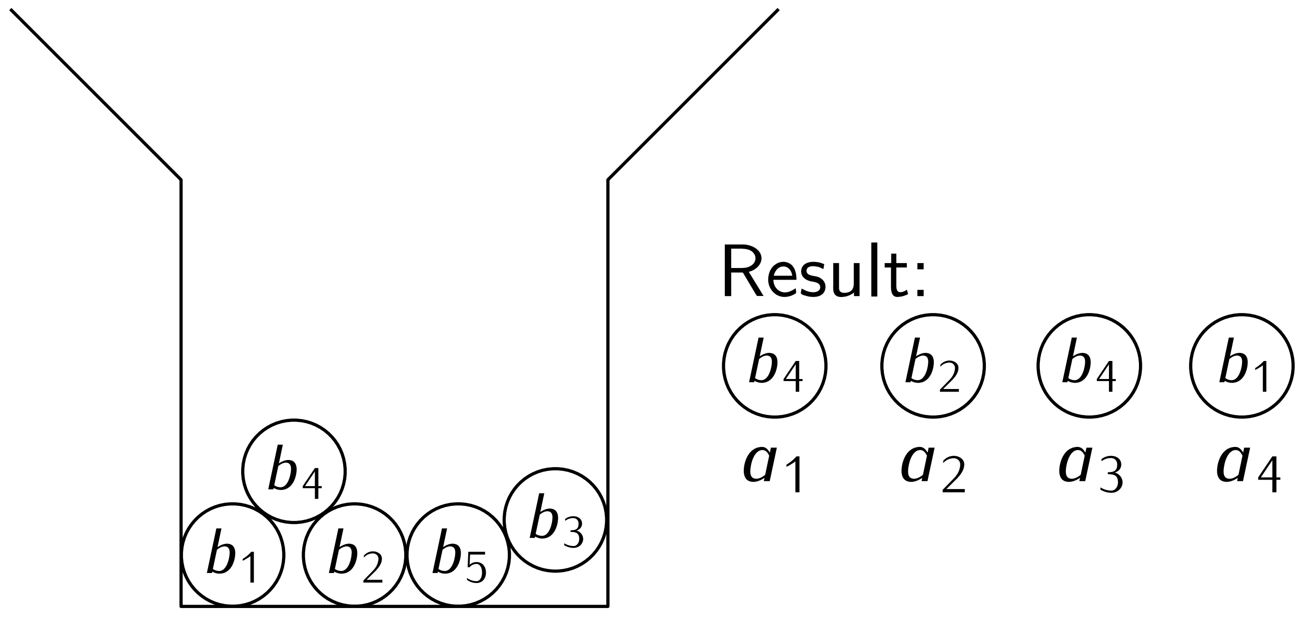
Definition. Drawing with order and with replacement:

- A is a set of size n
- Pick an item from A, remember the result and put it back
- Repeat k times
- The result is a tuple of length k

```
from itertools import product

for a in product(A, repeat=k):
  # ...
```

Set of possible results: A^k, by the product rule: n^k possible results

Application: counting functions $f : A \to B$



Definition. Drawing with order and with replacement:

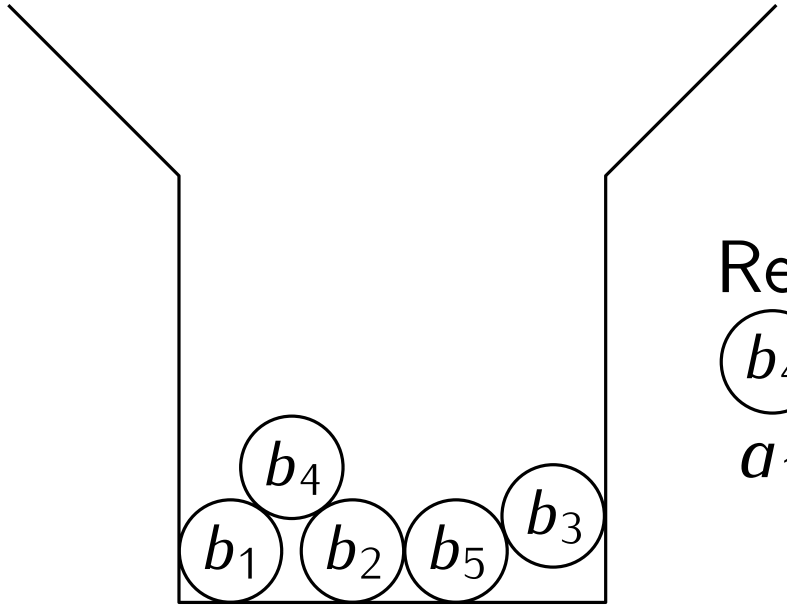
- A is a set of size n
- Pick an item from A, remember the result and put it back
- Repeat k times
- The result is a tuple of length k

```
from itertools import product

for a in product(A, repeat=k):
  # ...
```

Set of possible results: A^k, by the product rule: n^k possible results

Application: counting functions $f : A \to B$

$\Rightarrow |B|^{|A|}$ functions from A to B



Result:

b_4	b_2	b_4	b_1
a_1	a_2	a_3	a_4

Definition. Drawing with order and with replacement:

- A is a set of size n
- Pick an item from A, remember the result and put it back
- Repeat k times
- The result is a tuple of length k

```
from itertools import product

for a in product(A, repeat=k):
  # ...
```

Set of possible results: A^k, by the product rule: n^k possible results

How many binary strings with n bits are there?

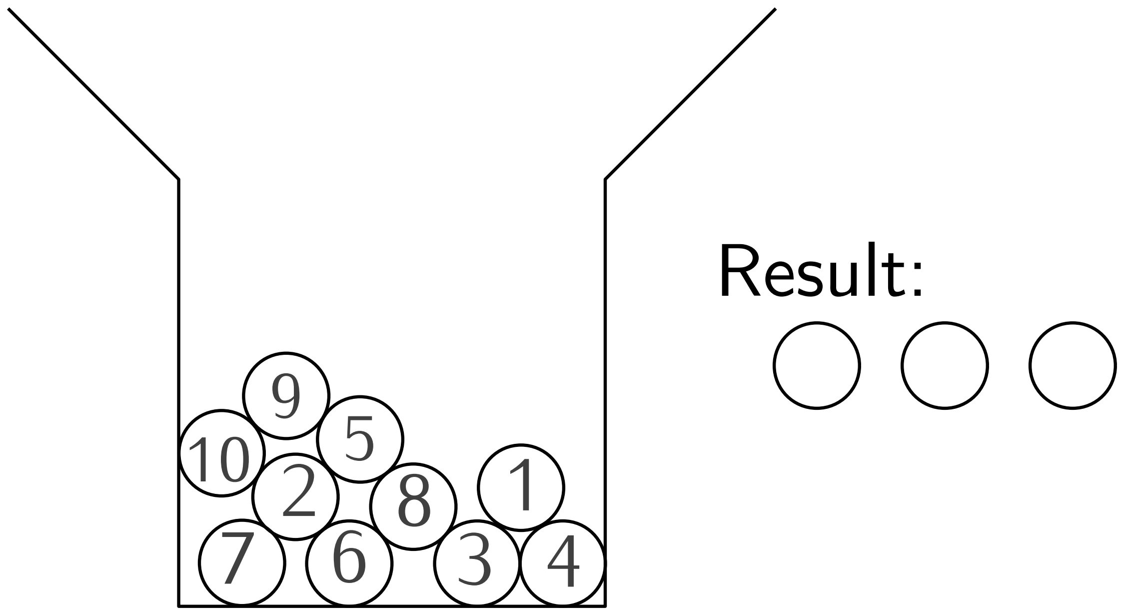
- 2
- $2n$
- n^2
- 2^n

Definition. Drawing with order and without replacement:

- A is a set of size n
- Pick an item from A, remember the result and keep it
- Repeat k times
- The result is a tuple of length k

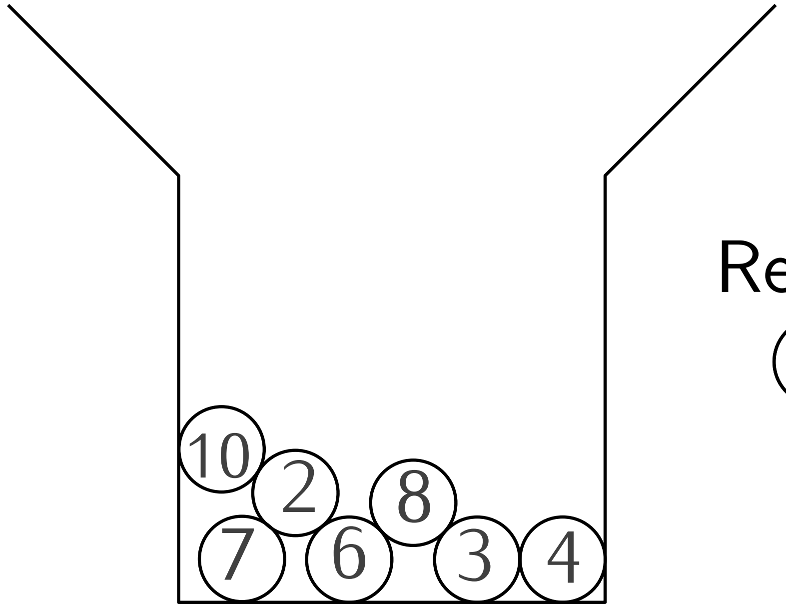
Definition. Drawing with order and without replacement:

- A is a set of size n
- Pick an item from A, remember the result and keep it
- Repeat k times
- The result is a tuple of length k



Definition. Drawing with order and without replacement:

- A is a set of size n
- Pick an item from A, remember the result and keep it
- Repeat k times
- The result is a tuple of length k

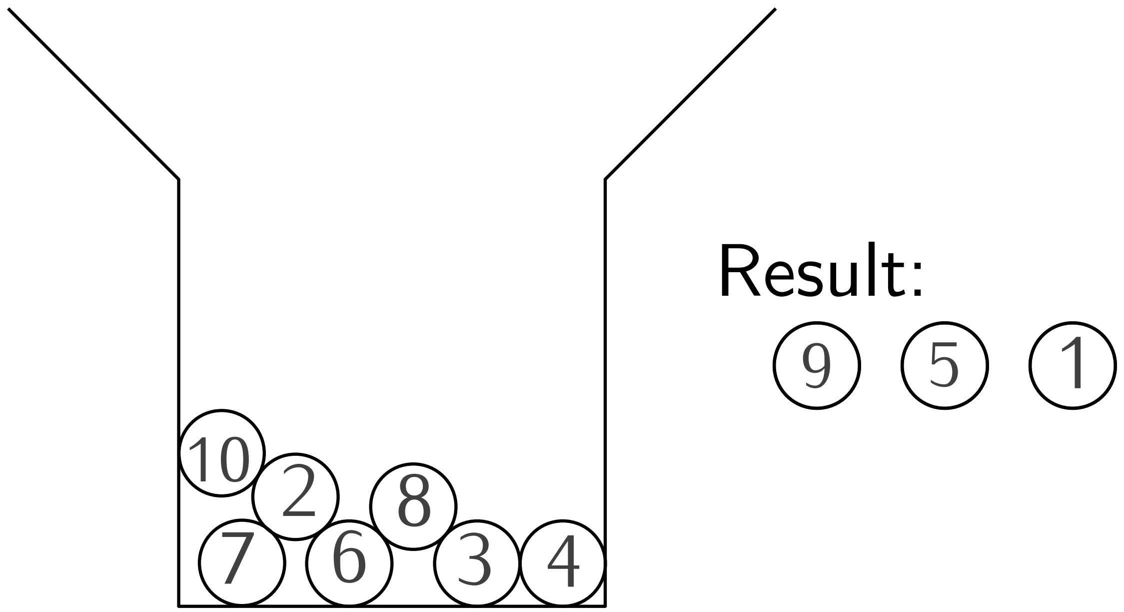


Result:

Definition. Drawing with order and without replacement:

- A is a set of size n
- Pick an item from A, remember the result and keep it
- Repeat k times
- The result is a tuple of length k

Set of possible results: $\{(a_1, a_2, \ldots, a_k) \in A^k \mid \forall i \neq j : a_i \neq a_j\}$

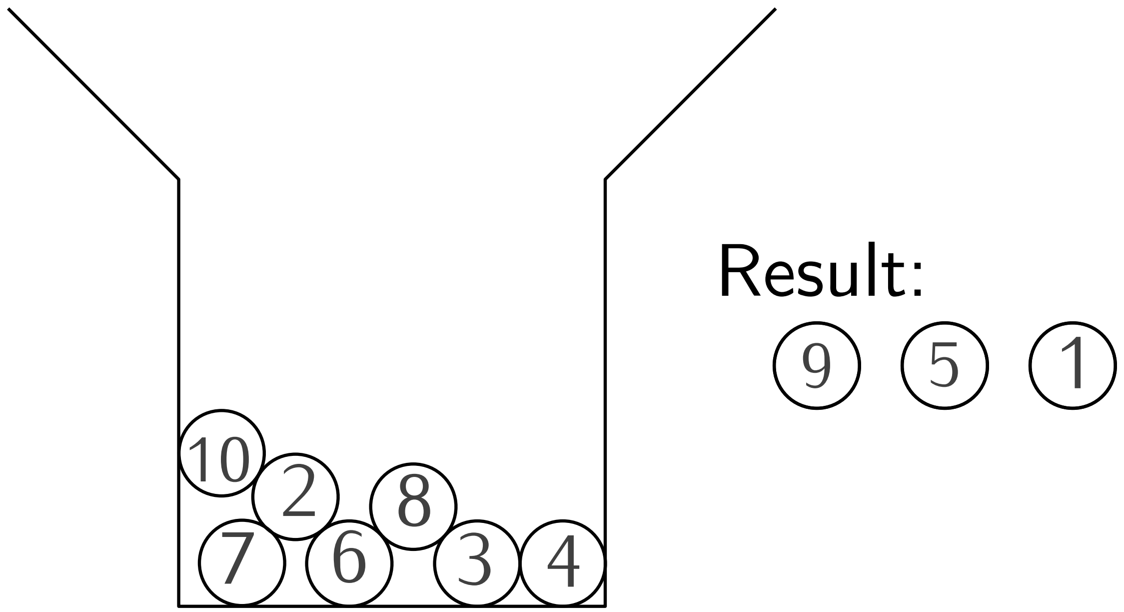


Definition. Drawing with order and without replacement:

- A is a set of size n
- Pick an item from A, remember the result and keep it
- Repeat k times
- The result is a tuple of length k

```
from itertools import permutations

for a in permutations(A,r=k):
  print(a)
```

Set of possible results: $\{(a_1, a_2, \ldots, a_k) \in A^k \mid \forall i \neq j : a_i \neq a_j\}$



Definition. Drawing with order and without replacement:

- A is a set of size n
- Pick an item from A, remember the result and keep it
- Repeat k times
- The result is a tuple of length k

```
from itertools import permutations

for a in permutations(A,r=k):
  print(a)
```

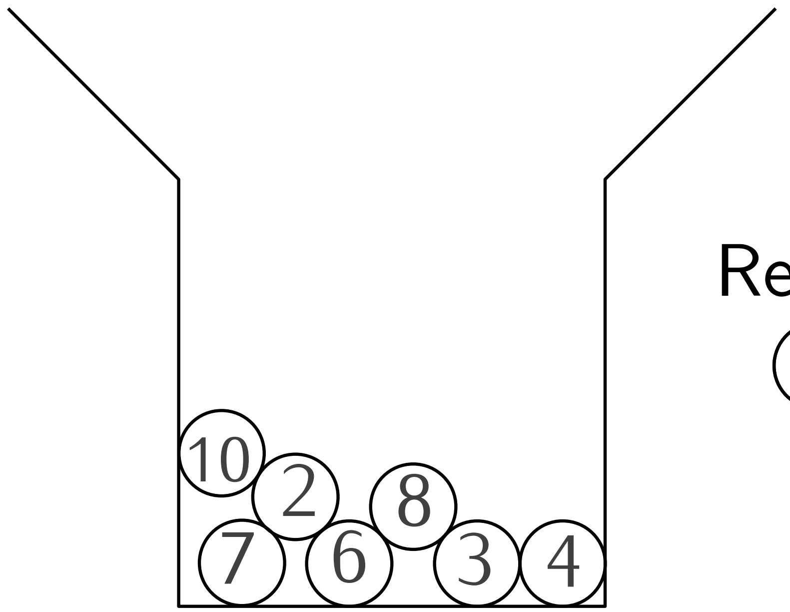
Set of possible results: $\{(a_1, a_2, \ldots, a_k) \in A^k \mid \forall i \neq j : a_i \neq a_j\}$

n choices for a_1

$n-1$ choices for a_2

$\ddots$

$n-k+1$ choices for a_k



Result:

9 5 1

Definition. Drawing with order and without replacement:

- A is a set of size n
- Pick an item from A, remember the result and keep it
- Repeat k times
- The result is a tuple of length k

```python
from itertools import permutations

for a in permutations(A,r=k):
  print(a)
```

Set of possible results: $\{(a_1, a_2, \ldots, a_k) \in A^k \mid \forall i \neq j : a_i \neq a_j\}$

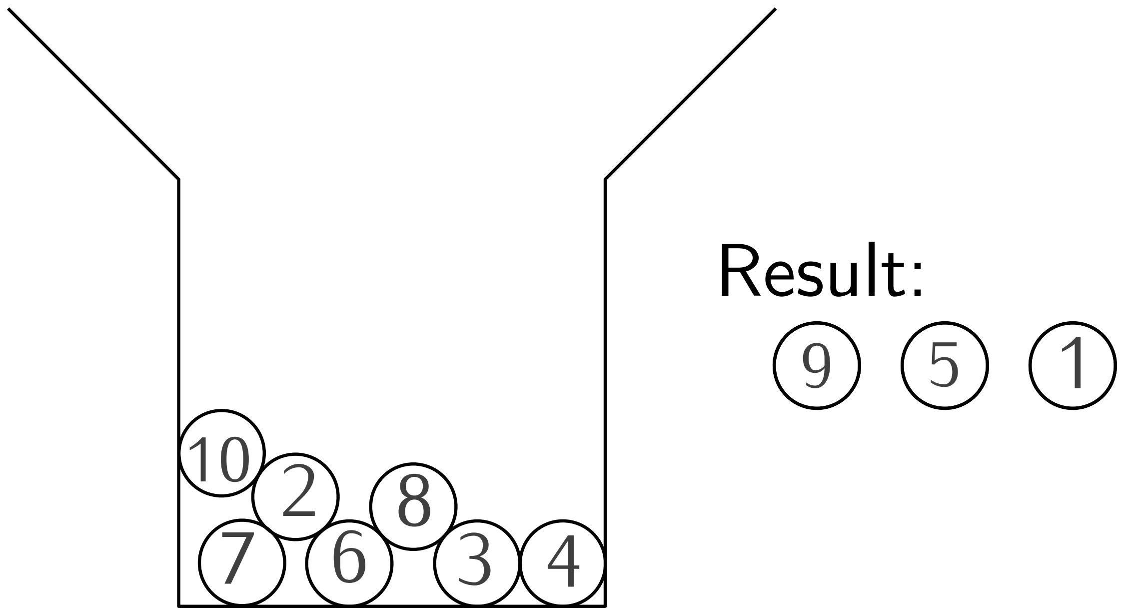
n choices for a_1

$n-1$ choices for a_2

$\ddots$

$n-k+1$ choices for a_k

$\Rightarrow n \times (n-1) \times \cdots \times (n-k+1)$ possible results



Definition. Drawing with order and without replacement:

- A is a set of size n
- Pick an item from A, remember the result and keep it
- Repeat k times
- The result is a tuple of length k

```
from itertools import permutations

for a in permutations(A,r=k):
  print(a)
```

Set of possible results: $\{(a_1, a_2, \ldots, a_k) \in A^k \mid \forall i \neq j : a_i \neq a_j\}$

