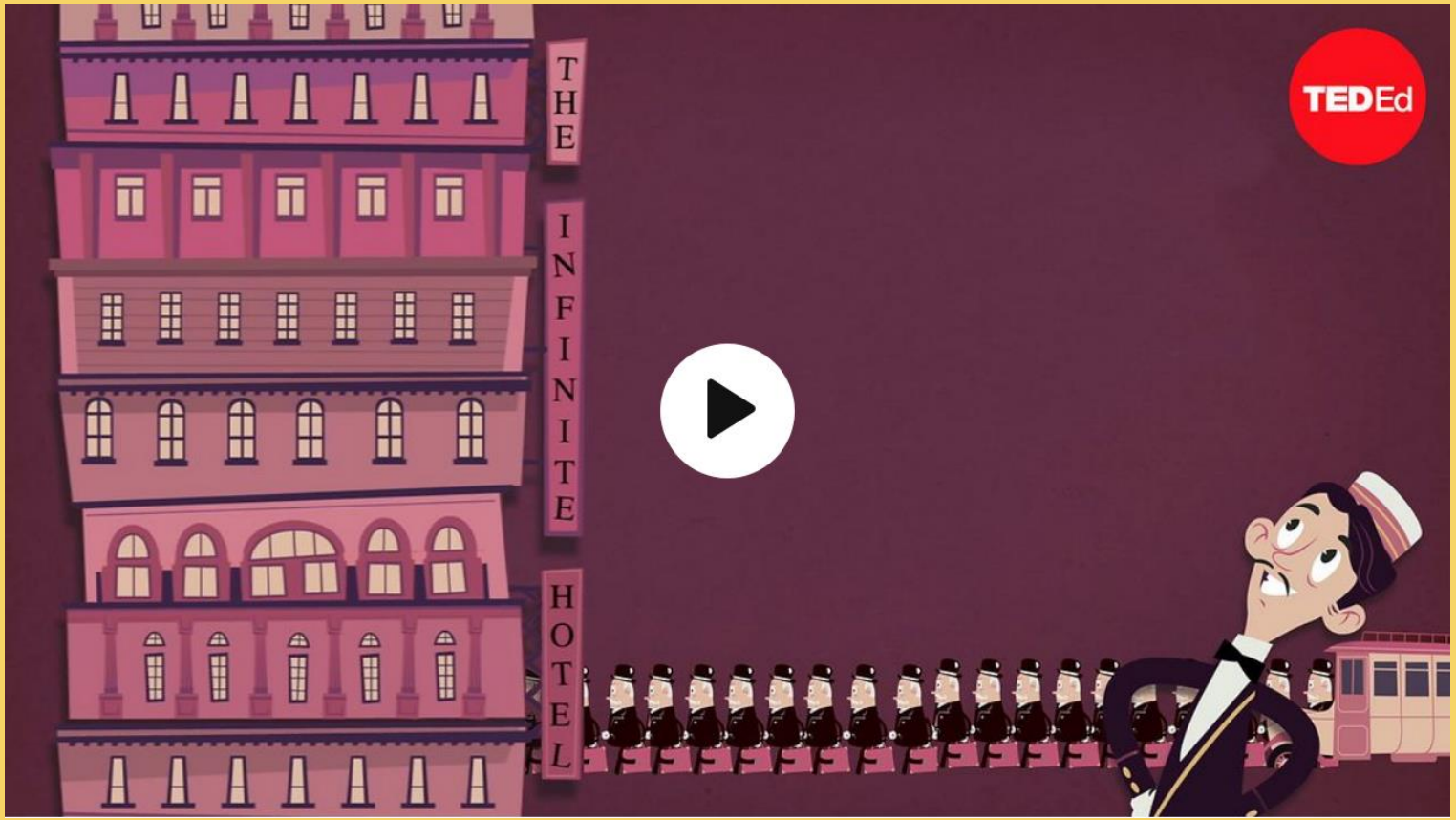
Problem Set 1 is due
This Friday, Jan 24 (10pm)



https://www.ted.com/talks/jeff_dekofsky_the_infinite_hotel_paradox

Class 3: *What Can Be Represented by Bits?*

University of Virginia
cs3120: DMT2
Wei-Kai Lin

What can we do to best prepare and learn for the material in this class?

Only this lecture:
question / answer for bonus

Rule:

- Each question or answer is **+0.5%** to semester
- Once per person
- Raise hand, say your name
- To record, submit on Ed Discuss after class

Recap from Class 2

Rule 1: $\mathbf{0}$ is a Natural Number

Rule 2: If n is a Natural Number, $\boldsymbol{S}(n)$ is a Natural Number.

Rule 1: $\emptyset$ is a Set.

Rule 2: If S is a set and x is anything, $S \cup \{x\}$ is a set.

$\mathbf{0}$, $\emptyset$, $\boldsymbol{S}$, =, $\cup$, {} are all just symbols:
their meaning comes from the definitions we agree to give them.

Recap from Class 2

Rule 1: $\mathbf{0}$ is a Natural Number
Rule 2: If n is a Natural Number, $\boldsymbol{S}(n)$ is a Natural Number.

Also: we show the Principle of Induction
Based on this definition.

Recap from Class 2

Rule 1: $\emptyset$ is a Set.

