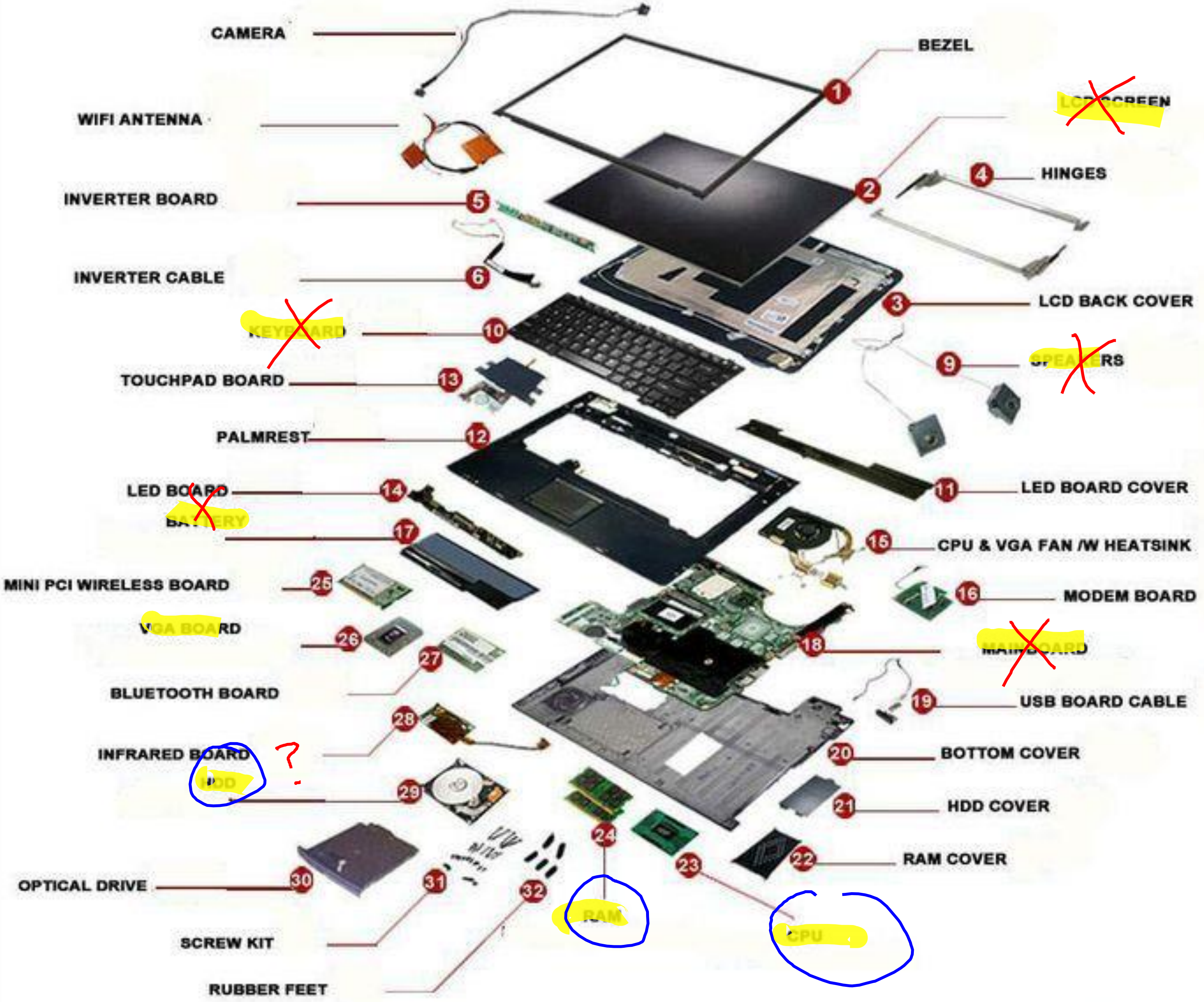
CS3102 Theory of Computation

www.cs.virginia.edu/~njb2b/cstheory/s2020

Warm up:

1. What piece(s) of a computer actually make it compute?
2. Is there one responsible component or many?
3. If there are many, which is/are most important?