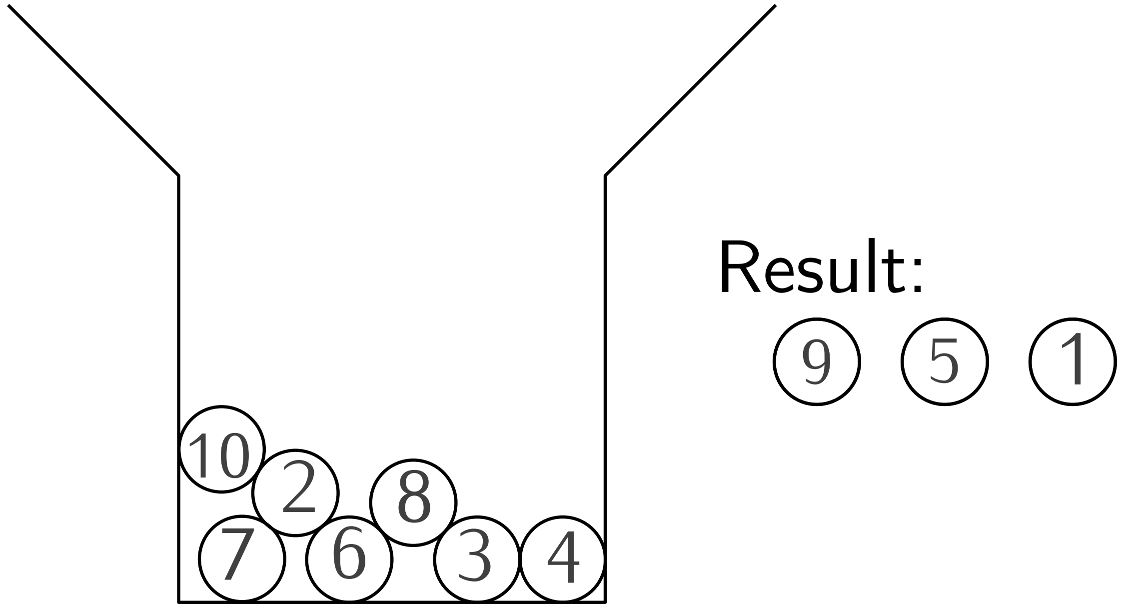
n choices for a_1

$n-1$ choices for a_2

$\ddots$

$n-k+1$ choices for a_k

$\Rightarrow \underbrace{n \times (n-1) \times \cdots \times (n-k+1)}_{n^{\underline{k}}}$ possible results



Definition. Drawing with order and without replacement:

- A is a set of size n
- Pick an item from A, remember the result and keep it
- Repeat k times
- The result is a tuple of length k

$\Rightarrow n \times (n-1) \times \cdots \times (n-k+1)$ possible results

```
from itertools import permutations

for a in permutations(A,r=k):
  print(a)
```

Theorem. If A, B are finite, there are $|B|^{\underline{|A|}}$ injective functions from A to B.

Definition. Drawing with order and without replacement:

- A is a set of size n
- Pick an item from A, remember the result and keep it
- Repeat k times
- The result is a tuple of length k

```
from itertools import permutations

for a in permutations(A,r=k):
  print(a)
```

$\Rightarrow n \times (n-1) \times \cdots \times (n-k+1)$ possible results

Theorem. If A, B are finite, there are $|B|^{\underline{|A|}}$ injective functions from A to B.

How many ways are there to write all numbers $1, 2, \ldots, n$ in an arbitrary order?

- 2
- $n^{\underline{n}}$
- $n^{\underline{2}}$
- n^n

Definition. Drawing with order and without replacement:

- A is a set of size n
- Pick an item from A, remember the result and keep it
- Repeat k times
- The result is a tuple of length k

```
from itertools import permutations

for a in permutations(A,r=k):
  print(a)
```

$\Rightarrow n \times (n-1) \times \cdots \times (n-k+1)$ possible results

Theorem. If A, B are finite, there are $|B|^{\underline{|A|}}$ injective functions from A to B.

How many ways are there to write all numbers $1, 2, \ldots, n$ in an arbitrary order?

- 2
- $n^{\underline{n}}$
- $n^{\underline{2}}$
- n^n

We often write $n!$ instead of $n^{\underline{n}}$, this is the factorial of n.

Definition. Drawing without order and without replacement:

- A is a set of size n
- Pick an item from A and keep it
- Do this k times in total ($k \leq n$ otherwise it is not possible to do this)
- Result is a set of size k (we forget which item was picked when)

Definition. Drawing without order and without replacement:

- A is a set of size n
- Pick an item from A and keep it
- Do this k times in total ($k \leq n$ otherwise it is not possible to do this)
- Result is a set of size k (we forget which item was picked when)

Set of possible results: $\{S \subseteq A \mid |S| = k\}$

Definition. Drawing without order and without replacement:

- A is a set of size n
- Pick an item from A and keep it
- Do this k times in total ($k \leq n$ otherwise it is not possible to do this)
- Result is a set of size k (we forget which item was picked when)

Set of possible results: $\{S \subseteq A \mid |S| = k\} =: \mathcal{P}_k(A)$

Definition. Drawing without order and without replacement:

- A is a set of size n
- Pick an item from A and keep it
- Do this k times in total ($k \leq n$ otherwise it is not possible to do this)
- Result is a set of size k (we forget which item was picked when)

Set of possible results: $\{S \subseteq A \mid |S| = k\} =: \mathcal{P}_k(A)$

$|A|$ ↓

				1				
			1		1			
		1		2		1		
	1		3		3		1	
1		4		6		4		1

Pascal's triangle:
Size of $\mathcal{P}_k(A)$

```
from itertools import combinations

for a in permutations(A,r=k):
  print(set(a))
```

Definition. Drawing without order and without replacement:

- A is a set of size n
- Pick an item from A and keep it
- Do this k times in total ($k \leq n$ otherwise it is not possible to do this)
- Result is a set of size k (we forget which item was picked when)

Set of possible results: $\{S \subseteq A \mid |S| = k\} =: \mathcal{P}_k(A)$

$$
\begin{array}{ccccccccc}
 & & & & 1 & & & & \\
 & & & 1 & & 1 & & & \\
 & & 1 & & 2 & & 1 & & \\
 & 1 & & 3 & & 3 & & 1 & \\
1 & & 4 & & 6 & & 4 & & 1
\end{array}
\quad \Big\downarrow |A|
$$

Pascal's triangle:
Size of $\mathcal{P}_k(A)$

Notation. We write $\binom{n}{k}$ for the size of $\mathcal{P}_k(A)$, where $n = |A|$.

```
from itertools import combinations

for a in permutations(A,r=k):
  print(set(a))
```

Definition. Drawing without order and without replacement:

- A is a set of size n
- Pick an item from A and keep it
- Do this k times in total ($k \leq n$ otherwise it is not possible to do this)
- Result is a set of size k (we forget which item was picked when)

Set of possible results: $\{S \subseteq A \mid |S| = k\} =: \mathcal{P}_k(A)$

$$\begin{array}{ccccccccc} & & & & 1 & & & & \\ & & & 1 & & 1 & & & \\ & & 1 & & 2 & & 1 & & \\ & 1 & & 3 & & 3 & & 1 & \\ 1 & & 4 & & 6 & & 4 & & 1 \end{array}$$

$|A| \downarrow$

Pascal's triangle:
Size of $\mathcal{P}_k(A)$

Notation. We write $\binom{n}{k}$ for the size of $\mathcal{P}_k(A)$, where $n = |A|$.