Rule 2: If S is a set and x is anything, $S \cup \{x\}$ is a set.

end of natural num?

Is this a good definition?

Q: Is (the whole of) the natural numbers a set?

Maybe not

Not by above

Constructive definitions are **finite.**

5.4. Sets

Python also includes a data type for *sets*. A set is an unordered collection with no duplicate elements. Basic uses include membership testing and eliminating duplicate entries. Set objects also support mathematical operations like union, intersection, difference, and symmetric difference.

Curly braces or the `set()` function can be used to create sets. Note: to create an empty set you have to use `set()`, not `{}`; the latter creates an empty dictionary, a data structure that we discuss in the next section.

Here is a brief demonstration:

```
>>> basket = {'apple', 'orange', 'apple', 'pear', 'orange', 'banana'}
>>> print(basket)                      # show that duplicates have been removed
{'orange', 'banana', 'pear', 'apple'}
>>> 'orange' in basket                 # fast membership testing
True
>>> 'crabgrass' in basket
False
```

(Often) Good in programing

Not so in this course!

```
{'r', 'd', 'b'}
>>> a | b                              # letters in a or b or both
{'a', 'c', 'r', 'd', 'b', 'm', 'z', 'l'}
>>> a & b                              # letters in both a and b
{'a', 'c'}
>>> a ^ b                              # letters in a or b but not both
{'r', 'd', 'b', 'm', 'z', 'l'}
```

3. Set

A **set** is a value whose only property is having other values as its **members**. The most common representation of a set is as its members written between braces. The members of a set have no position, order, number of times appearing in the set, or any other properties beyond being a member of the set.

The number of distinct members of a
and is denoted $|S|$. A set with cardin
singleton set.

A set can be defined based on any predicate using **set-builder notation**: $\{f(x) \mid P(x)\}$ is the set of all $f(x)$ where $P(x)$ is true.

> *Example —*
> $\{x + 2 \mid 1 < x \leq 2\}$ is the set of all numbers greater than 3 and no greater than 4.
>
> $x \in \{x \mid P(x)\}$ is a long way of writing $P(x)$.
>
> $\{x \mid x \in S\}$ is a long way of writing S.

DMT1-OTB https://www.cs.virginia.edu/~emo7bf/cs2120/s2023/structures.html

Rule 1: 0 is a Natural Number
Rule 2: If n is a Natural Number, $S(n)$ is a Natural Number.

Practice: Defining +

a b

Rule 1: if a = 0 : a+b is b

Rule 2: else a = S(p) s.t. p ∈ ℕ

a+b is S(p+b)

Question: S(p)+b

↓

S(p+b)

a = S(S(0)) 2

b = S(0) 1

a+b = S(S(0)+S(0))

= S(S(0+S(0)))

Rule 1: $\mathbf{0}$ is a Natural Number
Rule 2: If n is a Natural Number, $\boldsymbol{S}(n)$ is a Natural Number.

Practice: Defining +

Definition. The *sum* of two natural numbers a and b (denoted as $a + b$) is defined as:

(1) If a is $\mathbf{0}$, the $a + b$ sum is b. $\quad \mathbf{0} + b = b$

(2) Otherwise, a is $\boldsymbol{S}(p)$ for some Natural Number p and $a + b$ is $\boldsymbol{S}(p + b)$.

$$(p + 1) + b = (b + p) + 1$$

PS 1 Problem 3: use this definition to prove that addition is commutative: $a + b = b + a$.

Representing the Natural Numbers

Rule 1: $\mathbf{0}$ is a Natural Number
Rule 2: If n is a Natural Number, $\boldsymbol{S}(n)$ is a Natural Number.

Can we represent the Natural Numbers using *finite binary strings*?

TCS, 1.4.2

Strings: Another set we will use time and again is

$$\{0,1\}^n = \{(x_0, \ldots, x_{n-1}) : x_0, \ldots, x_{n-1} \in \{0,1\}\}$$

which is the set of all n-length binary strings for some natural number n. That is $\{0,1\}^n$ is the set of all n-tuples of zeroes and ones. This is consistent with our notation above: $\{0,1\}^2$ is the Cartesian product $\{0,1\} \times \{0,1\}$, $\{0,1\}^3$ is the product $\{0,1\} \times \{0,1\} \times \{0,1\}$ and so on.

We will write the string $(x_0, x_1, \ldots, x_{n-1})$ as simply $x_0 x_1 \cdots x_{n-1}$. For example,

$$\{0,1\}^3 = \{000, 001, 010, 011, 100, 101, 110, 111\} .$$

For every string $x \in \{0,1\}^n$ and $i \in [n]$, we write x_i for the i^{th} element of x.

We will also often talk about the set of binary strings of *all* lengths, which is

$$\{0,1\}^* = \{(x_0, \ldots, x_{n-1}) : n \in \mathbb{N} , , x_0, \ldots, x_{n-1} \in \{0,1\}\} .$$

Another way to write this set is as

$$\{0,1\}^* = \{0,1\}^0 \cup \{0,1\}^1 \cup \{0,1\}^2 \cup \cdots$$

or more concisely as

$$\{0,1\}^* = \cup_{n \in \mathbb{N}} \{0,1\}^n .$$

Binary Strings

Defining Binary Strings

If we want to define binary strings with a constructive recursive definition, what is the best choice for the base clause?