CAMERA
BEZEL
1
WIFI ANTENNA
LCD SCREEN
2
HINGES
4
INVERTER BOARD
5
INVERTER CABLE
6
LCD BACK COVER
3
KEYBOARD
10
SPEAKERS
9
TOUCHPAD BOARD
13
PALMREST
12
LED BOARD
14
LED BOARD COVER
11
BATTERY
17
CPU & VGA FAN /W HEATSINK
15
MINI PCI WIRELESS BOARD
25
MODEM BOARD
16
VGA BOARD
26
MAINBOARD
18
27
BLUETOOTH BOARD
USB BOARD CABLE
19
28
INFRARED BOARD
?
BOTTOM COVER
20
HDD
29
HDD COVER
21
RAM COVER
22
24
23
OPTICAL DRIVE
30
31
32
RAM
CPU
SCREW KIT
RUBBER FEET

Most important parts (according to Nate)

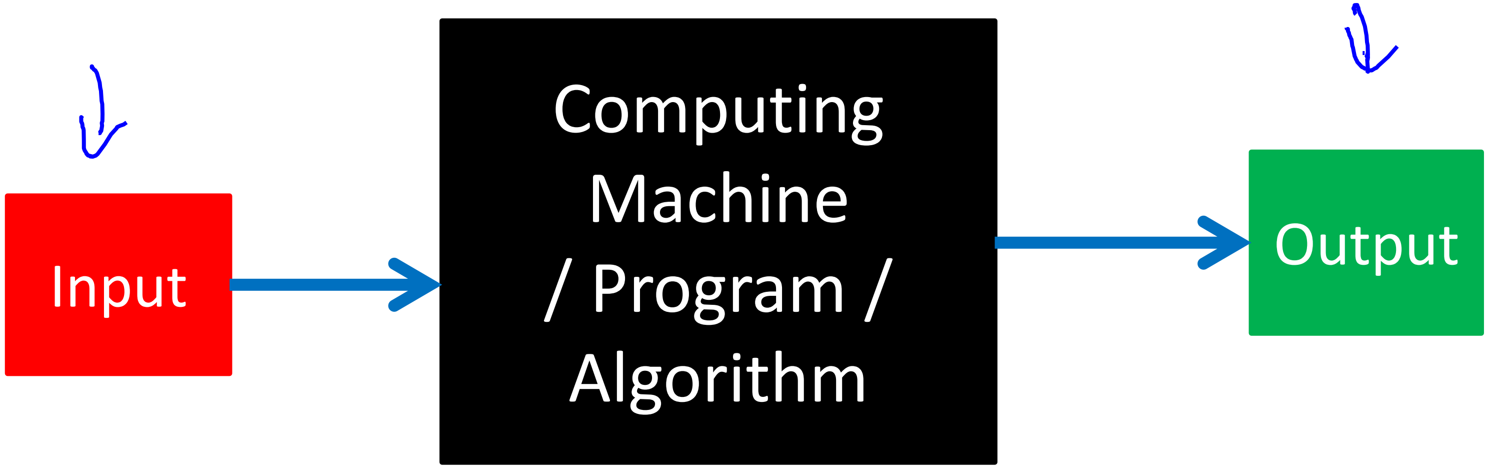
- CPU "work"
  - Circuits of transistors
- RAM ← storing some "smallish" # of things
- HDD/SSD ← stores data, and lots of it

Agenda

- Last class:
  - History, Motive, Logistics
- Today:
  - Breaking down a computer, building up to models of computing
- Upcoming deadlines:
  - Today: Exercise 0-1, Course registration survey
  - Tuesday: Exercise 0-2, Getting Started with LaTeX

What does it mean to compute?