Theorem. For all n and $k \in \{0, \ldots, n\}$: $\binom{n}{k} = \frac{n^k}{k!}$

```
from itertools import combinations

for a in permutations(A,r=k):
  print(set(a))
```

Definition. Drawing without order and without replacement:

- A is a set of size n
- Pick an item from A and keep it
- Do this k times in total ($k \leq n$ otherwise it is not possible to do this)
- Result is a set of size k (we forget which item was picked when)

Set of possible results: $\{S \subseteq A \mid |S| = k\} =: \mathcal{P}_k(A)$

$$\begin{array}{c} 1 \\ 1 \quad 2 \quad 1 \\ 1 \quad 3 \quad 3 \quad 1 \\ 1 \quad 4 \quad 6 \quad 4 \quad 1 \end{array}$$

$|A| \downarrow$

Pascal's triangle:
Size of $\mathcal{P}_k(A)$

Notation. We write $\binom{n}{k}$ for the size of $\mathcal{P}_k(A)$, where $n = |A|$.

```
from itertools import combinations

for a in permutations(A,r=k):
  print(set(a))
```

Theorem. For all n and $k \in \{0, \ldots, n\}$: $\binom{n}{k} = \frac{\cancel{n^k}}{k!}$

$$k! \times \binom{n}{k} = \cancel{n^k}$$

Definition. Drawing without order and without replacement:

- A is a set of size n
- Pick an item from A and keep it
- Do this k times in total ($k \leq n$ otherwise it is not possible to do this)
- Result is a set of size k (we forget which item was picked when)

Set of possible results: $\{S \subseteq A \mid |S| = k\} =: \mathcal{P}_k(A)$

$$
\begin{array}{ccccccccc}
 & & & & 1 & & & & \\
 & & & 1 & & 1 & & & \\
 & & 1 & & 2 & & 1 & & \\
 & 1 & & 3 & & 3 & & 1 & \\
1 & & 4 & & 6 & & 4 & & 1
\end{array}
$$

$|A|$

Pascal's triangle:
Size of $\mathcal{P}_k(A)$

Notation. We write $\binom{n}{k}$ for the size of $\mathcal{P}_k(A)$, where $n = |A|$.

```
from itertools import combinations

for a in permutations(A,r=k):
  print(set(a))
```

Theorem. For all n and $k \in \{0, \dots, n\}$: $\binom{n}{k} = \frac{n^k}{k!}$

$$k! \times \binom{n}{k} = n^k$$

number of permutations of a set with k elements

number of subsets with k elements

number of tuples of length k

Drawing a set without replacement

Definition. Drawing without order and without replacement:

- A is a set of size n
- Pick an item from A and keep it
- Do this k times in total ($k \leq n$ otherwise it is not possible to do this)
- Result is a set of size k (we forget which item was picked when)

Set of possible results: $\{S \subseteq A \mid |S| = k\} =: \mathcal{P}_k(A)$

$$
\begin{array}{ccccccccc}
 & & & & 1 & & & & \\
 & & & 1 & & 1 & & & \\
 & & 1 & & 2 & & 1 & & \\
 & 1 & & 3 & & 3 & & 1 & \\
1 & & 4 & & 6 & & 4 & & 1
\end{array}
$$

$|A| \downarrow$

Pascal's triangle:
Size of $\mathcal{P}_k(A)$

Notation. We write $\binom{n}{k}$ for the size of $\mathcal{P}_k(A)$, where $n = |A|$.

```
from itertools import combinations

for a in permutations(A,r=k):
  print(set(a))
```

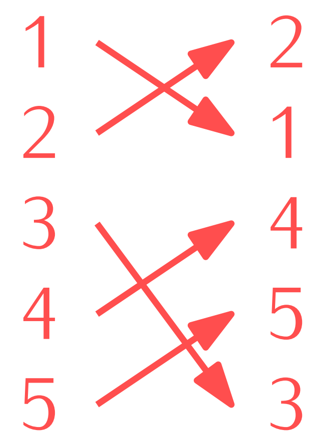
Theorem. For all n and $k \in \{0, \ldots, n\}$: $\binom{n}{k} = \frac{n^k}{k!}$

$$k! \times \binom{n}{k} = n^{\underline{k}}$$

number of permutations of a set with k elements

number of subsets with k elements

number of tuples of length k



Definition. Drawing without order and without replacement:

- A is a set of size n
- Pick an item from A and keep it
- Do this k times in total ($k \leq n$ otherwise it is not possible to do this)
- Result is a set of size k (we forget which item was picked when)

Set of possible results: $\{S \subseteq A \mid |S| = k\} =: \mathcal{P}_k(A)$

$$
\begin{array}{ccccccccc}
 & & & & 1 & & & & \\
 & & & 1 & & 1 & & & \\
 & & 1 & & 2 & & 1 & & \\
 & 1 & & 3 & & 3 & & 1 & \\
1 & & 4 & & 6 & & 4 & & 1
\end{array}
$$

$|A| \downarrow$

Pascal's triangle:
Size of $\mathcal{P}_k(A)$

Notation. We write $\binom{n}{k}$ for the size of $\mathcal{P}_k(A)$, where $n = |A|$.

Theorem. For all n and $k \in \{0, \dots, n\}$: $\binom{n}{k} = \frac{n^k}{k!}$

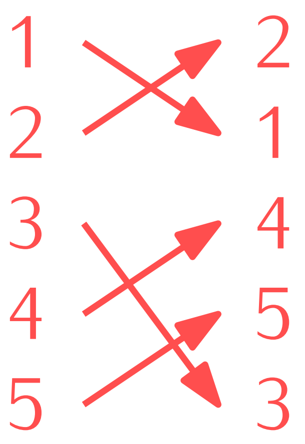
```
from itertools import combinations

for a in permutations(A,r=k):
  print(set(a))
```

$$k! \times \binom{n}{k} = n^k$$

number of permutations of a set with k elements

number of subsets with k elements

number of tuples of length k



$\{1, 3, 4, 7, 9\}$

Definition. Drawing without order and without replacement:

- A is a set of size n
- Pick an item from A and keep it
- Do this k times in total ($k \leq n$ otherwise it is not possible to do this)
- Result is a set of size k (we forget which item was picked when)

Set of possible results: $\{S \subseteq A \mid |S| = k\} =: \mathcal{P}_k(A)$

$$\begin{array}{ccccccccc} & & & & 1 & & & & \\ & & & 1 & & 1 & & & \\ & & 1 & & 2 & & 1 & & \\ & 1 & & 3 & & 3 & & 1 & \\ 1 & & 4 & & 6 & & 4 & & 1 \end{array}$$

$|A|$

Pascal's triangle:
Size of $\mathcal{P}_k(A)$

Notation. We write $\binom{n}{k}$ for the size of $\mathcal{P}_k(A)$, where $n = |A|$.