Defining Binary Strings

Rule 1: " " empty string is binary string

Rule 2: binary string S

S0, S1 are also binary string.

Defining Binary Strings

Base: "" (empty) is a Binary String

Inductive: If ***s*** is a Binary String,
both **0*s*** and **1*s*** are Binary Strings.

Another way to write this set is as

$$\{0,1\}^* = \{0,1\}^0 \cup \{0,1\}^1 \cup \{0,1\}^2 \cup \cdots$$

or more concisely as

$$\{0,1\}^* = \cup_{n \in \mathbb{N}} \{0,1\}^n \ .$$

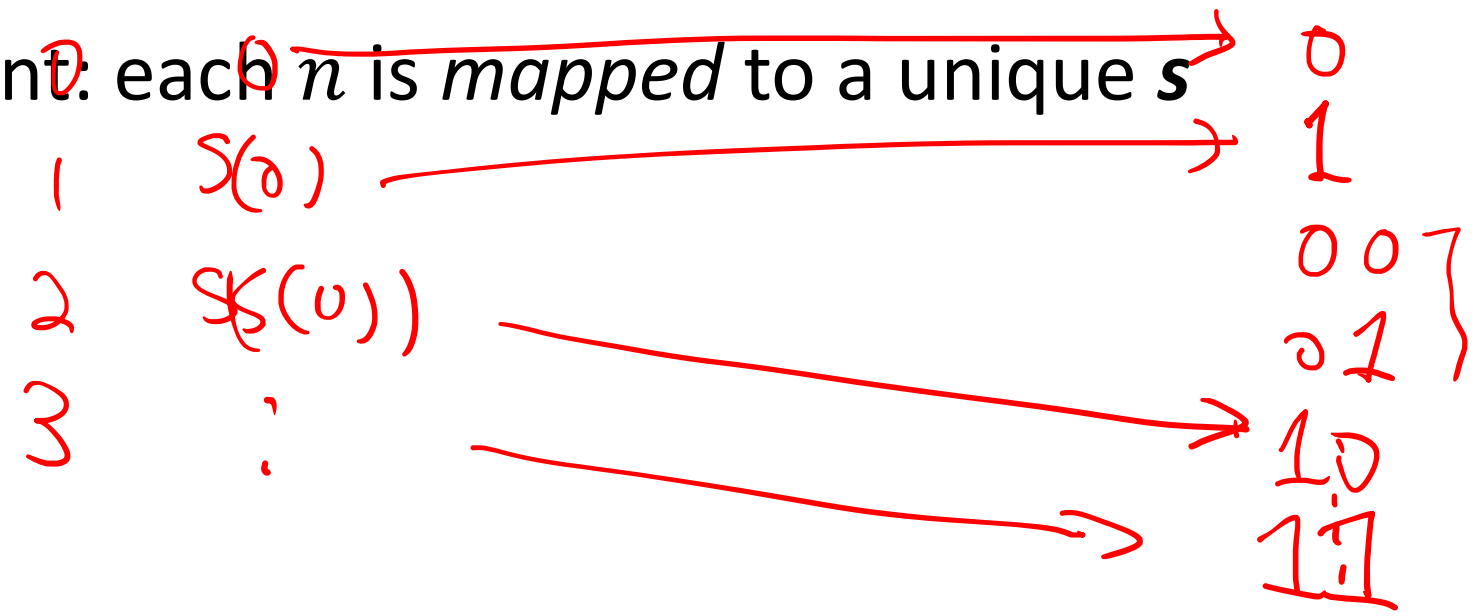
DMT1-OTB

Can all Natural Numbers be represented by Binary Strings?

Rule 1: 0 is a Natural Number
Rule 2: If n is a Natural Number, $S(n)$ is a Natural Number.

Rule 1: "" is a Binary String
Rule 2: If ***s*** is a Binary String, both **0*s*** and **1*s*** are Binary Strings.