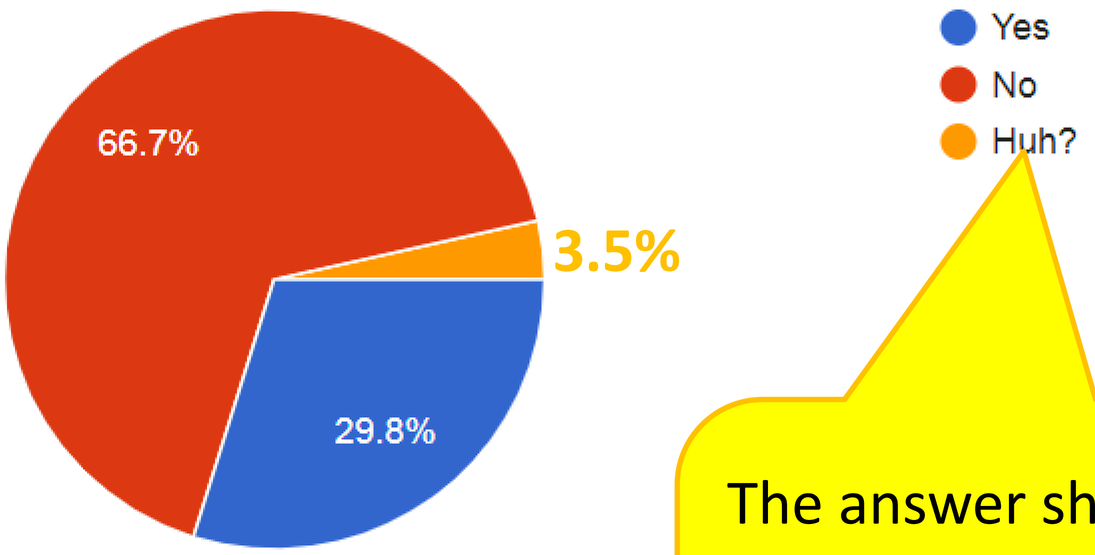
- We’ll discuss several ideas this semester
- Several “models” of computing
- Vague idea: take and input and produce an output



CS: Yes

Is 0 a Natural Number?

114 responses



The answer should follow from the definition of “Natural Number”, but there isn’t one universally accepted definition and you haven’t told us which one to use!

What makes a good definition?

- Non-recursive
- Unambiguous
  - There is only one interpretation
- Small # of assumptions
- All things match or don’t
- Self-explanatory:
  - Understandable for my audience

Defining Natural Numbers

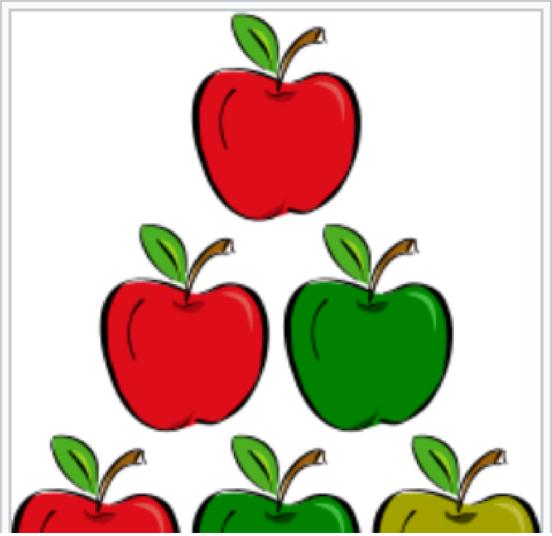
Natural number

From Wikipedia, the free encyclopedia

This article is about "positive integers" and "non-negative integers". For all the numbers ..., −2, −1, 0, 1, 2, ..., see Integer.

"ℕ" redirects here. For the cryptocurrency, see Namecoin.