Theorem. For all n and $k \in \{0, \ldots, n\}$: $\binom{n}{k} = \frac{n^k}{k!}$

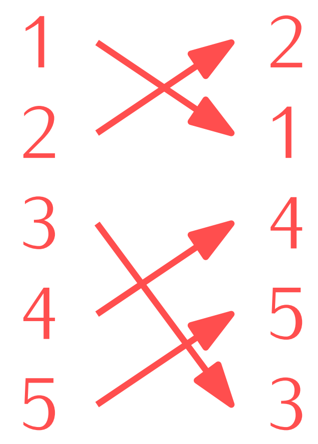
```
from itertools import combinations

for a in permutations(A,r=k):
  print(set(a))
```

$$k! \times \binom{n}{k} = n^k$$

number of permutations of a set with k elements

number of subsets with k elements

number of tuples of length k



$\{1,3,4,7,9\} \mapsto (3,1,7,9,3)$

Definition. Drawing without order and without replacement:
- A is a set of size n
- Pick an item from A and keep it
- Do this k times in total ($k \leq n$ otherwise it is not possible to do this)
- Result is a set of size k (we forget which item was picked when)

Set of possible results: $\{S \subseteq A \mid |S| = k\} =: \mathcal{P}_k(A)$

$|A|$

				1				
		1		2		1		
	1		3		3		1	
1		4		6		4		1

Pascal's triangle:
Size of $\mathcal{P}_k(A)$

Notation. We write $\binom{n}{k}$ for the size of $\mathcal{P}_k(A)$, where $n = |A|$.

Theorem. For all n and $k \in \{0, \ldots, n\}$: $\binom{n}{k} = \frac{n^k}{k!}$

How many binary strings of length 2 are there that have exactly 2 times the bit 1?
- 2
- 10
- 20
- 30
- 40

Theorem. If A is finite, then $|\mathcal{P}(A)| = 2^{|A|}$.

Theorem. If A is finite, then $|\mathcal{P}(A)| = 2^{|A|}$.

subsets of A

binary strings with exactly $n = |A|$ bits

Theorem. If A is finite, then $|\mathcal{P}(A)| = 2^{|A|}$.

subsets of A

binary strings with exactly $n = |A|$ bits

Define $f \colon \mathcal{P}(A) \to \{0,1\}^n$

$$S \mapsto (b_1, \ldots, b_i, \ldots, b_n)$$

1 if $i \in S$, 0 otherwise

Theorem. If A is finite, then $|\mathcal{P}(A)| = 2^{|A|}$.

subsets of A

binary strings with exactly $n = |A|$ bits

Define $f : \mathcal{P}(A) \to \{0,1\}^n$

$$S \mapsto (b_1, \ldots, b_i, \ldots, b_n)$$

1 if $i \in S$, 0 otherwise

Theorem. For all $n \geq 0$, $2^n = \sum_{k=0}^{n} \binom{n}{k}$.

Theorem. If A is finite, then $|\mathcal{P}(A)| = 2^{|A|}$.

subsets of A

binary strings with exactly $n = |A|$ bits

Define $f \colon \mathcal{P}(A) \to \{0,1\}^n$

$$S \mapsto (b_1, \dots, b_i, \dots, b_n)$$

1 if $i \in S$, 0 otherwise

Theorem. For all $n \geq 0$, $2^n = \sum_{k=0}^{n} \binom{n}{k}$.

size of $\mathcal{P}(A)$, for any set A of size n

partition $\mathcal{P}(A)$ into $B_0, \dots, B_n$

B_k has size $\binom{n}{k}$

Theorem. If A is finite, then $|\mathcal{P}(A)| = 2^{|A|}$.

subsets of A

binary strings with exactly $n = |A|$ bits

Define $f \colon \mathcal{P}(A) \to \{0,1\}^n$

$$S \mapsto (b_1, \ldots, b_i, \ldots, b_n)$$

1 if $i \in S$, 0 otherwise

Theorem. For all $n \geq 0$, $2^n = \sum_{k=0}^{n} \binom{n}{k}$.

size of $\mathcal{P}(A)$, for any set A of size n

partition $\mathcal{P}(A)$ into $B_0, \ldots, B_n$

B_k has size $\binom{n}{k}$

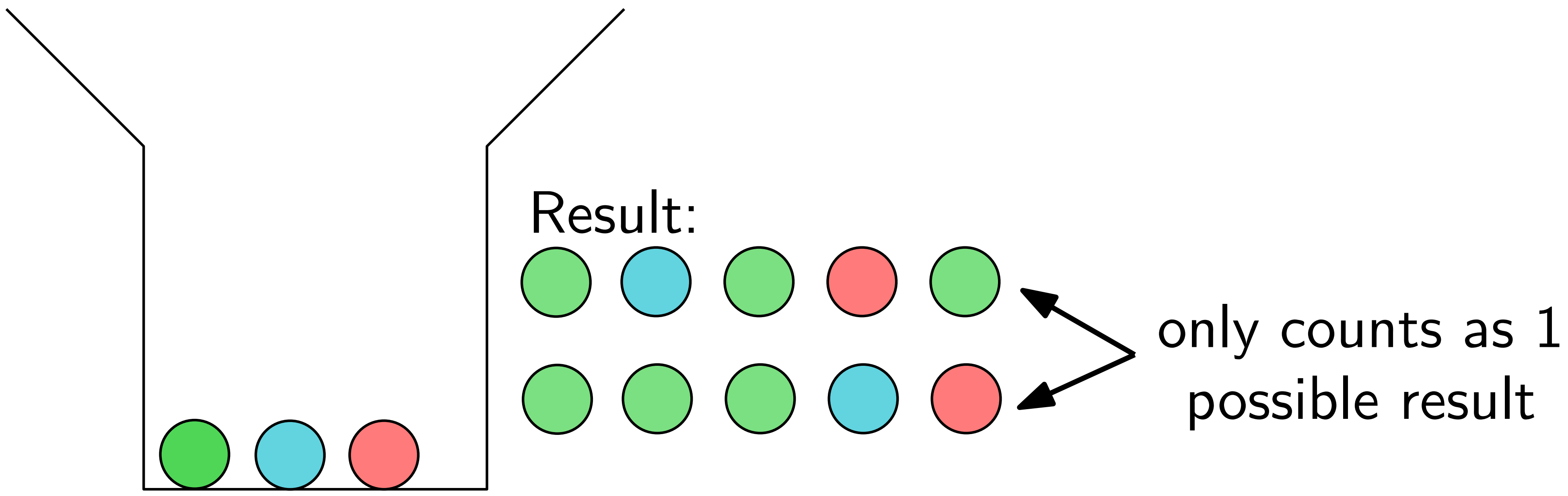
Proof. By the union rule: $\mathcal{P}(A) = \bigcup_{k=0}^{n} \mathcal{P}_k(A)$ and $\mathcal{P}_k(A) \cap \mathcal{P}_\ell(A) = \emptyset$ when $k \neq \ell$. $\square$