Want: each n is *mapped* to a unique ***s***

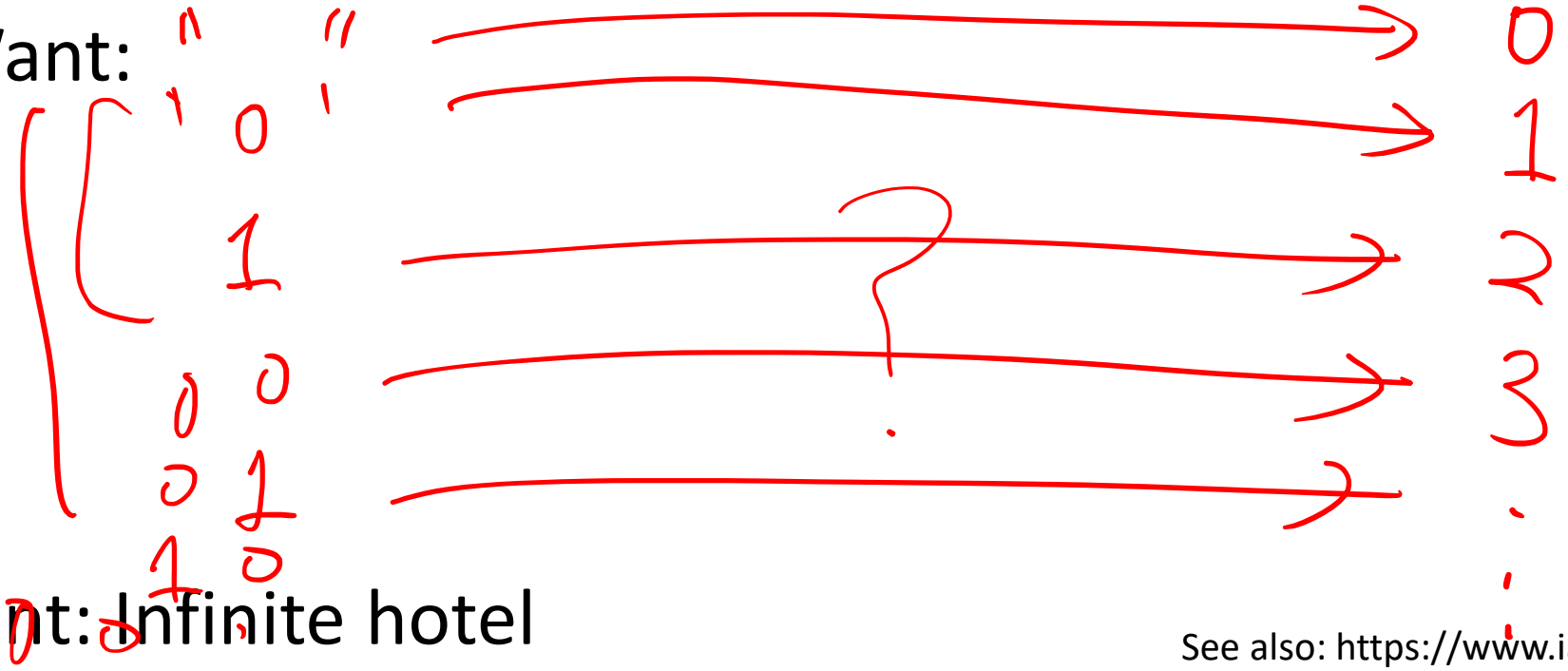


Can all Binary Strings be represented by Natural Numbers?

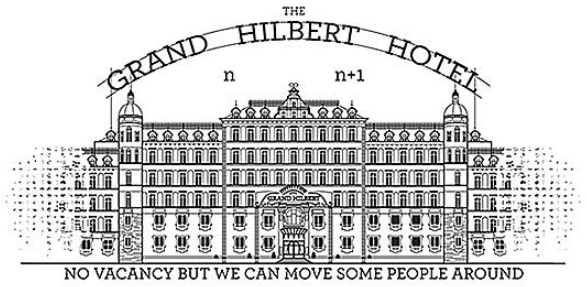
Rule 1: "" is a Binary String
Rule 2: If ***s*** is a Binary String, both **0*s*** and **1*s*** are Binary Strings.

Rule 1: 0 is a Natural Number
Rule 2: If n is a Natural Number, $S(n)$ is a Natural Number.

Want:



Hint: Infinite hotel



See also: https://www.ias.edu/ideas/2016/pires-hilbert-hotel

$\{0, 1\}^*$ vs. $\{0, 1\}^\infty$

Is “0101010101010101…” a Binary String?

Rule 1: “” is a Binary String
Rule 2: If ***s*** is a Binary String, both **0*s*** and **1*s*** are Binary Strings.

$\{0, 1\}^*$ vs. $\{0, 1\}^\infty$

Is “0101010101010101…” a Binary String?

Rule 1: “” is a Binary String
Rule 2: If ***s*** is a Binary String, both **0*s*** and **1*s*** are Binary Strings.

Constructive definitions are **finite.** We can product a binary string of *any* length from these rules but cannot produce an infinite binary string.

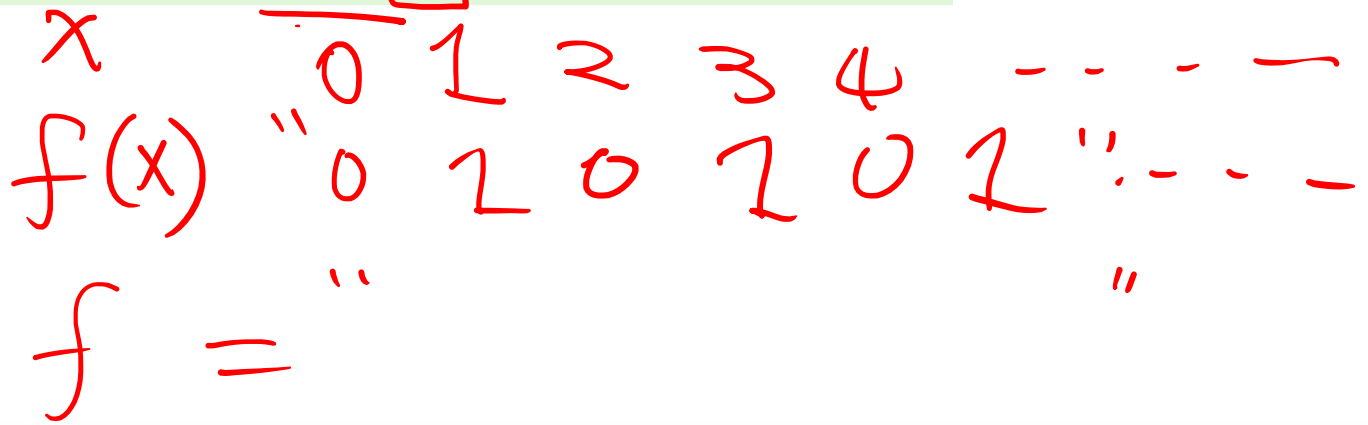
$\{0,1\}^*$ vs. $\{0,1\}^\infty$

$\{0,1\}^*$ finite binary strings of any length

$\{0,1\}^\infty$ "infinite binary strings"

Definition 2.7

We denote by $\{0,1\}^\infty$ the set $\{f \mid f : \mathbb{N} \to \{0,1\}\}$.



