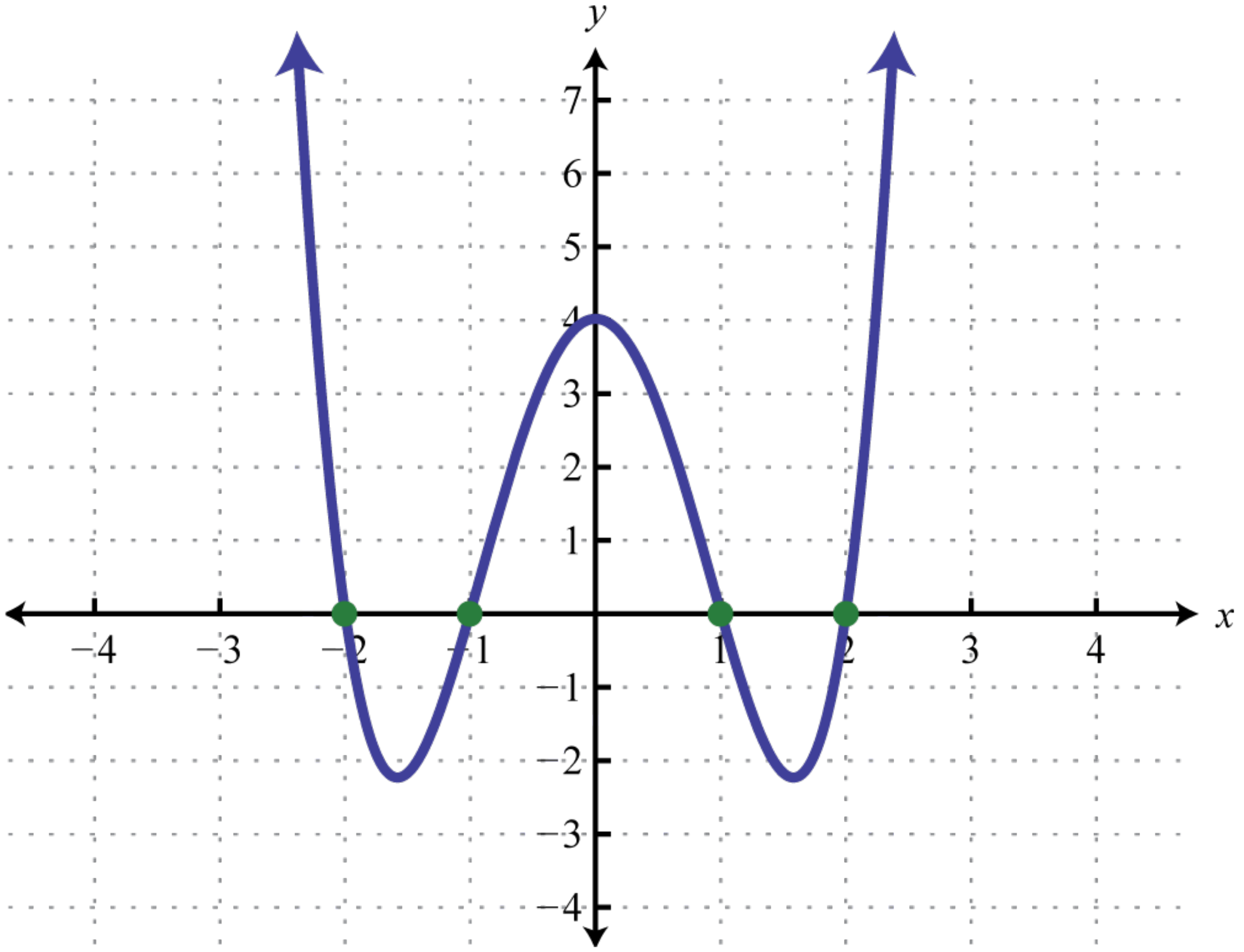
Functions

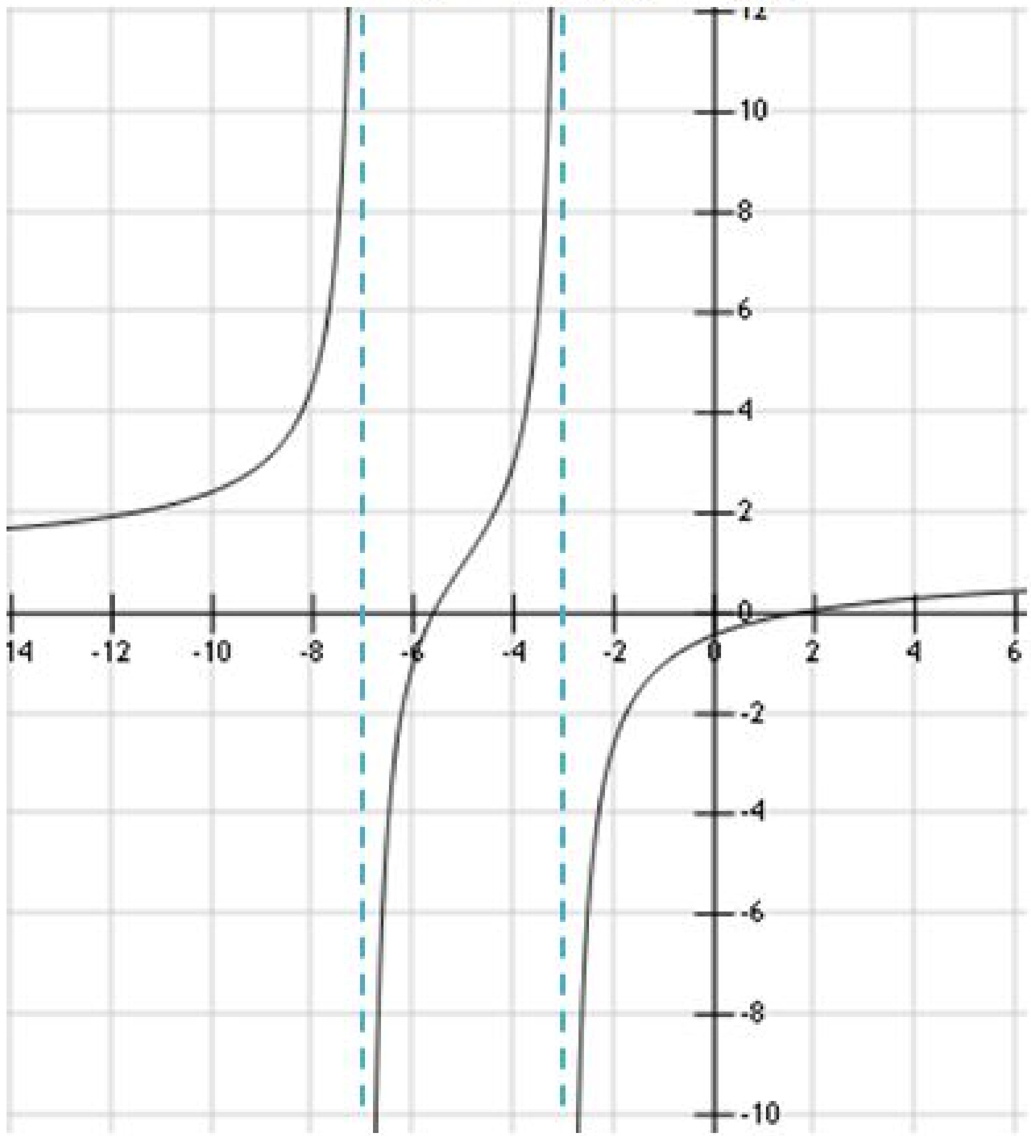
What is a function?

Functions, High-School Edition



$$f(x) = x^4 - 5x^2 + 4$$

$$f(x) = \frac{x^2 + 4x - 9}{x^2 + 10x + 21}$$



source: http://study.com/cimages/multimages/16/asymptote_1.JPG

Functions, High-School Edition

- In high school, functions are usually given as objects of the form

$$f(x) = \frac{x^3 + 3x^2 + 15x + 7}{1 - x^{137}}$$

- What does a function do?
  - Takes in as input a real number.
  - Outputs a real number.
  - … except when there are vertical asymptotes or other discontinuities, in which case the function doesn't output anything.

Functions, CS Edition

```
int flipUntil(int n) {
  int numHeads = 0;
  int numTries = 0;

  while (numHeads < n) {
    if (randomBoolean()) numHeads++;

    numTries++;
  }

  return numTries;
}
```

Functions, CS Edition

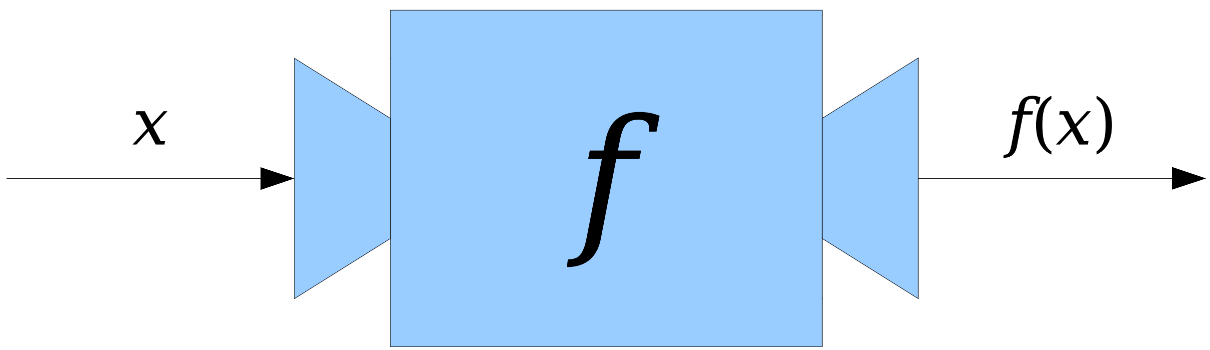
- In programming, functions
  - might take in inputs,
  - might return values,
  - might have side effects,
  - might never return anything,
  - might crash, and
  - might return different values when called multiple times.

What's Common?

- Although high-school math functions and CS functions are pretty different, they have two key aspects in common:
  - They take in inputs.
  - They produce outputs.
- In math, we like to keep things easy, so that's pretty much how we're going to define a function.

Rough Idea of a Function:

A function is an object f that takes in one input and produces exactly one output.



(This is not a complete definition – we'll revisit this in a bit.)