Definition. Drawing without order and with replacement:

- A is a set of size n
- Pick an item from A and put it back
- Do this k times in total
- Result is a ??? of size k (we forget which item was picked when)

Definition. Drawing without order and with replacement:

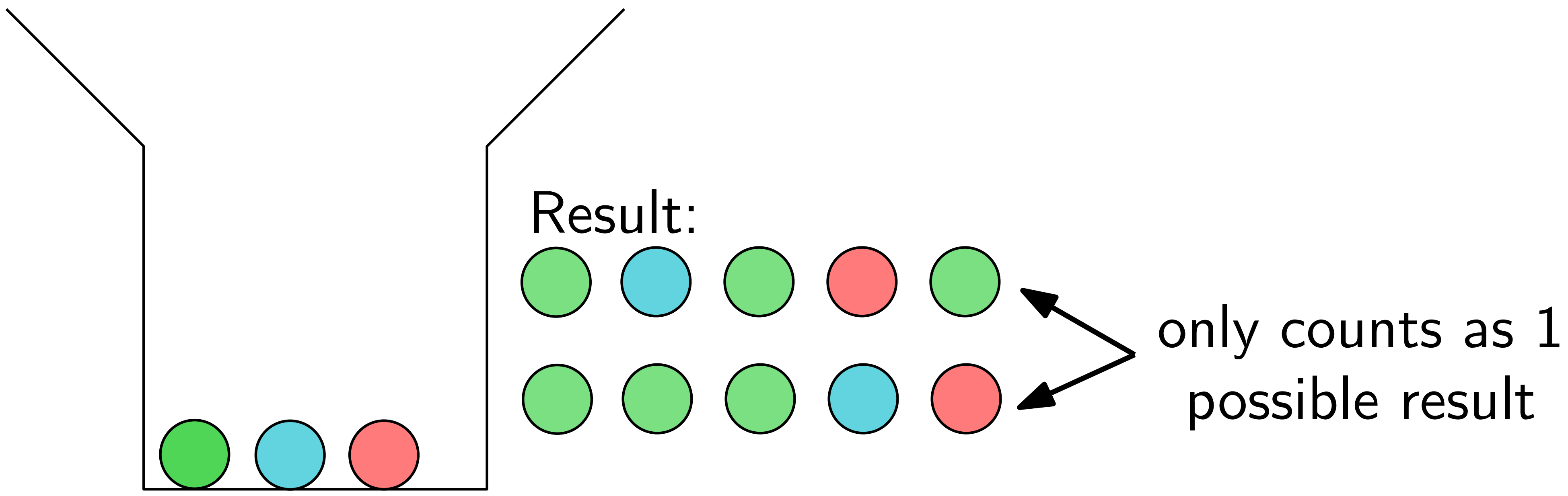
- A is a set of size n
- Pick an item from A and put it back
- Do this k times in total
- Result is a ??? of size k (we forget which item was picked when)



Definition. Drawing without order and with replacement:

- A is a set of size n
- Pick an item from A and put it back
- Do this k times in total
- Result is a multiset of size k (we forget which item was picked when)

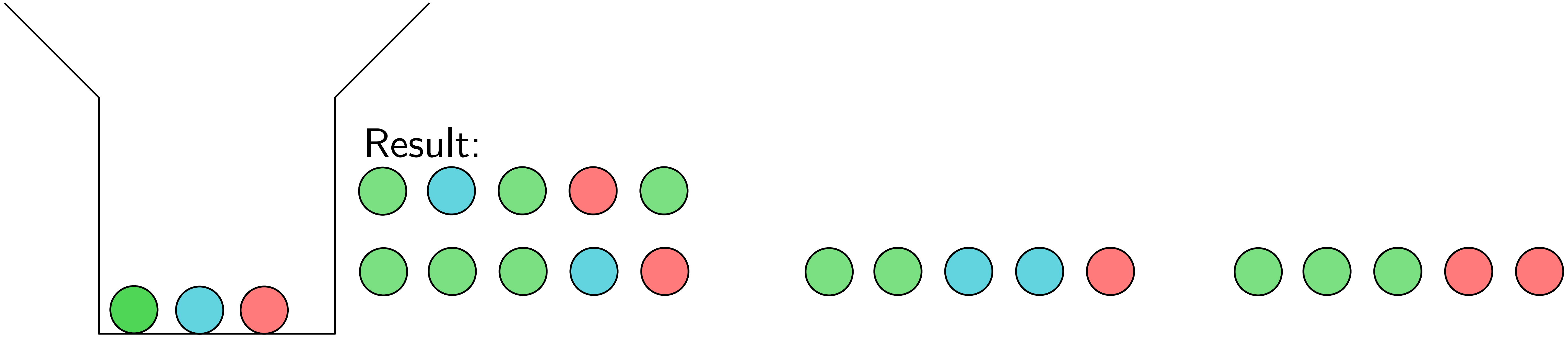
A multiset is a “set” that can contain an object several times. We use $\{\{\dots\}\}$ to write multisets.



Definition. Drawing without order and with replacement:

- A is a set of size n
- Pick an item from A and put it back
- Do this k times in total
- Result is a multiset of size k (we forget which item was picked when)

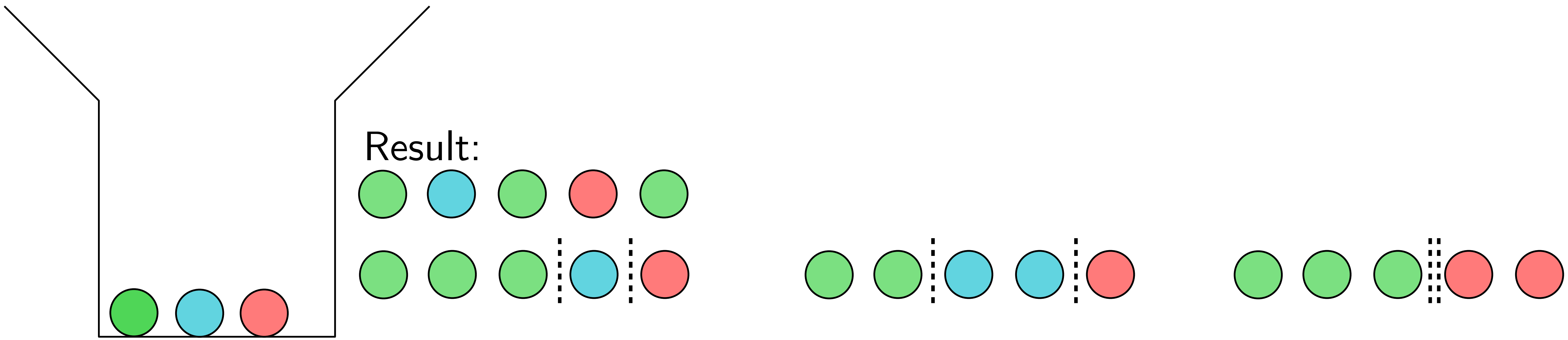
A multiset is a "set" that can contain an object several times. We use $\{\{\ldots\}\}$ to write multisets.