In mathematics, the **natural numbers** are those used for counting (as in "there are *six* coins on the table") and ordering (as in "this is the *third* largest city in the country"). In common mathematical terminology, words colloquially used for counting are "cardinal numbers" and words connected to ordering represent "ordinal numbers". The natural numbers can, at times, appear as a convenient set of codes (labels or "names"); that is, as what linguists call nominal numbers, forgoing many or all of the properties of being a number in a mathematical sense.



Some definitions, including the standard ISO 80000-2,[1][2] begin the natural numbers with 0, corresponding to the **non-negative integers**

Defining Natural Numbers

MCS (The textbook many of you used in cs2102)

~~Here the symbol ∀ is read "for all."~~ The symbol $\mathbb{N}$ stands for the set of *nonnegative integers*: 0, 1, 2, 3, ... (ask your instructor for the complete list). ~~The symbol "∈"~~

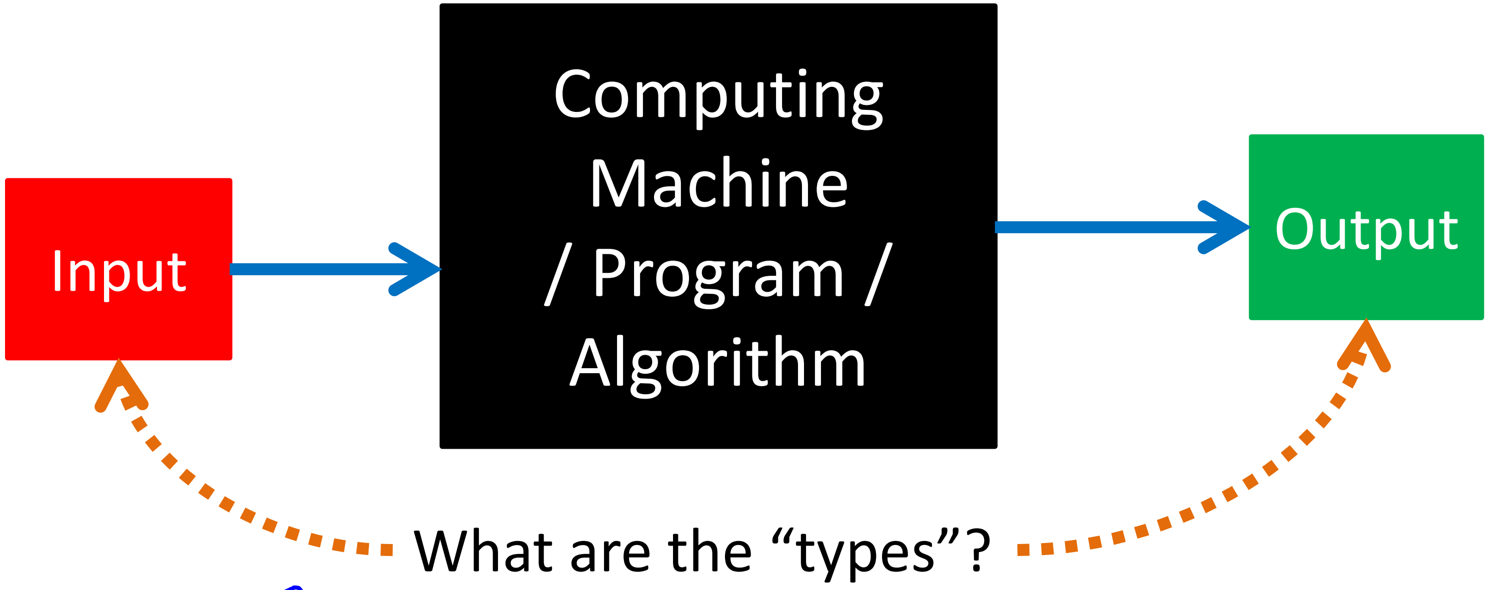
TCS (our textbook)

There are several sets that we will use in this book time and again. The set

$$\mathbb{N} = \{0, 1, 2, \ldots\} \qquad (1.3)$$

contains all *natural numbers*, i.e., non-negative integers. For any natural

Defining Our Input/Output



ON COMPUTABLE NUMBERS, WITH AN APPLICATION TO THE ENTSCHEIDUNGSPROBLEM

By A. M. Turing.