A set can be defined based on any predicate using **set-builder notation**: $\{f(x) \mid P(x)\}$ is the set of all $f(x)$ where $P(x)$ is true.

DMT1-OTB

$\{0,1\}^*$ vs. $\{0,1\}^\infty$

$\{0,1\}^*$ finite binary strings of any length

$\{0,1\}^\infty$ "infinite binary strings"

Definition 2.7

We denote by $\{0,1\}^\infty$ the set $\{f \mid f : \mathbb{N} \to \{0,1\}\}$.

Why we need infinite binary strings, $\{0,1\}^\infty$?

"$\{0,1\}^*$ cannot represent all nat $\mathbb{N}$" ?

$\pi = 3.14159\ldots$ need ∞ inf long bin string to represent $\mathbb{R}$

Buttered Cat?

https://www.reddit.com/r/funny/comments/6ws5or/buttered_cat_paradox/

Cardinality of Sets

$$|S|$$

Cardinality of Finite Sets

Definition. Two sets have the *same cardinality* if there is a bijection between the two sets.

9. For two sets A and B, which of the following implies |A| = |B|? (select all that apply) *

- [ ] A = B
- [ ] A is a subset of B and B is a subset of A
- [ ] A intersected with the complement of B is the empty set
- [ ] There is a bijection (1-1 correspondence) between A and B
- [ ] I don't know what's |A| = |B|?

Cardinality of Finite Sets

Definition. Two sets have the *same cardinality* if there is a bijection between the two sets.

The ***cardinality*** of the set

$$[k] = \{\, n \mid n \in \mathbb{N} \wedge n < k \,\}$$

is k.

Example: $S = \{C, S, 3120\}$, $|S| = ?$

3

$[3] = \{0, 1, 2\}$

$|[3]| = 3$

Cardinality of Infinite Sets

Definition. Two sets have the *same cardinality* if there is a bijection between the two sets.

The ***cardinality*** of the set

$$[k] = \{\, n \mid n \in \mathbb{N} \wedge n < k \,\}$$

is k.

countable

$$|\{0,1\}^*| = ?\ \infty$$

Cardinality of Infinite Sets

Definition. Two sets have t… *…dinality* if there is a bijection b… two sets.

The ***cardinal…*** …set

$$\{\, n \mid n \in \mathbb{N} \wedge n < k \,\}$$

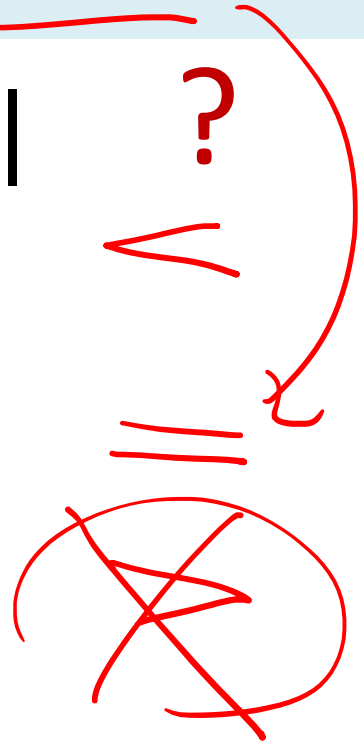
is k.

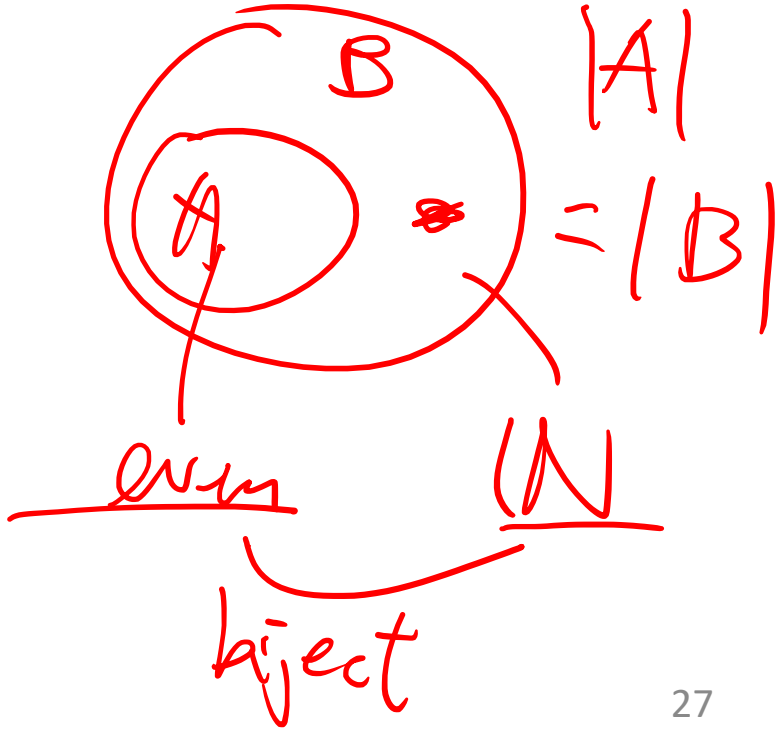
A set S is infinite, if there is no bijection between S and any $[k]$.