Definition. Drawing without order and with replacement:

- A is a set of size n
- Pick an item from A and put it back
- Do this k times in total
- Result is a multiset of size k (we forget which item was picked when)

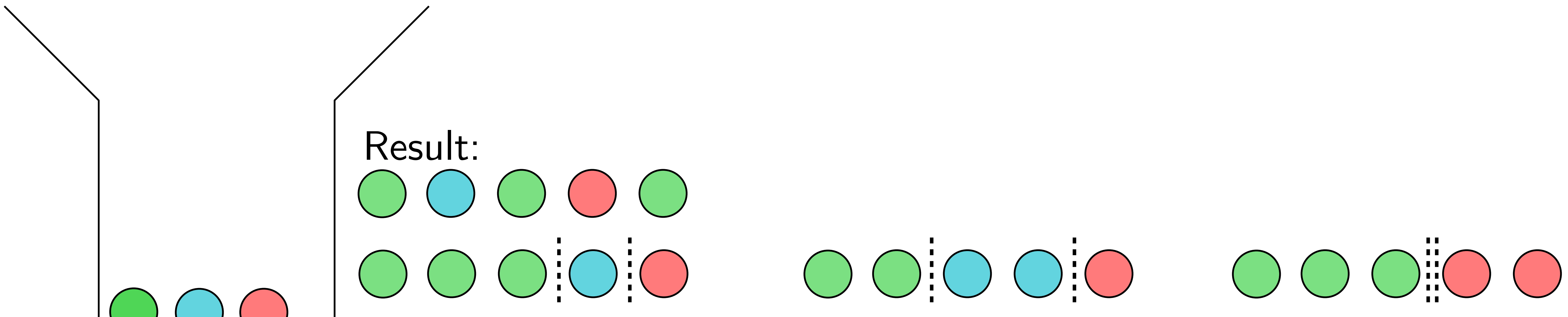
A multiset is a "set" that can contain an object several times. We use $\{\{\ldots\}\}$ to write multisets.



Definition. Drawing without order and with replacement:

- A is a set of size n
- Pick an item from A and put it back
- Do this k times in total
- Result is a multiset of size k (we forget which item was picked when)

A multiset is a "set" that can contain an object several times. We use $\{\{\ldots\}\}$ to write multisets.

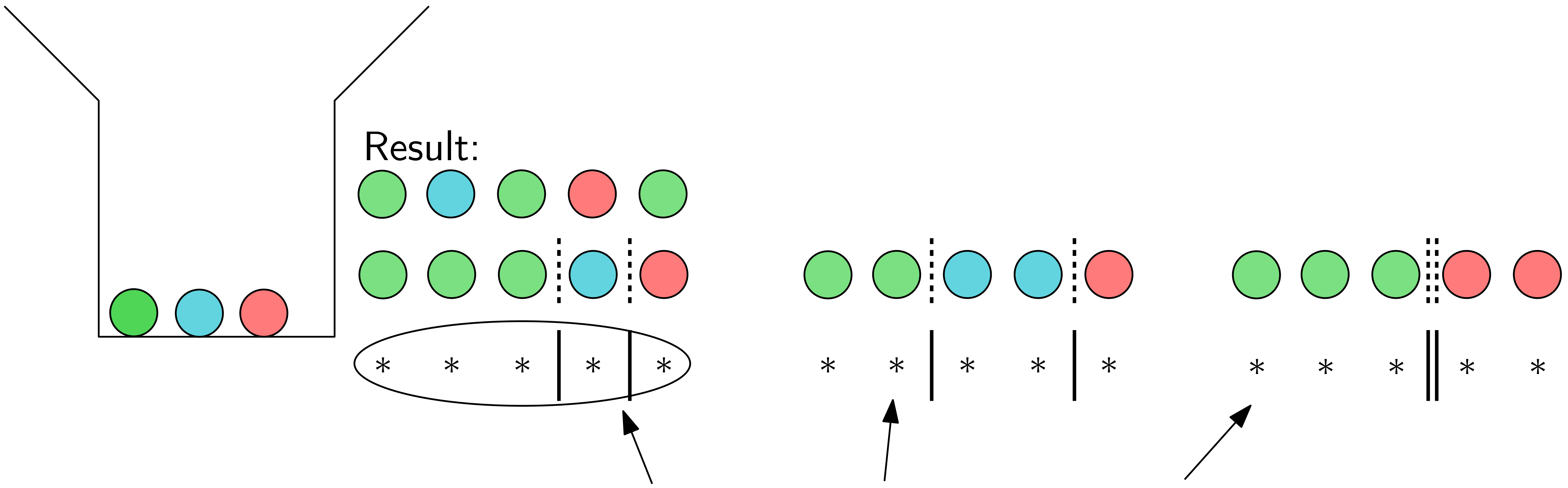


$* \; * \; * \mid * \mid *$ $\qquad$ $* \; * \mid * \; * \mid *$ $\qquad$ $* \; * \; * \mid\mid * \; *$

Definition. Drawing without order and with replacement:

- A is a set of size n
- Pick an item from A and put it back
- Do this k times in total
- Result is a multiset of size k (we forget which item was picked when)

A multiset is a "set" that can contain an object several times. We use $\{\{\ldots\}\}$ to write multisets.



strings with symbols from the set $\{|, *\}$ and exactly bars

- Countable sets, uncountable sets
- Countable sets = what can be represented exactly on a computer
- Combinatorial proofs as a way to prove equalities/inequalities about numbers using functions

$$\begin{aligned} \text{Injection } A \to B &\quad\leftrightarrow\quad |A| \leq |B| \\ \text{Surjection } A \to B &\quad\leftrightarrow\quad |A| \geq |B| \\ \text{Bijection } A \to B &\quad\leftrightarrow\quad |A| = |B| \end{aligned}$$

- Drawing a tuple/(multi)set with/without replacement

  n = size of the set we are drawing from

  k = number of draws

	Order matters	Order does not matter
Replacement	n^k	$\frac{n!}{k!(n-k)!}$
No replacement	$n^{\underline{k}}$	$\binom{n+k-1}{n-1}$