What we compute on: representations of things (e.g. numbers)

What do we compute on?

Finite

- **String**: an ordered sequence of characters
- Is a representation of something
- Characters come from an alphabet

- Let's formally define them

Alphabet

- Alphabet is a finite set of characters
  - Notation: Σ (\Sigma in LaTeX)
  - Examples: $\{0,1\}$, $\{a, b, c, d, \dots, z\}$, $\{0,1,2, \dots, 9\}$

Binary

Sesame St. alphabet

decimal

String

$2 \cdot 2 \cdot 2$ $\quad 2^3$

- The set Σ^n refers to the set of all length n strings over alphabet Σ
- $\{0,1\}^3$ is the set of a 3-bit strings
  - $(0, 1, 0)$
  - $\{0,1\}^3 = \{(x_0, x_1, x_2) : x_0, x_1, x_2 \in \{0,1\}\}$
  - We write $x_0x_1x_2$ instead of (x_0, x_1, x_2)
  - $\{0,1\}^3 = \{000,001,010,011,100,101,110,111\}$
- If $|\Sigma| = m$, then $|\Sigma^n| =$? $\quad \cancel{n^m} \quad m^n$

Kleene Star Operator

- Σ^* refers to the set of all strings over the alphabet Σ (of any length, including 0)

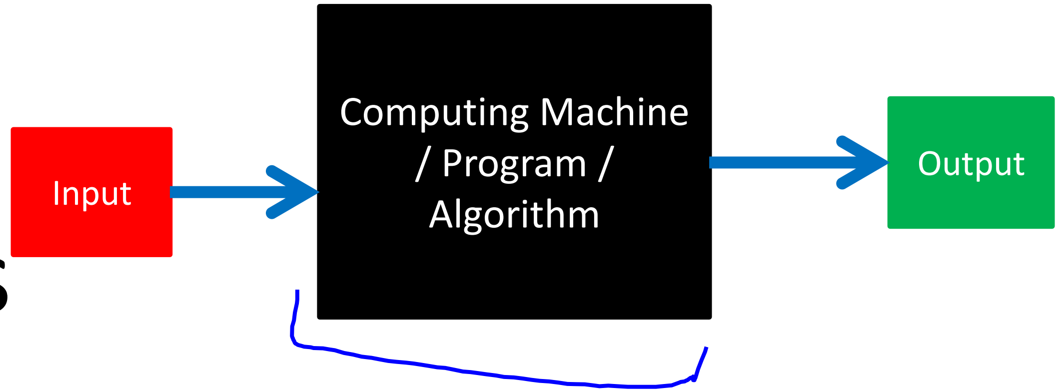
λ

ε

∈

- $\boxed{\{0,1\}^*} = \{(x_0, x_1, \dots, x_{n-1}) : n \in \mathbb{N},\ x_0, \dots, x_{n-1} \in \{0,1\}\}$
- $\{0,1\}^* = \{"", 0, 1, 00, 01, 10, 11, 000, 001, \dots\}$
- $\{0,1\}^* = \{0,1\}^0 \cup \{0,1\}^1 \cup \{0,1\}^2 \cup \cdots$ \cup
- $\{0,1\}^* = \bigcup_{n\in\mathbb{N}} \{0,1\}^n$

What do we compute, then?