Cardinality of (Infinite) Sets

Definition. Two sets have the *same cardinality* if there is a bijection between the two sets.

If $A \subsetneq B$, then $|A| \; ? \; |B|$



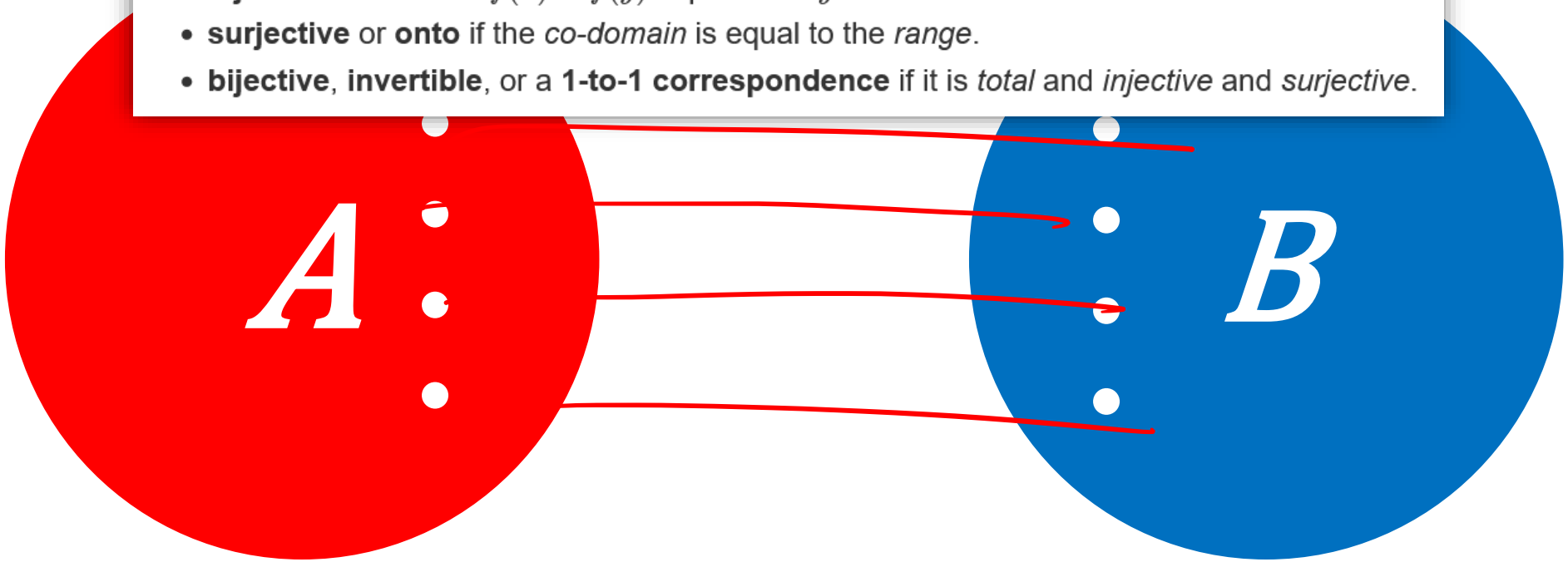


Function (DMT1 Review)