High School versus CS Functions

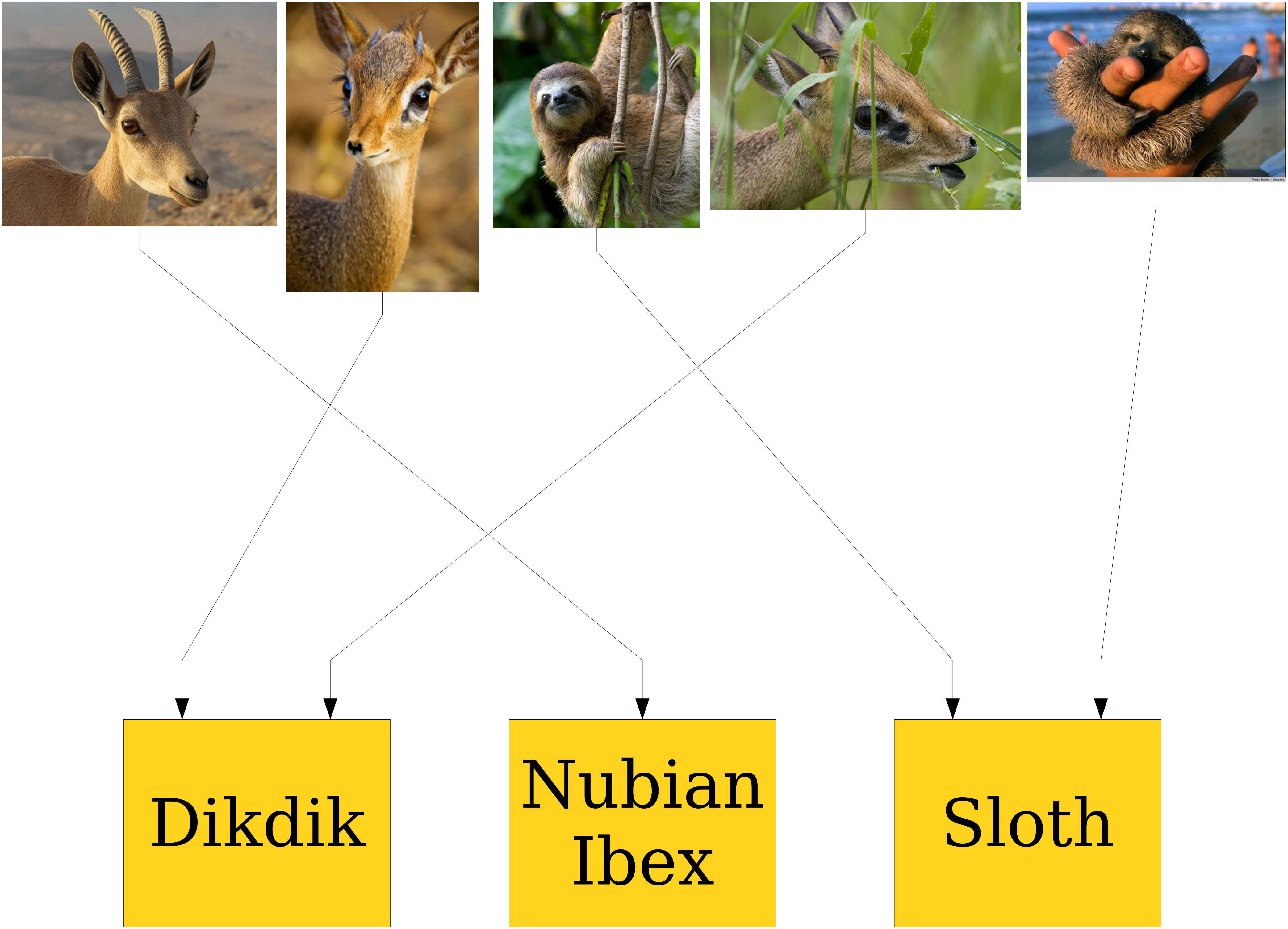
- In high school, functions usually were given by a rule:

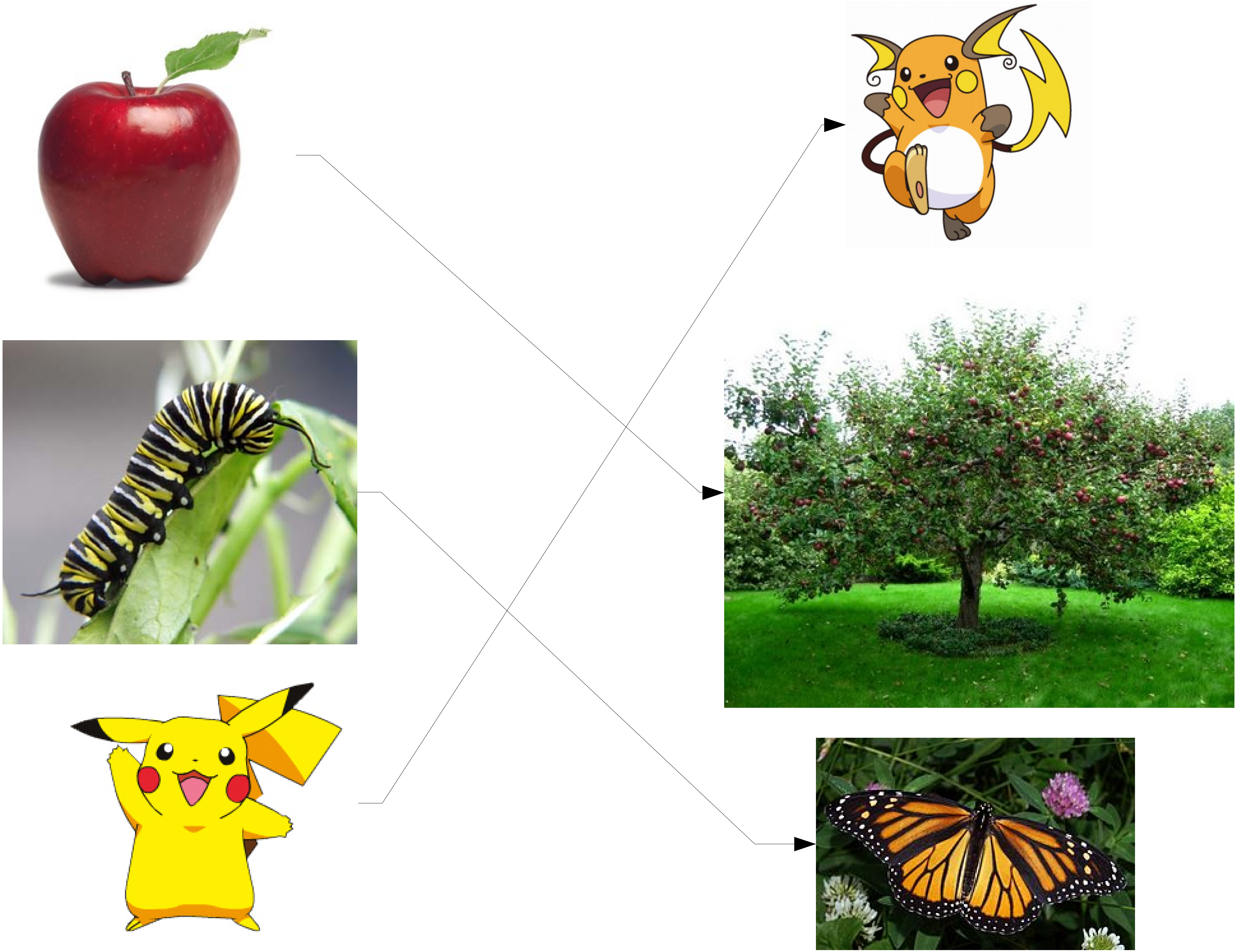
$$f(x) = 4x + 15$$

- In CS, functions are usually given by code:

```
int factorial(int n) {
    int result = 1;
    for (int i = 1; i <= n; i++) {
        result *= i;
    }
    return result;
}
```

- What sorts of functions are we going to allow from a mathematical perspective?


Dikdik
Nubian Ibex
Sloth



… but also …

$$f(x) = x^2 + 3x - 15$$

$$f(n)=\begin{cases} -n/2 & \text{if } n \text{ is even} \\ (n+1)/2 & \text{otherwise} \end{cases}$$

Functions like these are called ***piecewise functions***.

To define a function, you will typically either

- draw a picture, or
- give a rule for determining the output.

In mathematics, functions are ***deterministic***.

That is, given the same input, a function must always produce the same output.

The following is a perfectly valid piece of C++ code, but it's not a valid function under our definition:

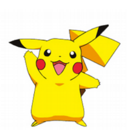
```
int randomNumber(int numOutcomes) {
    return rand() % numOutcomes;
}
```