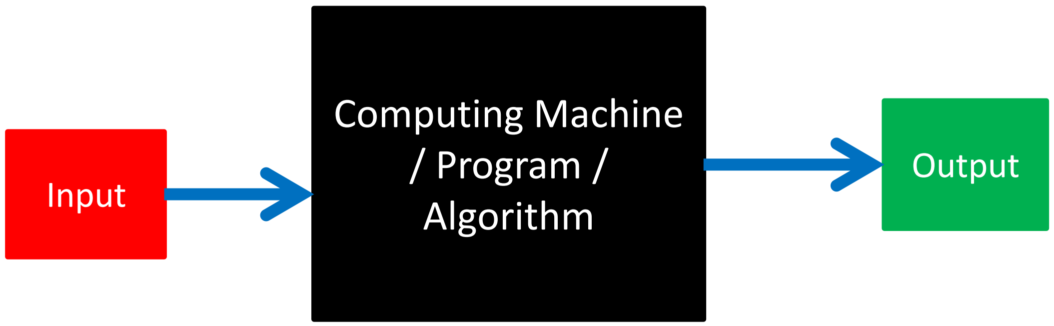


- Input and output are strings
- Black box is an implementation
- What are we implementing? function
  - Functions
  - Languages * later

Functions

- Function: a "mapping" from input to output
- $f: D \to C$ $\quad f: \Sigma^* \to \Sigma^*$
  - Function f maps elements from the set D to an element from the set C
  - D: the domain of f
  - C: the co-domain of f
  - Range/image of f: $\{f(d): d \in D\}$
    - The elements of C that are "mapped to" by something
- Finite function: $f: D \to C$ is a finite function if D is finite
  - Otherwise it's an infinite function

Computing a Function

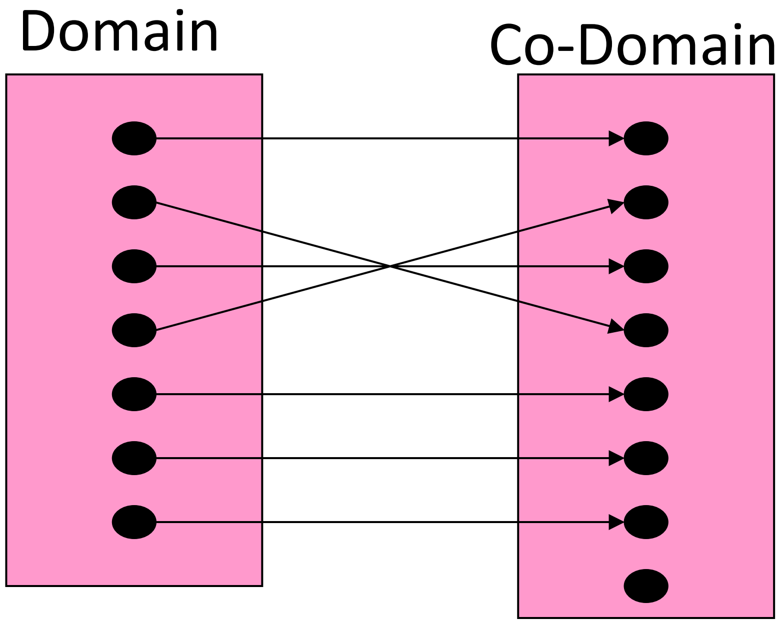


- A function f is computable under a computing model if:
- That model allows for an implementation (way of filling in the black box) such that,
  - For any input $x \in D$ (string representing an element from the domain of f)
  - The implementation "produces" the correct output