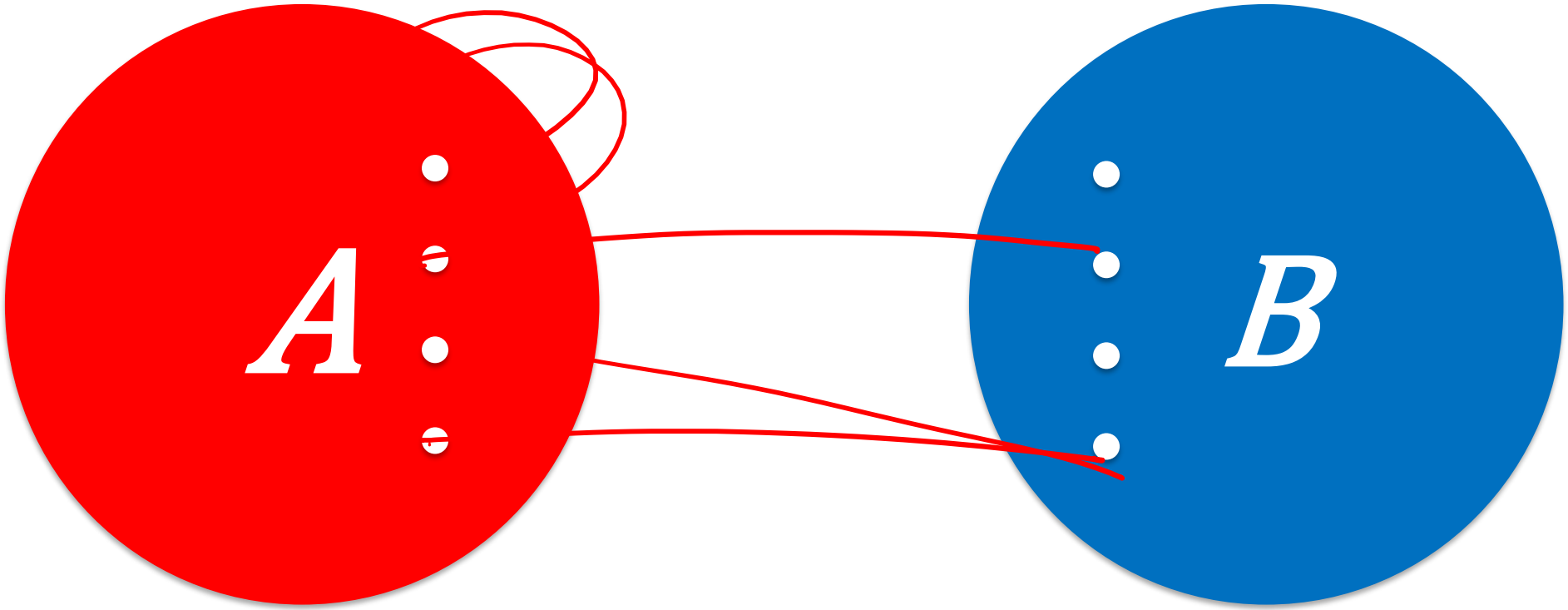
DMT1-OTB

A *function* $f : D \to C$ is

- **total** if $f(x)$ is defined for all $x \in D$.
- **partial** means "either total or not total" and is used in contexts where most functions are assumed to be total to identify subcontexts where that assumption does not apply.
- **injective** or **1-to-1** if $f(x) = f(y)$ implies $x = y$.
- **surjective** or **onto** if the *co-domain* is equal to the *range*.
- **bijective**, **invertible**, or a **1-to-1 correspondence** if it is *total* and *injective* and *surjective*.