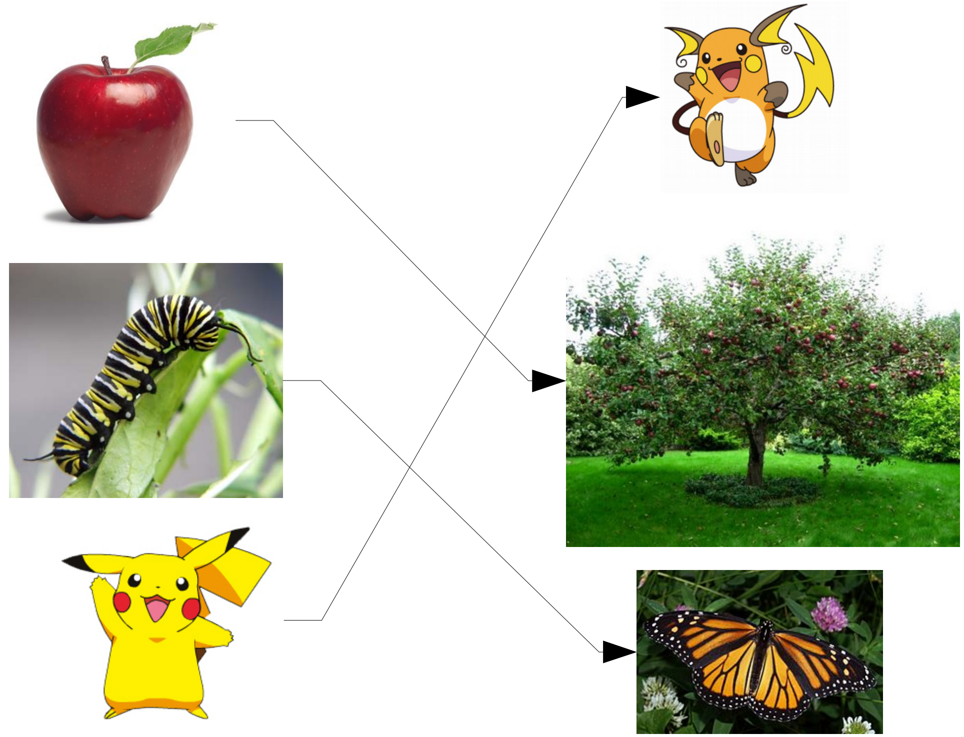
One Challenge

$$f(x) = x^2 + 2x + 5$$

$$f(\ 3\ ) = 3^2 + 3 \cdot 2 + 5 = 20$$

$$f(\ 0\ ) = 0^2 + 0 \cdot 2 + 5 = 5$$

$f($  $) = \ldots\ ?$



$f($  $) =$ 

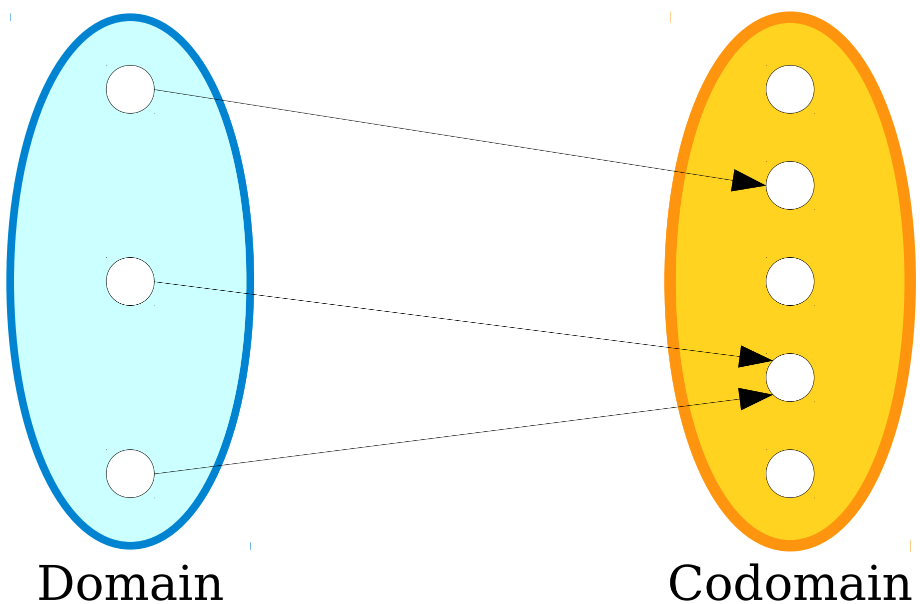
$f(137) = \ldots ?$

We need to make sure we can't apply functions to meaningless inputs.

Domains and Codomains

- Every function *f* has two sets associated with it: its ***domain*** and its ***codomain***.
- A function *f* can only be applied to elements of its domain. For any *x* in the domain, *f*(*x*) belongs to the codomain.

The function must be defined for every element of the domain.



The output of the function must always be in the codomain, but not all elements of the codomain need to be produced.

Domains and Codomains

- Every function *f* has two sets associated with it: its ***domain*** and its ***codomain***.
- A function *f* can only be applied to elements of its domain. For any *x* in the domain, *f*(*x*) belongs to the codomain.

The codomain of this function is $\mathbb{R}$. Everything returned is a real number, but not all real numbers can necessarily be returned.

The domain of this function is $\mathbb{R}$. Any real number can be provided as input.

```
private double absoluteValue(double x) {
    if (x >= 0) {
        return x;
    } else {
        return -x;
    }
}
```

Domains and Codomains

- If f is a function whose domain is A and whose codomain is B, we write $\boldsymbol{f : A \to B}$.
- This notation just says what the domain and codomain of the function is. It doesn't say how the function is evaluated.
- Think of it like a "function prototype" in C or C++. The notation $f : ArgType \to RetType$ is like writing

  RetType *f*(*ArgType* `argument`);

  We know that f takes in an *ArgType* and returns a *RetType*, but we don't know exactly which *RetType* it's going to return for a given *ArgType*.

Domains and Codomains

- A function *f* must be defined for every element of the domain.
  - For example, if $f : \mathbb{R} \to \mathbb{R}$, then the following function is ***not*** a valid choice for *f*:

$$f(x) = 1 / x$$

- The output of *f* on any element of its domain must be an element of the codomain.
  - For example, if $f : \mathbb{R} \to \mathbb{N}$, then the following function is ***not*** a valid choice for *f*:

$$f(x) = x$$

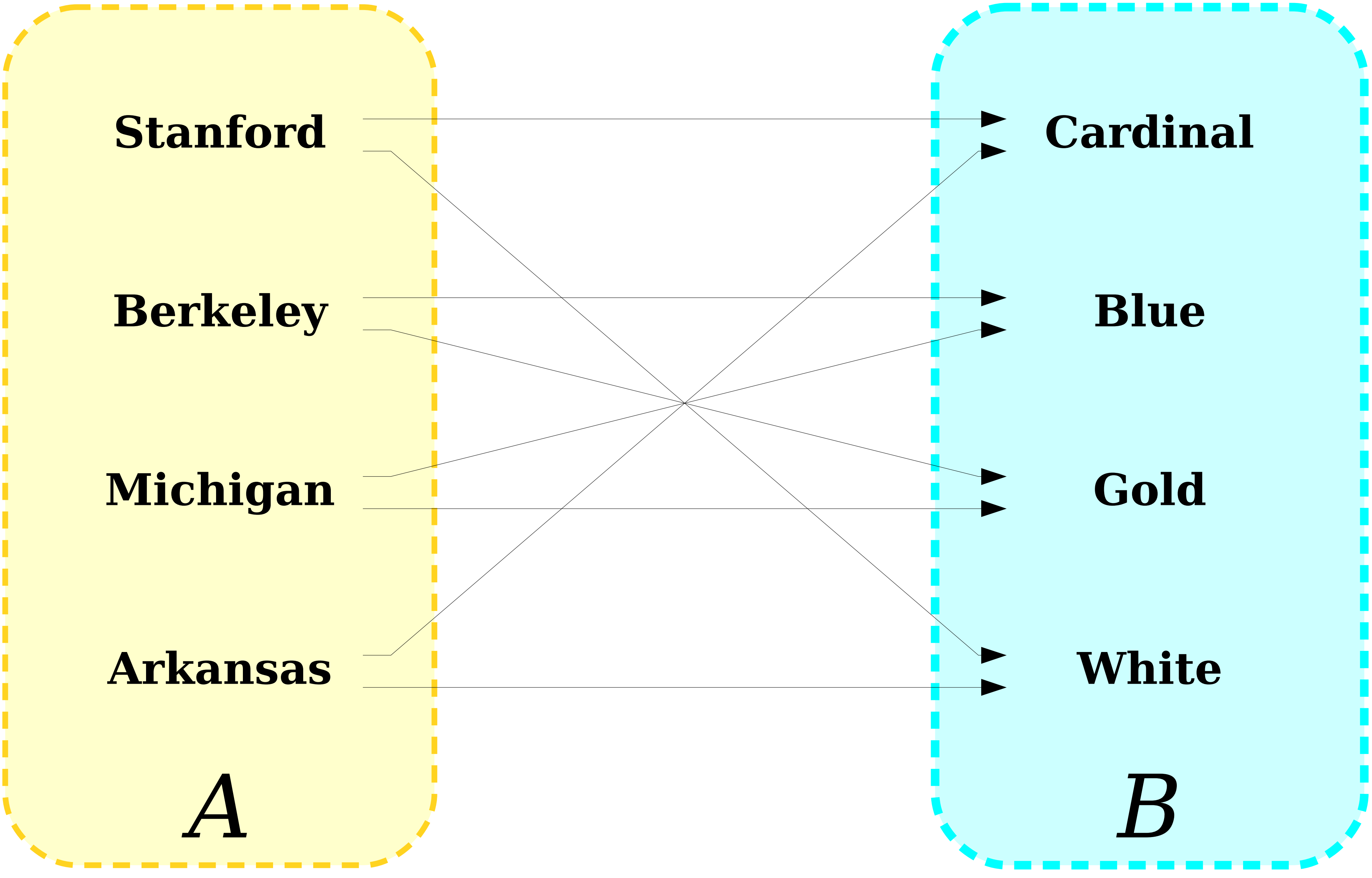
- However, a function *f* does not have to produce all possible values in its codomain.
  - For example, if $f : \mathbb{N} \to \mathbb{N}$, then the following function is a valid choice for *f*:

$$f(n) = n^2$$

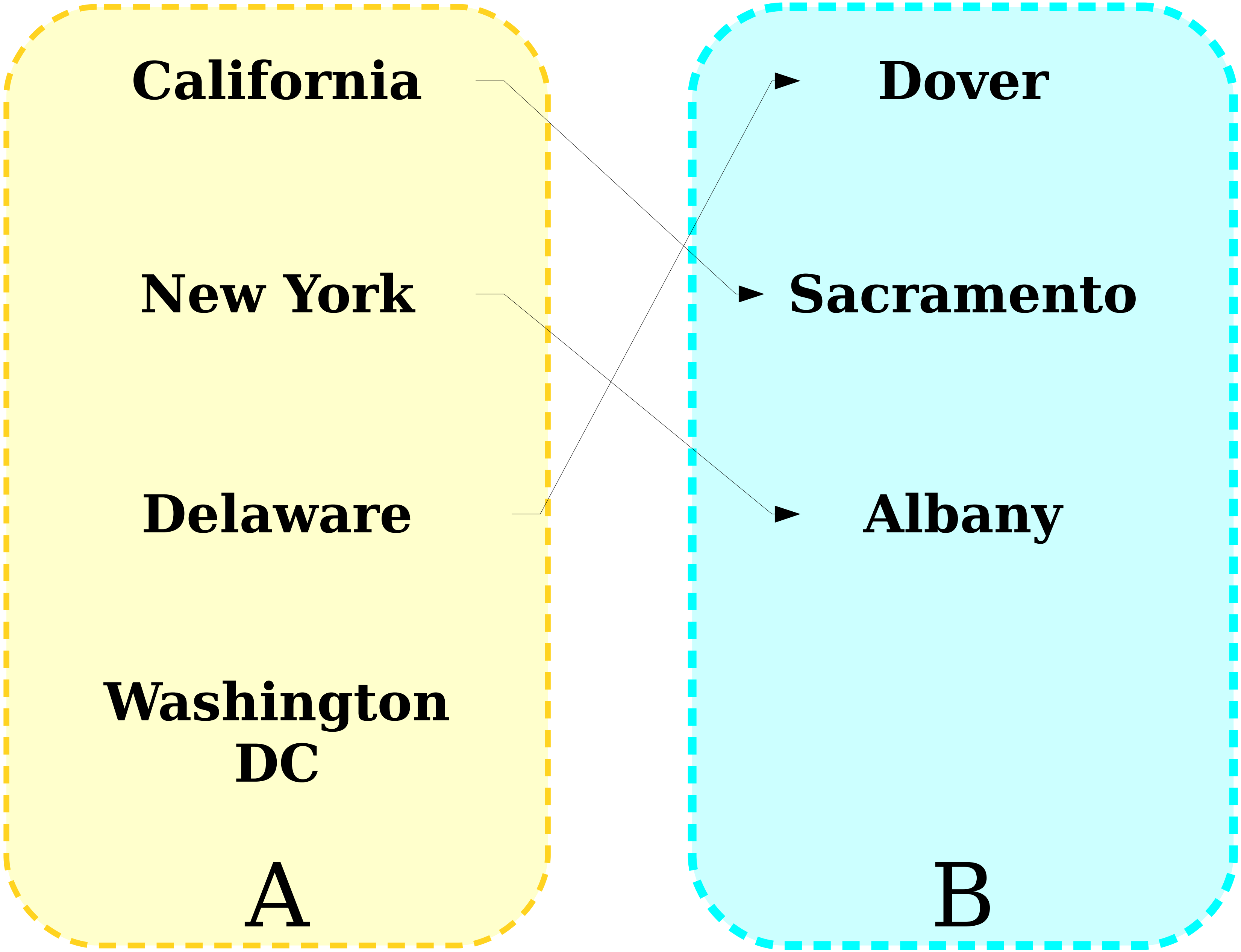
Defining Functions

- Typically, we specify a function by describing a rule that maps every element of the domain to some element of the codomain.
- Examples:
  - $f(n) = n + 1$, where $f : \mathbb{Z} \to \mathbb{Z}$
  - $f(x) = \sin x$, where $f : \mathbb{R} \to \mathbb{R}$
  - $f(x) = \lceil x \rceil$, where $f : \mathbb{R} \to \mathbb{Z}$
- Notice that we're giving both a rule and the domain/codomain.

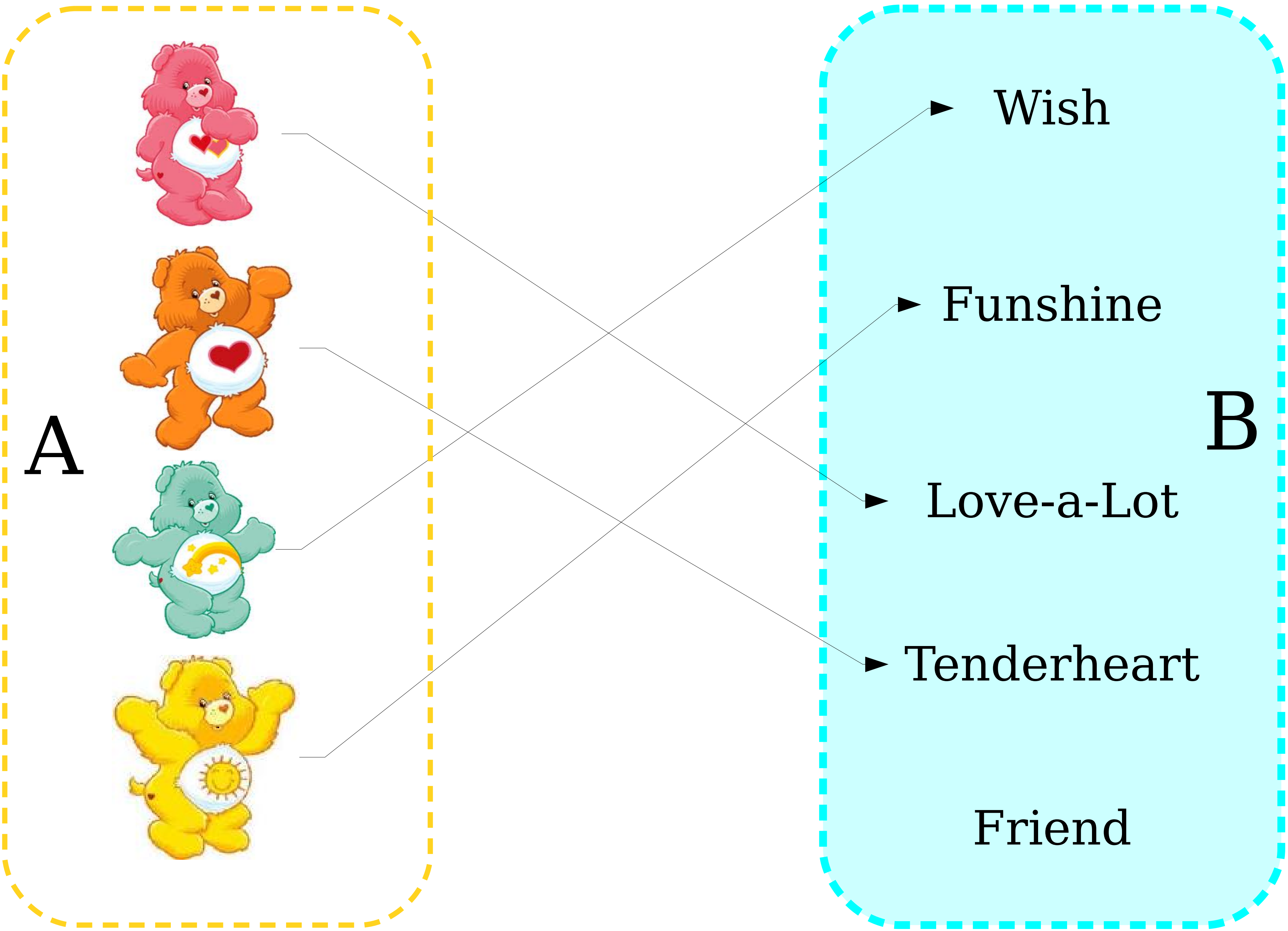
Is this a function from A to B?



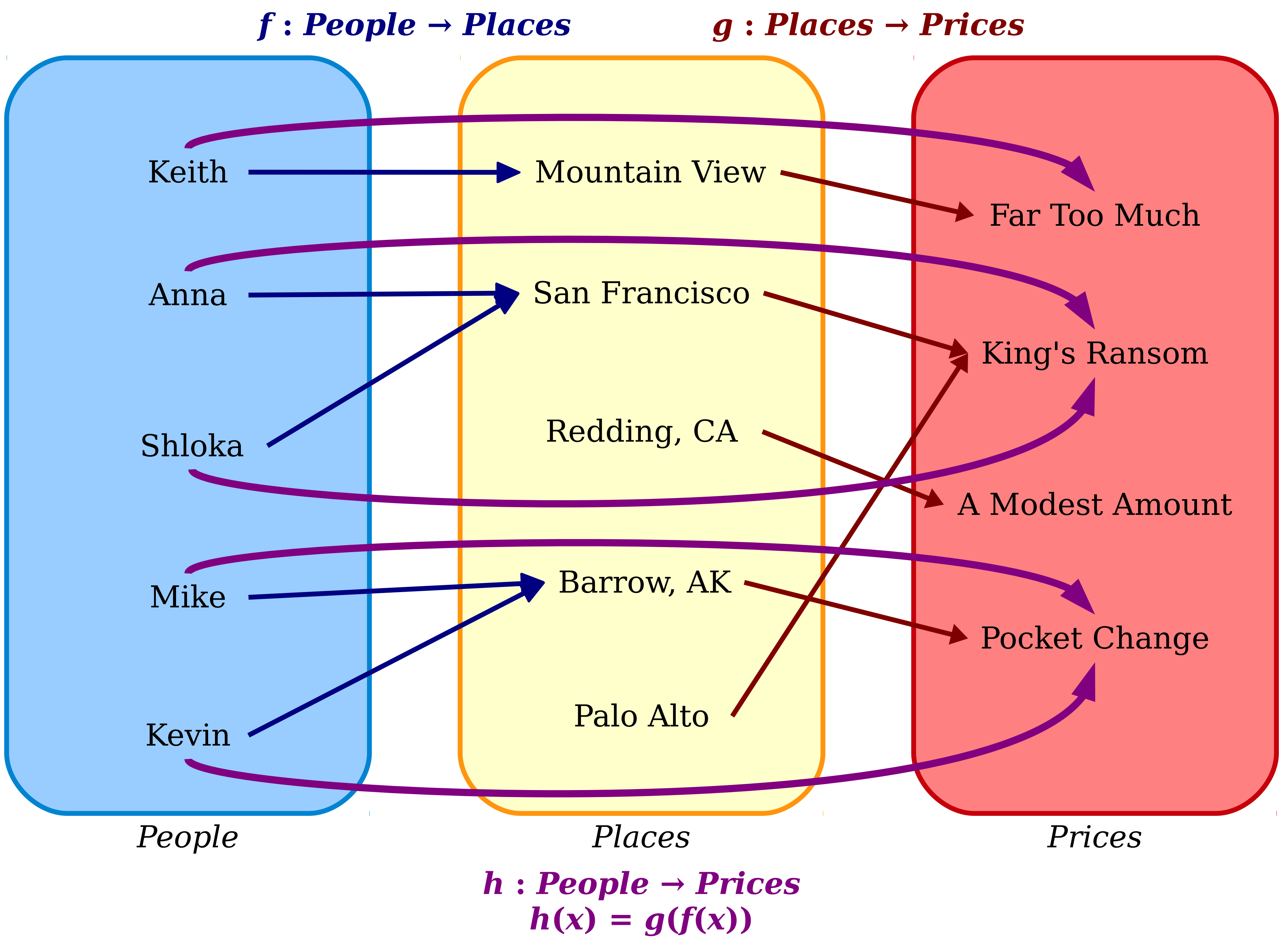
Is this a function from A to B?



Is this a function from A to B?



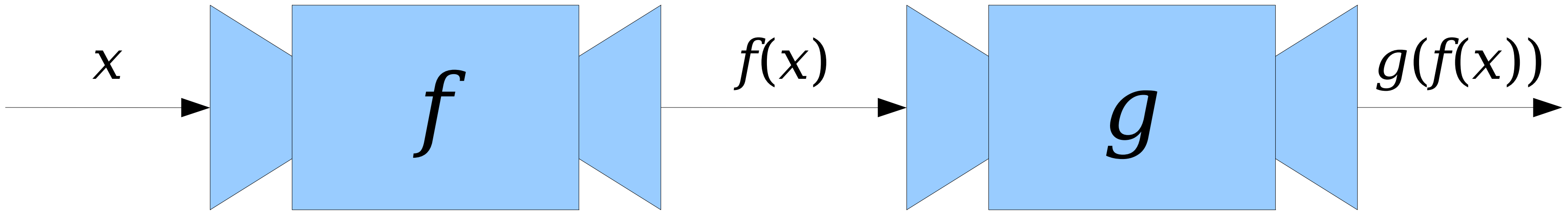
Combining Functions



f : People → Places
g : Places → Prices
Keith
Anna
Shloka
Mike
Kevin
Mountain View
San Francisco
Redding, CA
Barrow, AK
Palo Alto
Far Too Much
King's Ransom
A Modest Amount
Pocket Change
People
Places
Prices
h : People → Prices
h(x) = g(f(x))

Function Composition

- Suppose that we have two functions $f : A \to B$ and $g : B \to C$.
- Notice that the codomain of f is the domain of g. This means that we can use outputs from f as inputs to g.