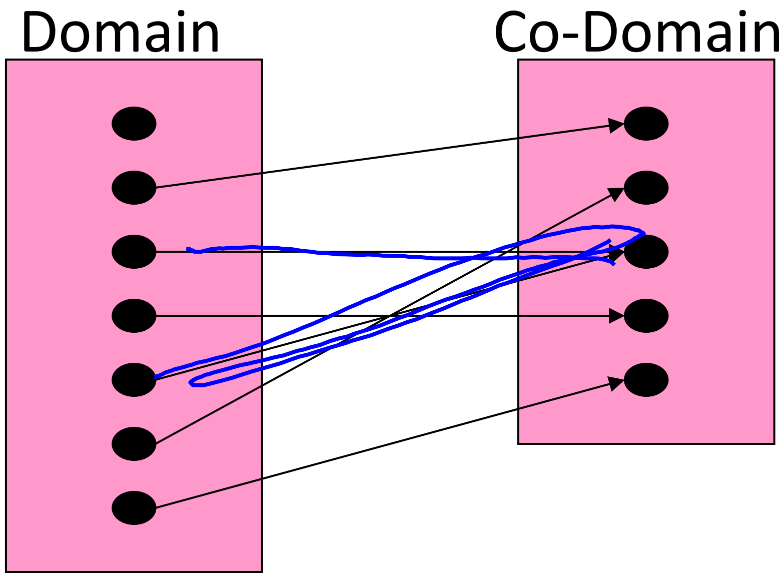
Properties of Functions

- One-to-one (injective)
  - $x \neq y \Rightarrow f(x) \neq f(y)$
  - Different inputs yield different outputs
  - No two inputs share an output

1-1, Injective Functions



INJECTIVE FUNCTION



NON-INJECTIVE FUNCTION

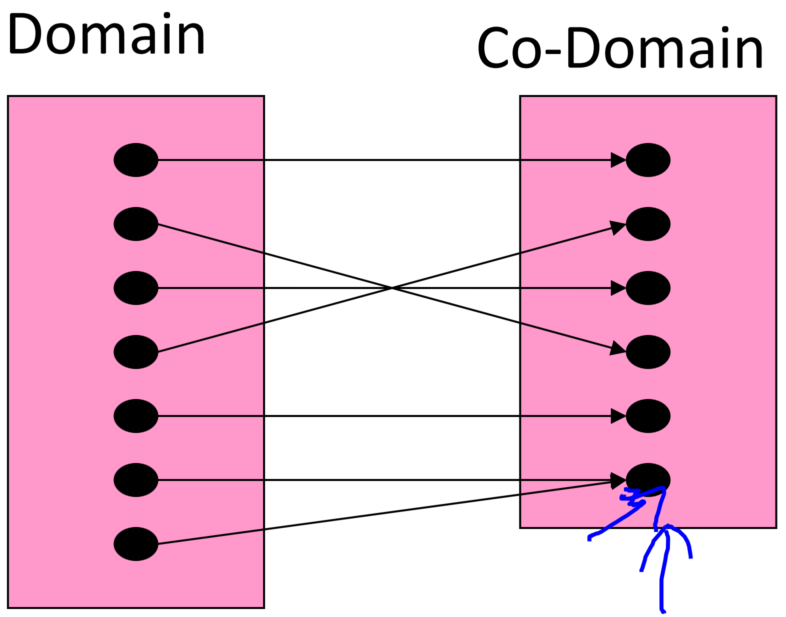
Nothing in Co-Domain “receives” two things