Function Properties



function: ≤ 1 out

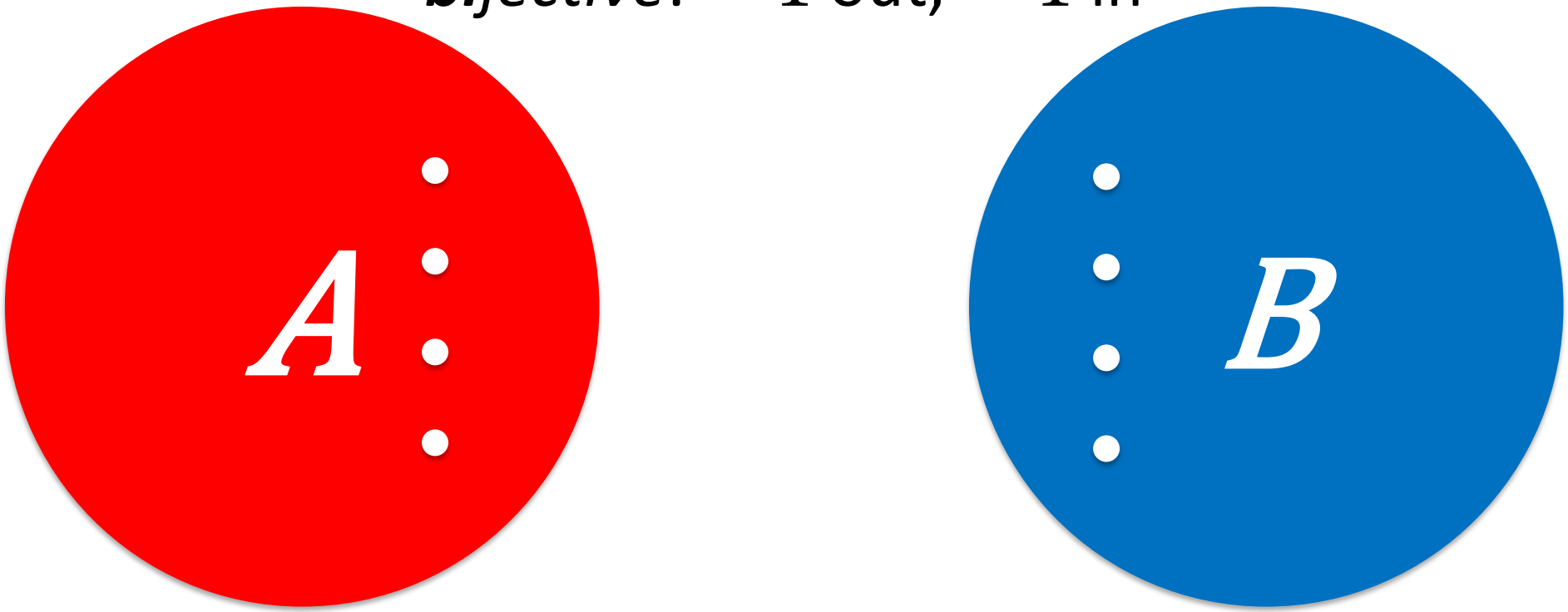
total: ≥ 1 out

***in**jective*: ≤ 1 in

***sur**jective*: ≥ 1 in

Function Properties

bi*jective*: $=1$ out, $=1$ in



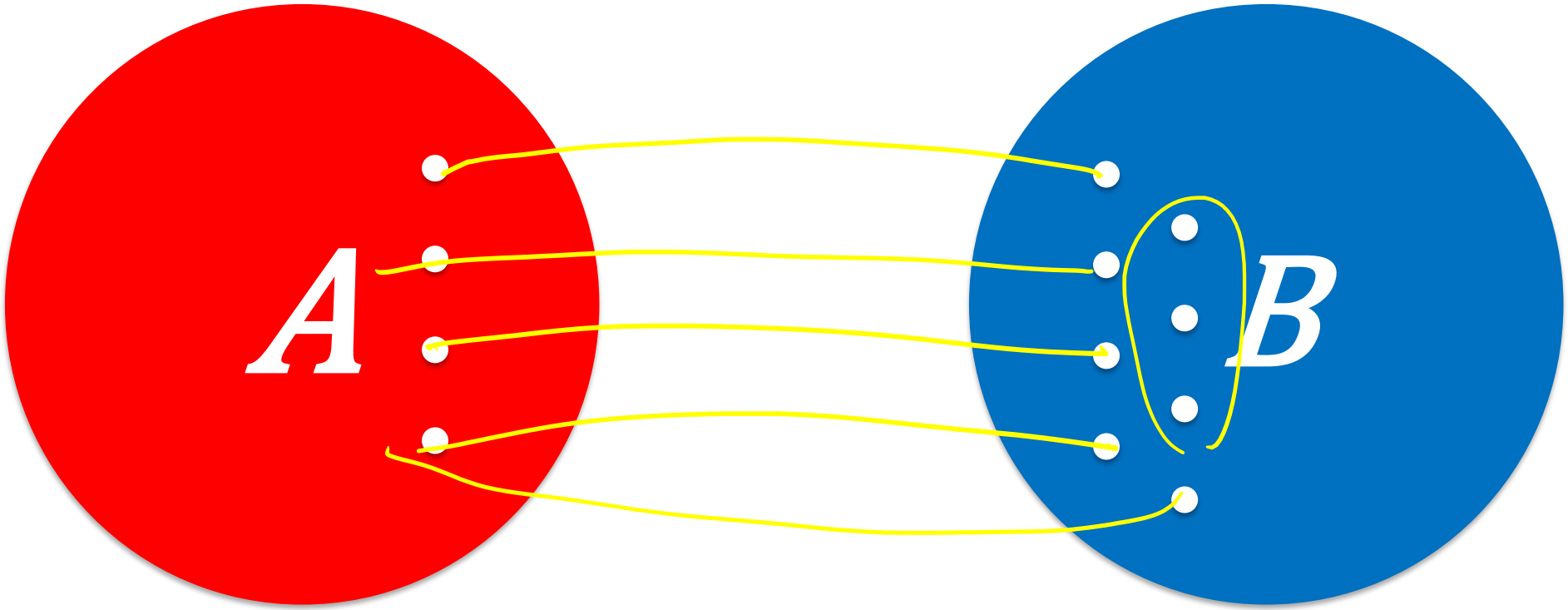
function: ≤ 1 out

total: ≥ 1 out

in*jective*: ≤ 1 in

sur*jective*: ≥ 1 in

Surjective function from B to A



function: ≤ 1 out

sur*jective*: ≥ 1 in

$|A| \leq |B|$

Cardinality of (Infinite) Sets

Definition. Two sets have the *same cardinality* if there is a bijection between the two sets.

Definition. If there exists a **surjective function** from sets B to A, then we say the cardinality of B is ***greater than or equal to*** the cardinality of A.

We denote this as $|A| \leq |B|$.

$|\text{even}| \leq |\mathbb{N}|$

$|\mathbb{N}| \leq |\text{even}| \Rightarrow$ "$=$"

Two Useful (Intuitively obvious?) Facts

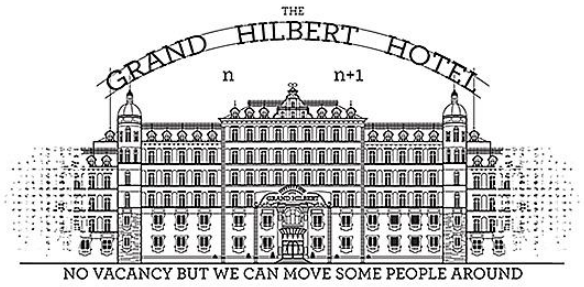
1. If $|A| = |B|$ then $|A| \leq |B|$ and $|B| \leq |A|$.

biject → surj → surj

2. If $|A| \leq |B|$ and $|B| \leq |A|$ then $|A| = |B|$.

surj, surj → biject

Are these “obvious” facts, or do we need a proof?



Charge

Definition Practice: Addition, Binary Strings

Set Cardinality

Infinite

Countable

Power set

Problem Set 1 is due
This Friday, Jan 24 (10:00pm)

Tomorrow 10am (Jan 22), Greenberry's Coffee (Wilsdorf)
Chat with Wei-Kai, free food & drink