Function Composition

- Suppose that we have two functions $f : A \to B$ and $g : B \to C$.
- The ***composition of f and g***, denoted $\boldsymbol{g \circ f}$, is a function where
  - $g \circ f : A \to C$, and
  - $(g \circ f)(x) = g(f(x))$.

> The name of the function is $g \circ f$. When we apply it to an input x, we write $(g \circ f)(x)$. I don't know why, but that's what we do.

- A few things to notice:
  - The domain of $g \circ f$ is the domain of f. Its codomain is the codomain of g.
  - Even though the composition is written $g \circ f$, when evaluating $(g \circ f)(x)$, the function f is evaluated first.

Function Composition

- Let $f : \mathbb{N} \to \mathbb{N}$ be defined as $f(n) = 2n + 1$ and $g : \mathbb{N} \to \mathbb{N}$ be defined as $g(n) = n^2$.
- What is $g \circ f$?

$$\begin{aligned}(g \circ f)(n) &= g(f(n)) \\ &= g(2n + 1) \\ &= (2n + 1)^2 = 4n^2 + 4n + 1\end{aligned}$$

- What is $f \circ g$?

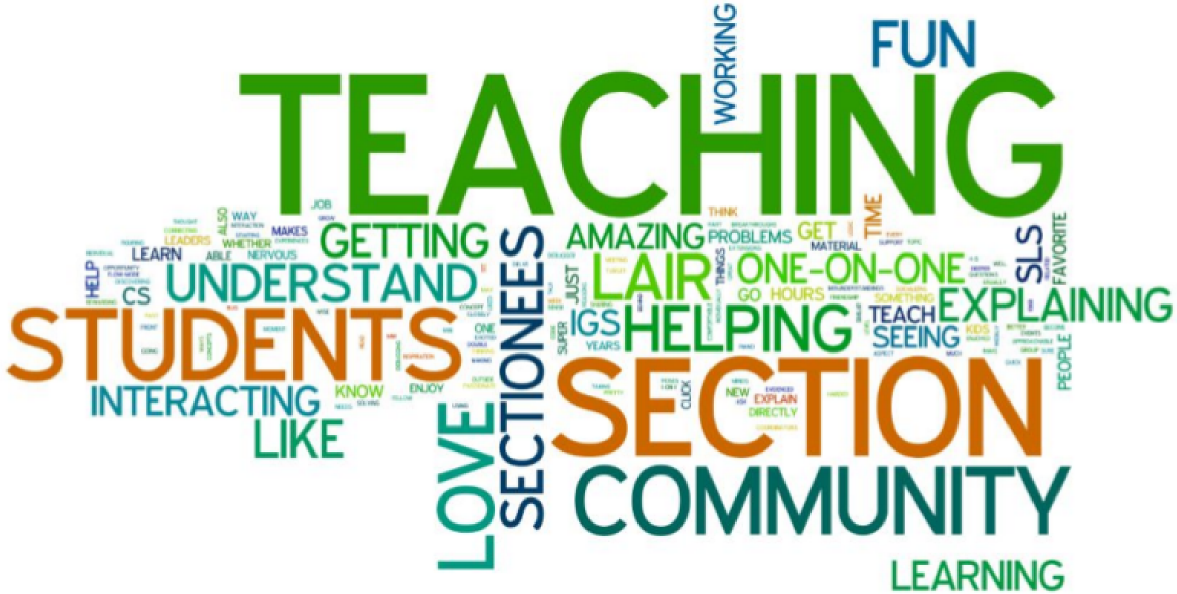
$$\begin{aligned}(f \circ g)(n) &= f(g(n)) \\ &= f(n^2) \\ &= 2n^2 + 1\end{aligned}$$

- In general, if they exist, the functions $g \circ f$ and $f \circ g$ are usually not the same function. ***Order matters in function composition!***

Time-Out for Announcements!

Problem Sets

- Problem Set Two was due at 3:00PM today. You can extend the deadline using your 24-hour late days until Monday at 3:00PM.
- Problem Set Three goes out right now.
  - The checkpoint problem is due on Monday at the start of class.
  - The remaining problems are due next Friday at the start of class.
  - Explore binary relations, functions, and proofs on discrete structures!
- As always, please keep asking us questions, whether in office hours, over Piazza, or using the staff list.



CS106 SECTION LEADING

Deadline for students who have completed CS106B/X (or equivalent): Thursday, October 20th, 11:59pm

Deadline for students currently enrolled in CS106B/X: Thursday, November 3rd, 11:59pm

All majors and backgrounds welcome!

Applicants must be currently enrolled in or have completed CS106B/X, or have equivalent experience.

Apply online!

cs198.stanford.edu

Questions? Contact the CS198 Coordinators (Aaron Broder and Greg Ramel) at cs198@cs.stanford.edu

Latinos in Technology Scholarship

HISPANIC FOUNDATION OF SILICON VALLEY

- "This scholarship, established with Silicon Valley Community Foundation (SVCF) by the Hispanic Foundation of Silicon Valley, will present up to 100 Latino students with up to $30,000 in scholarship aid to help them continue their education in science, technology, engineering and math (STEM) programs. Recipients will also have the opportunity to be placed into internships with leading companies in Silicon Valley."
- More information available via **this link**.

Problem Set One: A Common Mistake

An Incorrect Proof

Theorem: For any integers x, y, and k, if $x \equiv_k y$, then $y \equiv_k x$.

Proof: Consider any arbitrary integers x, y, and k where $x \equiv_k y$. This means that there is an integer q where $x - y = kq$. We need to prove that $y \equiv_k x$, meaning that we need to prove that there is an integer r where $y - x = kr$.

Since $y - x = kr$, we see that $x - y = -kr$. Earlier we noted that $x - y = kq$, so collectively we see that $-kr = kq$. Therefore, we see that $r = -q$. ■

An Incorrect Proof

Theorem: For any integers x, y, and k, if $x \equiv_k y$, then $y \equiv_k x$.

Proof: Consider any arbitrary integers x, y, and k where $x \equiv_k y$. This means that there is an integer q where $x - y = kq$. We need to prove that $y \equiv_k x$, meaning that we need to prove that there is an integer r where $y - x = kr$.

Since $y - x = kr$, we see that $x - y = -kr$. Earlier we noted that $x - y = kq$, so collectively we see that $-kr = kq$. Therefore, we see that $r = -q$. ■

We're assuming what we're trying to prove!

A Better Proof

Theorem: For any integers x, y, and k, if $x \equiv_k y$, then $y \equiv_k x$.

Proof: Consider any arbitrary integers x, y, and k where $x \equiv_k y$. This means that there is an integer q where $x - y = kq$. We need to prove that $y \equiv_k x$, meaning that we need to prove that there is an integer r where $y - x = kr$.

Since $x - y = kq$, we see that $y - x = -kq = k(-q)$. Therefore, there is an integer r, namely $-q$, such that $y - x = kr$. Consequently, we see that $y \equiv_k x$, as required. ■

Notice that we start with our initial assumptions and use them to derive the required result.

General Advice

- ***Be careful not to assume what you're trying to prove.***
- In a proof, we recommend using the phrases “we need to show that” or “we need to prove that” to clearly indicate your goals.
- If you later find yourself relying on a statement marked “we need to prove that,” chances are you've made an error in your proof.

Your Questions

“How do you know CS is right for you or if you're just being sucked down the Stanford CS vortex?”

Rather than giving you positive reasons why you should stay in the field, let me give you a list of bad reasons for leaving the field.

“Everyone has been doing this forever and I'll never catch up.” That's just not the case. You'd be amazed how much and how quickly you'll learn.

“Everyone can do this faster than me.” Trust us, we can tell when someone rushed through something.

“Everyone knows more than me.” Allow me to demonstrate with a visual.

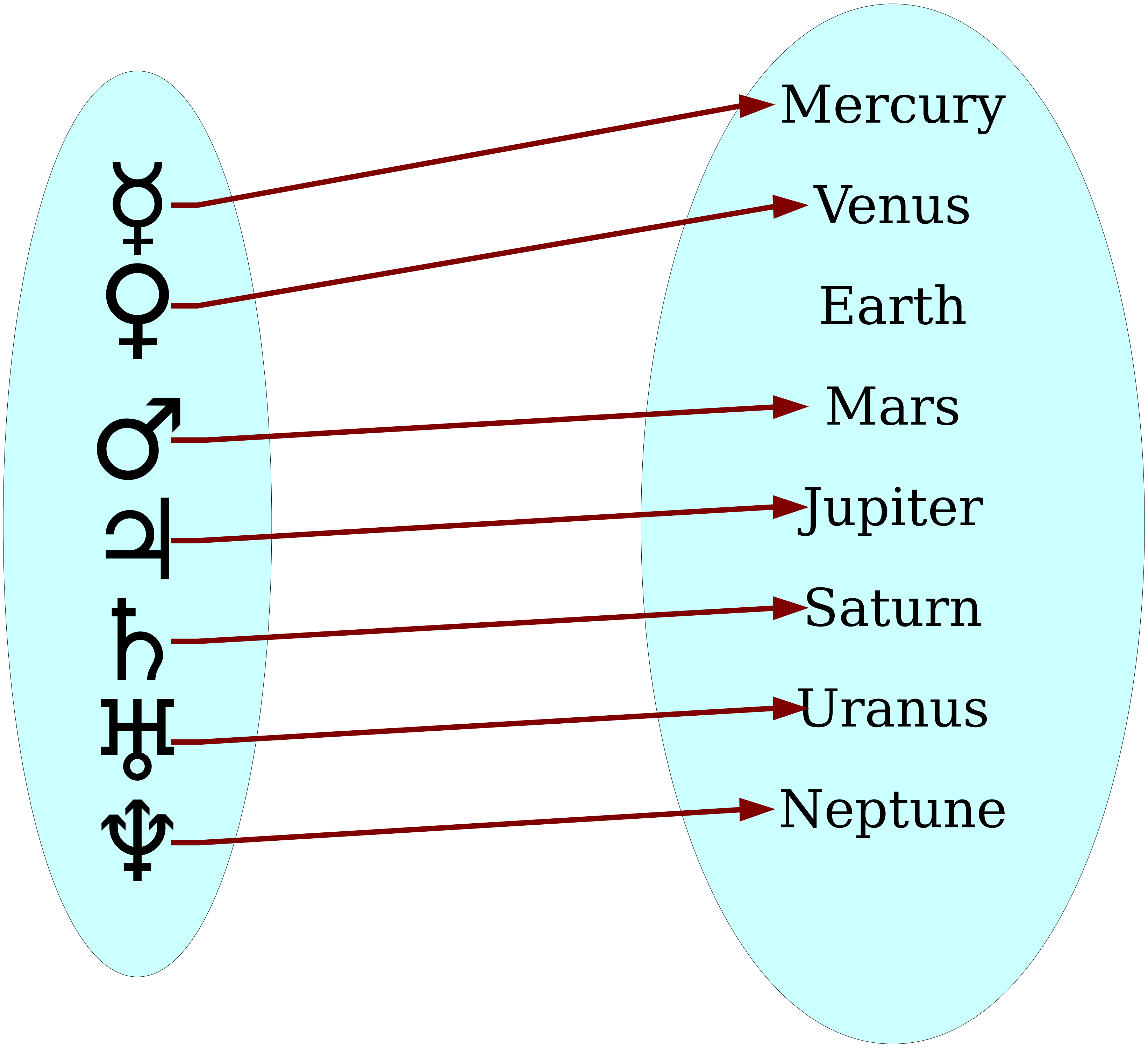
“I can't do anything useful with this.” Check out biocomputation, Code the Change, CS + Social Good, the Kapor Center, Khan Academy, 18F, etc. for places where you can. Also look up Bill Thies, who just got a MacArthur Genius Grant for work in this space.