$$|D| \leq |C|$$

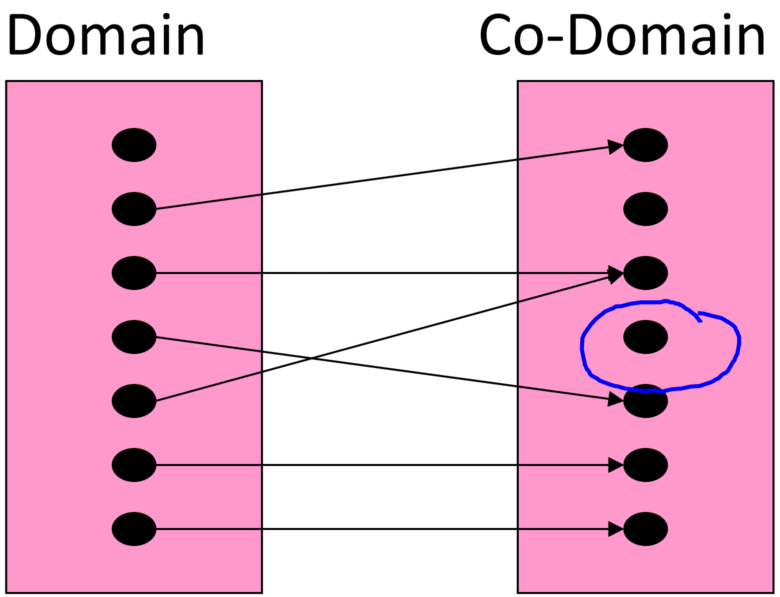
Properties of Functions

- One-to-one (injective)
  - $x \neq y \Rightarrow f(x) \neq f(y)$
- Onto (surjective)
  - $\forall c \in C, \exists d \in D : f(d) = c$
  - Everything in C is the output of something in d

Onto, Surjective Functions



SURJECTIVE FUNCTION



NON-SURJECTIVE FUNCTION

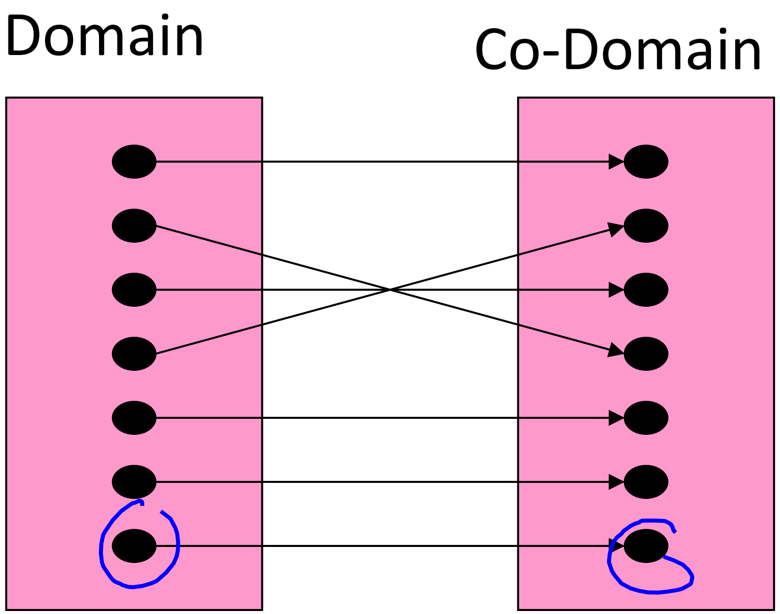
Everything in Co-Domain “receives” something

$$|D| \geq |C|$$

Properties of Functions

- One-to-one (injective)
  - $x \neq y \Rightarrow f(x) \neq f(y)$
- Onto (surjective)
  - $\forall c \in C, \exists d \in D : f(d) = c$
- One-to-one Correspondance (bijective)
  - Both one-to-one and injective
  - Everything in C is mapped to by a unique element in D
  - All elements from domain and co-domain are perfectly "partnered"

Bijective Functions



BIJECTIVE FUNCTION

Because Onto:
Everything in Co-Domain "receives" something

$$|D| \geq |C|$$

Because 1-1:
Nothing in Co-Domain "receives" two things

$$|D| \leq |C|$$

Conclusion:
Things in the Domain exactly "partner" with things in Co-Domain

$$|D| = |C|$$

Cardinality

- The number of elements in a set
- Two sets have the same cardinality if there is a bijection between them
- What does it mean for a set to have cardinality 5?
  - It has a bijection with the set $[5] = \{0,1,2,3,4\}$
- A finite set has cardinality k if it has a bijection with the set $[k] = \{n : n \in \mathbb{N} \land n < k\}$
- An infinite set has no bijections with any set $[k]$ for $k \in \mathbb{N}$

Are all functions computable?

- How could we approach this question?

Implementing a Function

- Examples of ways to implement a function:

- Properties we want of implementations:

Next Time

- For any “reasonable” model of computing, there will be some uncomputable functions