“I'm scared of CS107.” CS107 is hard. It does not eat kittens. It's a very manageable course provided that you have the right approach.

“I'm going to end up in a cubicle and I'm going to have to work there until I die! Aaaahhh!!” Don't worry! You have a lot of control over what job you get! And avoid “minivanning” – it can really mess with your head!

Back to CS103!

Special Types of Functions


☿
♀
♂
♃
♄
♅
♆
Mercury
Venus
Earth
Mars
Jupiter
Saturn
Uranus
Neptune

Injective Functions

- A function $f : A \to B$ is called ***injective*** (or ***one-to-one***) if the following first-order logic statement is true about f:

$$\forall a_1 \in A.\ \forall a_2 \in A.\ (a_1 \neq a_2 \to f(a_1) \neq f(a_2))$$

(*“If the inputs are different, the outputs are different.”*)

- The following first-order logic definition is equivalent and tends to be more useful in proofs:

$$\forall a_1 \in A.\ \forall a_2 \in A.\ (f(a_1) = f(a_2) \to a_1 = a_2)$$

(*“If the outputs are the same, the inputs are the same.”*)

- A function with this property is called an ***injection***.
- Intuitively, in an injection, every element of the codomain has at most one element of the domain mapping to it.

Injective Functions

Theorem: Let $f : \mathbb{N} \to \mathbb{N}$ be defined as $f(n) = 2n + 7$. Then f is injective.

Proof:

What does it mean for the function f to be injective?

$$\forall n_1 \in \mathbb{N}. \forall n_2 \in \mathbb{N}. (f(n_1) = f(n_2) \to n_1 = n_2)$$

$$\forall n_1 \in \mathbb{N}. \forall n_2 \in \mathbb{N}. (n_1 \neq n_2 \to f(n_1) \neq f(n_2))$$

Therefore, we'll pick arbitrary $n_1, n_2 \in \mathbb{N}$ where $f(n_1) = f(n_2)$, then prove that $n_1 = n_2$.

Injective Functions

Theorem: Let $f : \mathbb{N} \to \mathbb{N}$ be defined as $f(n) = 2n + 7$. Then f is injective.

Proof: Consider any $n_1, n_2 \in \mathbb{N}$ where $f(n_1) = f(n_2)$. We will prove that $n_1 = n_2$.

Since $f(n_1) = f(n_2)$, we see that

$$2n_1 + 7 = 2n_2 + 7.$$

This in turn means that

$$2n_1 = 2n_2,$$

so $n_1 = n_2$, as required. ■

Injective Functions

Theorem: Let $f : \mathbb{Z} \to \mathbb{N}$ be defined as $f(x) = x^4$. Then f is not injective.

Proof:

What does it mean for f to be injective?

$$\forall x_1 \in \mathbb{Z}.\ \forall x_2 \in \mathbb{Z}.\ (x_1 \neq x_2 \to f(x_1) \neq f(x_2))$$

What is the negation of this statement?

$$\neg\forall x_1 \in \mathbb{Z}.\ \forall x_2 \in \mathbb{Z}.\ (x_1 \neq x_2 \to f(x_1) \neq f(x_2))$$
$$\exists x_1 \in \mathbb{Z}.\ \neg\forall x_2 \in \mathbb{Z}.\ (x_1 \neq x_2 \to f(x_1) \neq f(x_2))$$
$$\exists x_1 \in \mathbb{Z}.\ \exists x_2 \in \mathbb{Z}.\ \neg(x_1 \neq x_2 \to f(x_1) \neq f(x_2))$$
$$\exists x_1 \in \mathbb{Z}.\ \exists x_2 \in \mathbb{Z}.\ (x_1 \neq x_2 \land \neg(f(x_1) \neq f(x_2)))$$
$$\exists x_1 \in \mathbb{Z}.\ \exists x_2 \in \mathbb{Z}.\ (x_1 \neq x_2 \land f(x_1) = f(x_2))$$

Therefore, we need to find $x_1, x_2 \in \mathbb{Z}$ such that $x_1 \neq x_2$, but $f(x_1) = f(x_2)$. Can we do that?

Injective Functions

Theorem: Let $f : \mathbb{Z} \to \mathbb{N}$ be defined as $f(x) = x^4$. Then f is not injective.

Proof: We will prove that there exist integers x_1 and x_2 such that $x_1 \neq x_2$, but $f(x_1) = f(x_2)$.

Let $x_1 = -1$ and $x_2 = +1$. Then

$$f(x_1) = f(-1) = (-1)^4 = 1$$

and

$$f(x_2) = f(1) = 1^4 = 1,$$

so $f(x_1) = f(x_2)$ even though $x_1 \neq x_2$, as required. ■

Injections and Composition

Injections and Composition

- ***Theorem:*** If $f : A \to B$ is an injection and $g : B \to C$ is an injection, then the function $g \circ f : A \to C$ is an injection.
- Our goal will be to prove this result. To do so, we're going to have to call back to the formal definitions of injectivity and function composition.

Theorem: If $f : A \to B$ is an injection and $g : B \to C$ is an injection, then the function $g \circ f : A \to C$ is also an injection.

Proof: Let $f : A \to B$ and $g : B \to C$ be arbitrary injections. We will prove that the function $g \circ f : A \to C$ is also injective.

What does it mean for $g \circ f : A \to C$ to be injective?

There are two equivalent definitions, actually!

$$\forall a_1 \in A.\ \forall a_2 \in A.\ ((g \circ f)(a_1) = (g \circ f)(a_2) \to a_1 = a_2)$$

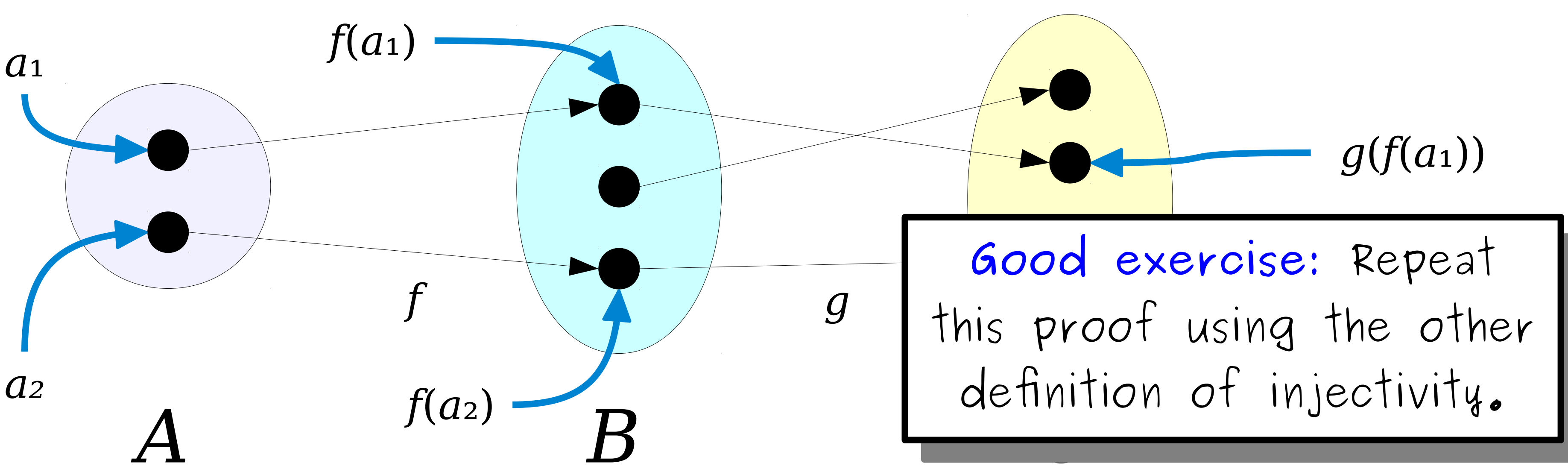
$$\forall a_1 \in A.\ \forall a_2 \in A.\ (a_1 \neq a_2 \to (g \circ f)(a_1) \neq (g \circ f)(a_2))$$

Therefore, we'll choose arbitrary $a_1 \in A$ and $a_2 \in A$ where $a_1 \neq a_2$, then prove that $(g \circ f)(a_1) \neq (g \circ f)(a_2)$.

Theorem: If $f : A \to B$ is an injection and $g : B \to C$ is an injection, then the function $g \circ f : A \to C$ is also an injection.

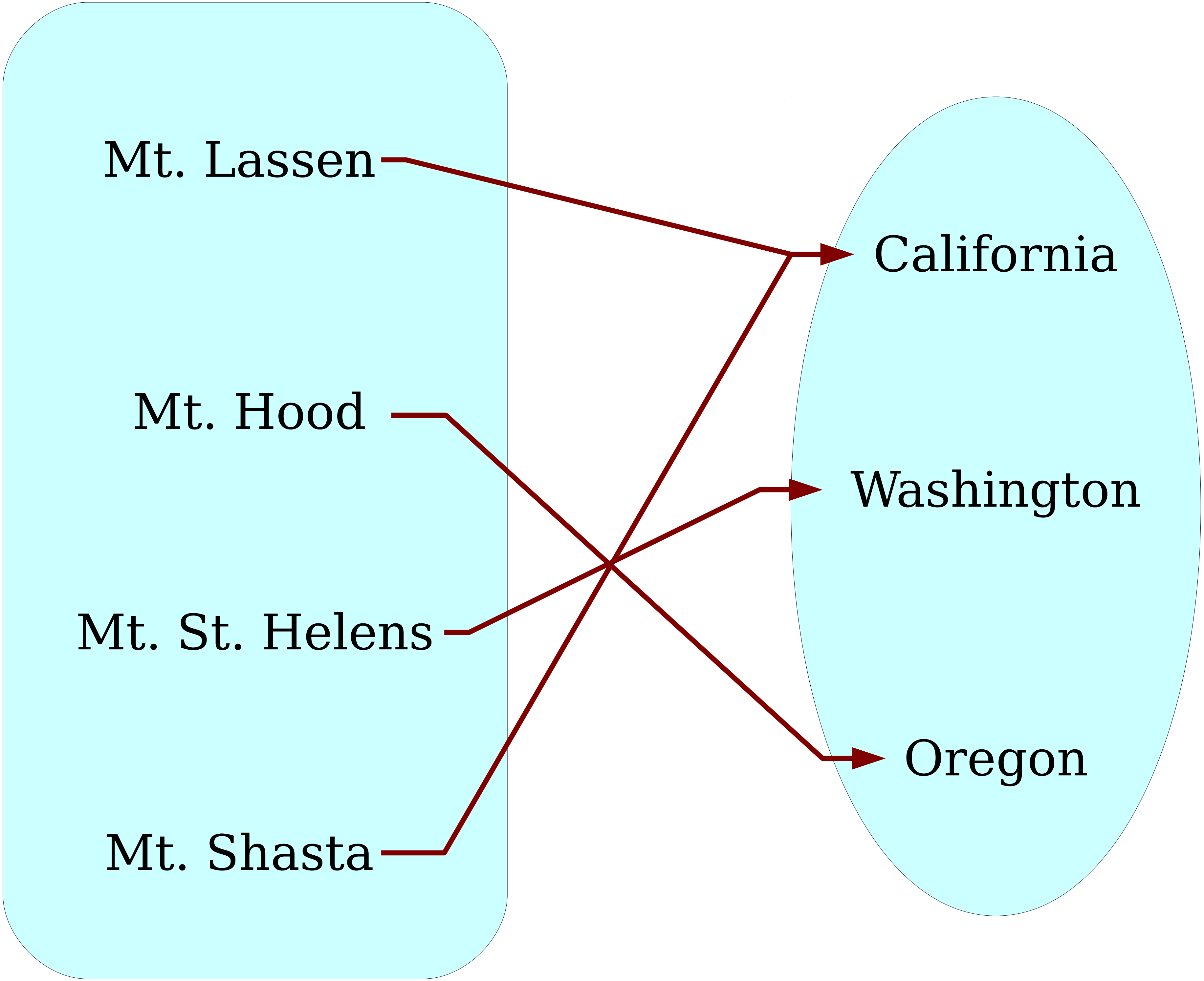
Proof: Let $f : A \to B$ and $g : B \to C$ be arbitrary injections. We will prove that the function $g \circ f : A \to C$ is also injective. To do so, we will prove for all $a_1 \in A$ and $a_2 \in A$ that if $a_1 \neq a_2$, then $(g \circ f)(a_1) \neq (g \circ f)(a_2)$. Equivalently, we need show that if $a_1 \neq a_2$, then $g(f(a_1)) \neq g(f(a_2))$.

Consider any arbitrary $a_1, a_2 \in A$ where $a_1 \neq a_2$. Since f is injective, we know that $f(a_1) \neq f(a_2)$. Similarly, since g is injective, we know that $g(f(a_1)) \neq g(f(a_2))$, as required. ■



Good exercise: Repeat this proof using the other definition of injectivity.

Another Class of Functions


Mt. Lassen
Mt. Hood
Mt. St. Helens
Mt. Shasta
California
Washington
Oregon

Surjective Functions

- A function $f : A \to B$ is called ***surjective*** (or ***onto***) if this first-order logic statement is true about f:

$$\forall b \in B.\ \exists a \in A.\ f(a) = b$$

(*“For every possible output, there's at least one possible input that produces it”*)

- A function with this property is called a ***surjection***.
- Intuitively, every element in the codomain of a surjection has at least one element of the domain mapping to it.

Surjective Functions

Theorem: Let $f : \mathbb{R} \to \mathbb{R}$ be defined as $f(x) = x / 2$. Then $f(x)$ is surjective.

Proof:

What does it mean for f to be surjective?

$$\forall y \in \mathbb{R}.\ \exists x \in \mathbb{R}.\ f(x) = y$$

Therefore, we'll choose an arbitrary $y \in \mathbb{R}$, then prove that there is some $x \in \mathbb{R}$ where $f(x) = y$.

Surjective Functions

Theorem: Let $f : \mathbb{R} \to \mathbb{R}$ be defined as $f(x) = x / 2$. Then $f(x)$ is surjective.

Proof: Consider any $y \in \mathbb{R}$. We will prove that there is a choice of $x \in \mathbb{R}$ such that $f(x) = y$.

Let $x = 2y$. Then we see that

$$f(x) = f(2y) = 2y / 2 = y.$$

So $f(x) = y$, as required. ■

Composing Surjections

Theorem: If $f : A \to B$ is surjective and $g : B \to C$ is surjective, then $g \circ f : A \to C$ is also surjective.

Proof: Let $f : A \to B$ and $g : B \to C$ be arbitrary surjections. We will prove that the function $g \circ f : A \to C$ is also surjective.

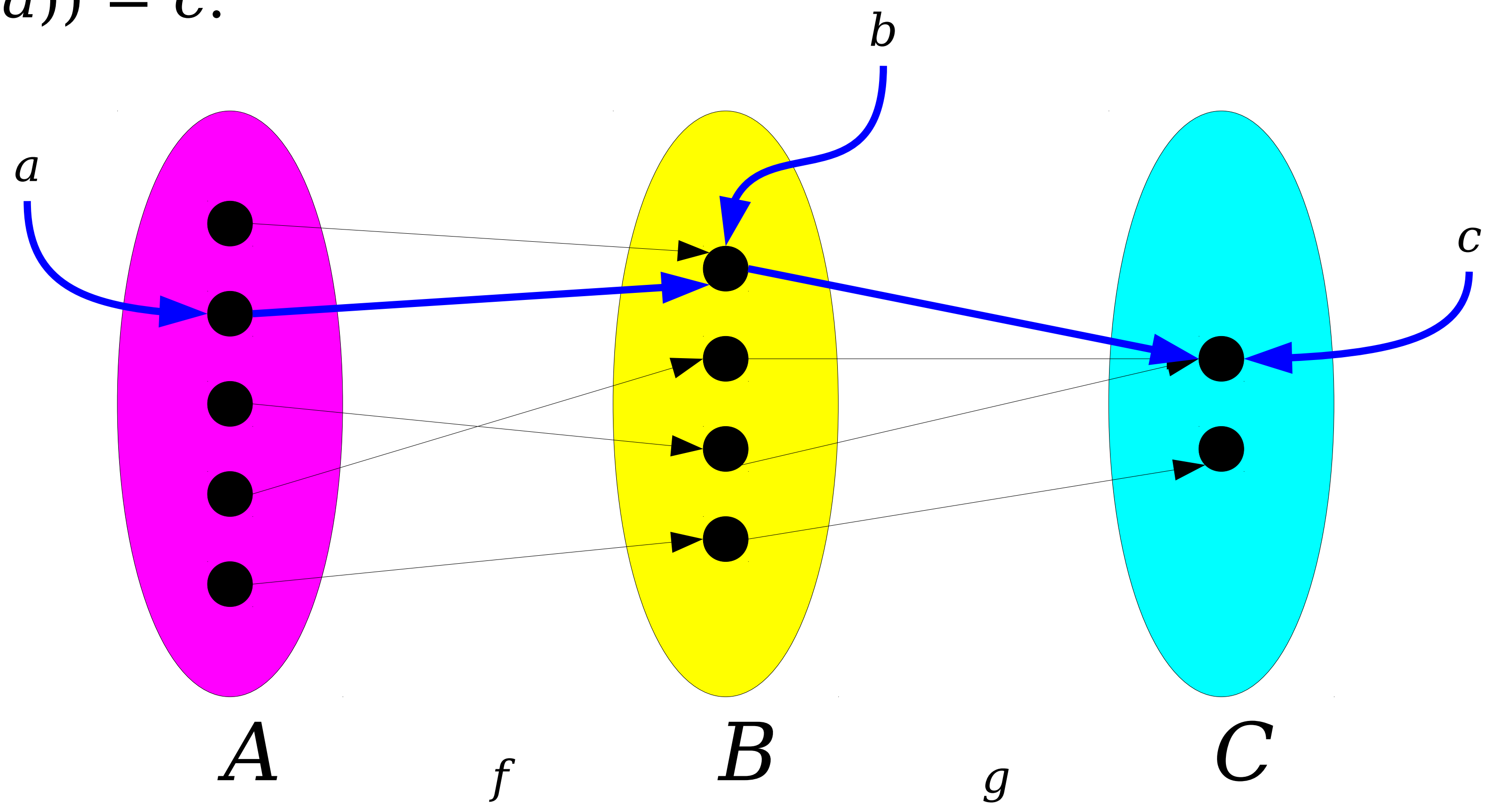
What does it mean for $g \circ f : A \to C$ to be surjective?

$$\forall c \in C.\ \exists a \in A.\ (g \circ f)(a) = c$$

Therefore, we'll choose arbitrary $c \in C$ and prove that there is some $a \in A$ such that $(g \circ f)(a) = c$.

Theorem: If $f : A \to B$ is surjective and $g : B \to C$ is surjective, then $g \circ f : A \to C$ is also surjective.

Proof: Let $f : A \to B$ and $g : B \to C$ be arbitrary surjections. We will prove that the function $g \circ f : A \to C$ is also surjective. To do so, we will prove that for any $c \in C$, there is some $a \in A$ such that $(g \circ f)(a) = c$. Equivalently, we will prove that for any $c \in C$, there is some $a \in A$ such that $g(f(a)) = c$.



Theorem: If $f : A \to B$ is surjective and $g : B \to C$ is surjective, then $g \circ f : A \to C$ is also surjective.

Proof: Let $f : A \to B$ and $g : B \to C$ be arbitrary surjections. We will prove that the function $g \circ f : A \to C$ is also surjective. To do so, we will prove that for any $c \in C$, there is some $a \in A$ such that $(g \circ f)(a) = c$. Equivalently, we will prove that for any $c \in C$, there is some $a \in A$ such that $g(f(a)) = c$.

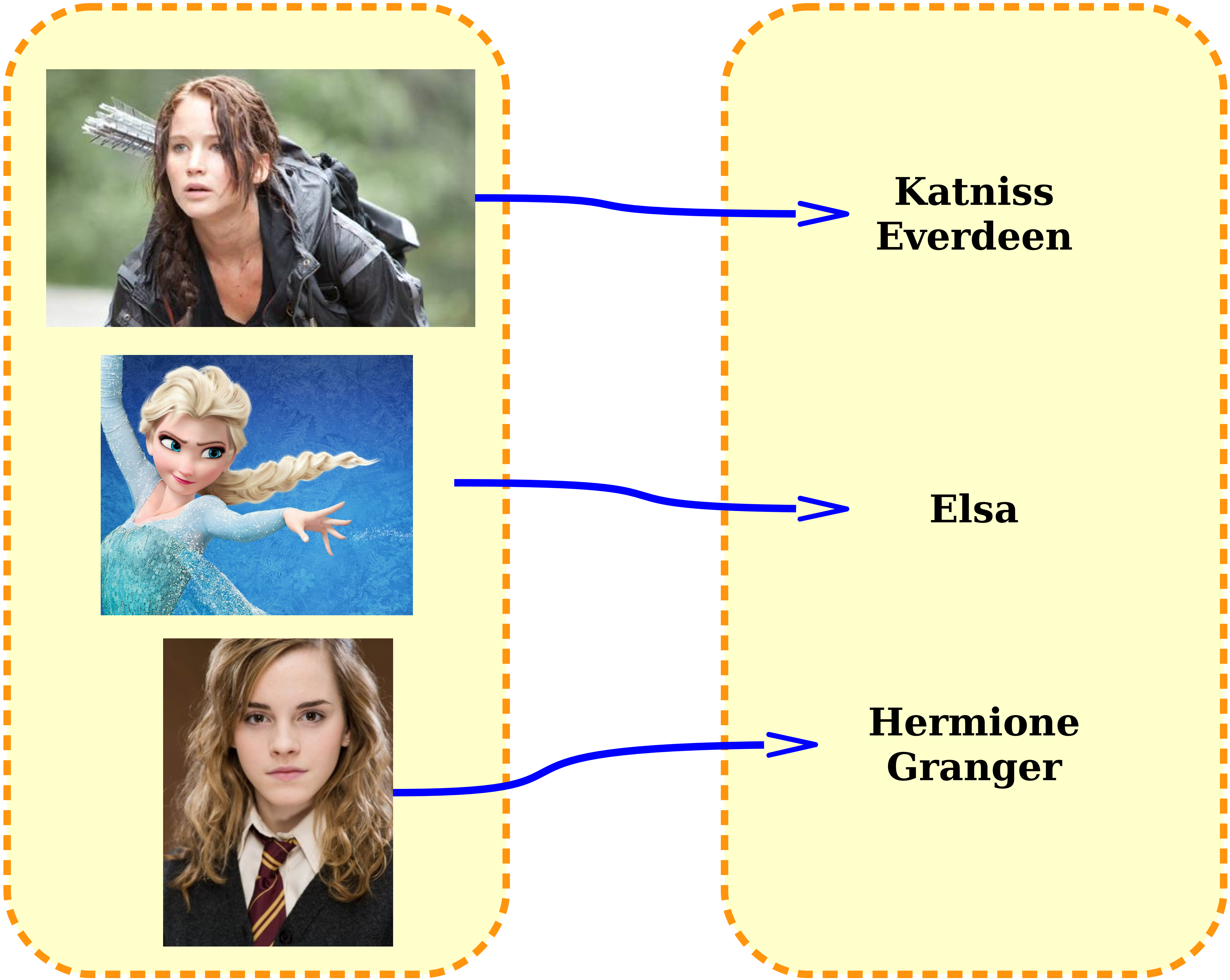
Consider any $c \in C$. Since $g : B \to C$ is surjective, there is some $b \in B$ such that $g(b) = c$. Similarly, since $f : A \to B$ is surjective, there is some $a \in A$ such that $f(a) = b$. This means that there is some $a \in A$ such that

$$g(f(a)) = g(b) = c,$$

which is what we needed to show. ■

Injections and Surjections

- An injective function associates ***at most*** one element of the domain with each element of the codomain.
- A surjective function associates ***at least*** one element of the domain with each element of the codomain.
- What about functions that associate ***exactly one*** element of the domain with each element of the codomain?


Katniss Everdeen
Elsa
Hermione Granger

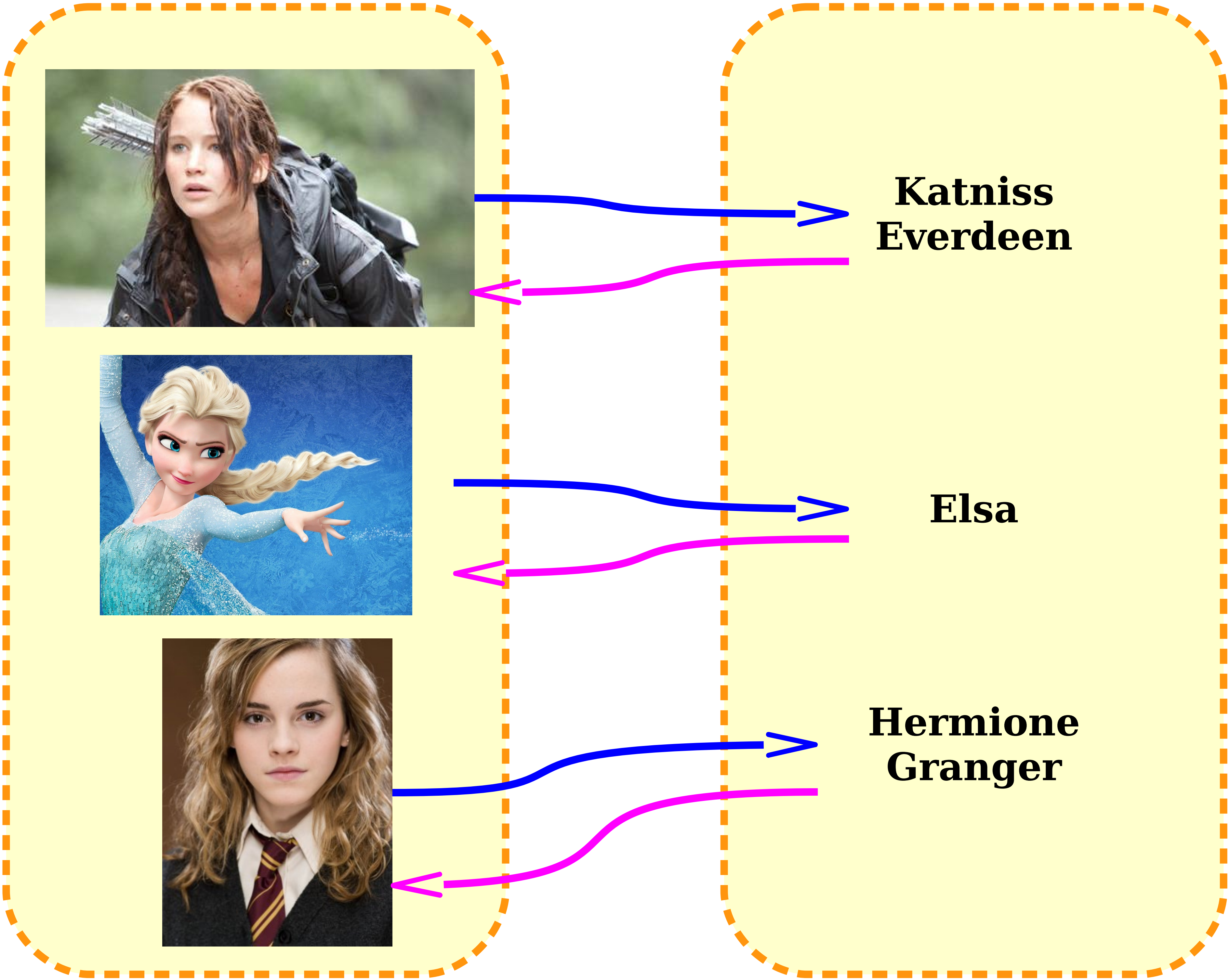
Bijections

- A function that associates each element of the codomain with a unique element of the domain is called ***bijective***.
  - Such a function is a ***bijection***.
- Formally, a bijection is a function that is both *injective* and *surjective*.
- Bijections are sometimes called ***one-to-one correspondences***.
  - Not to be confused with "one-to-one functions."

Bijections and Composition

- Suppose that $f : A \to B$ and $g : B \to C$ are bijections.
- Is $g \circ f$ necessarily a bijection?
- ***Yes!***
  - Since both f and g are injective, we know that $g \circ f$ is injective.
  - Since both f and g are surjective, we know that $g \circ f$ is surjective.
  - Therefore, $g \circ f$ is a bijection.

Inverse Functions


Katniss Everdeen
Elsa
Hermione Granger

Inverse Functions

- In some cases, it's possible to "turn a function around."
- Let $f : A \to B$ be a function. A function $f^{-1} : B \to A$ is called an ***inverse of f*** if the following statements are true:

$$\forall a \in A.\ f^{-1}(f(a)) = a \qquad \forall b \in B.\ f(f^{-1}(b)) = b$$

- In other words, if f maps a to b, then f^{-1} maps b back to a and vice-versa.
- Not all functions have inverses (we just saw a few examples of functions with no inverse).
- If f is a function that has an inverse, then we say that f is ***invertible***.

Inverse Functions

- ***Theorem:*** Let $f : A \to B$. Then f is invertible if and only if f is a bijection.
- A proof of this result is in the course reader – it's not required that you know it, but if you're curious, check it out!

Where We Are

- We now know
  - what an injection, surjection, and bijection are;
  - that the composition of two injections, surjections, or bijections is also an injection, surjection, or bijection, respectively; and
  - that bijections are invertible and invertible functions are bijections.
- You might wonder why this all matters. Well, there's a good reason…

Next Time

- ***Cardinality, Formally Speaking***
  - How do we rigorously define set cardinalities?
- ***Cantor's Theorem Revisited***
  - A formal proof of Cantor's theorem!