# CS3102 Theory of Computation

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

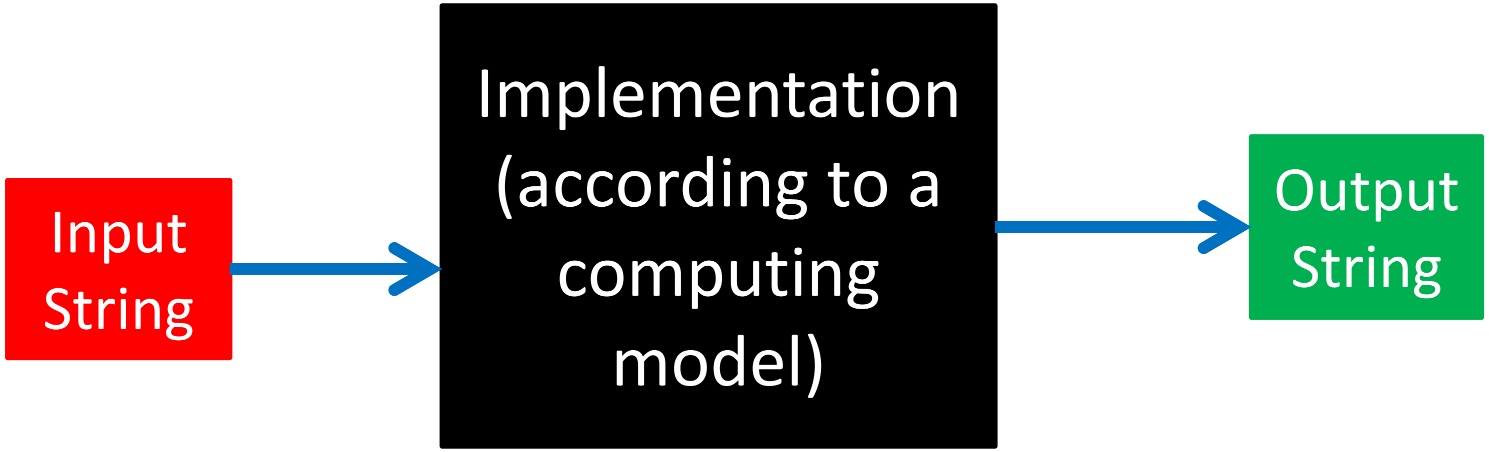
Warm up:

1. What are examples of ways to implement functions?
2. What properties should implementations have?

# Logistics

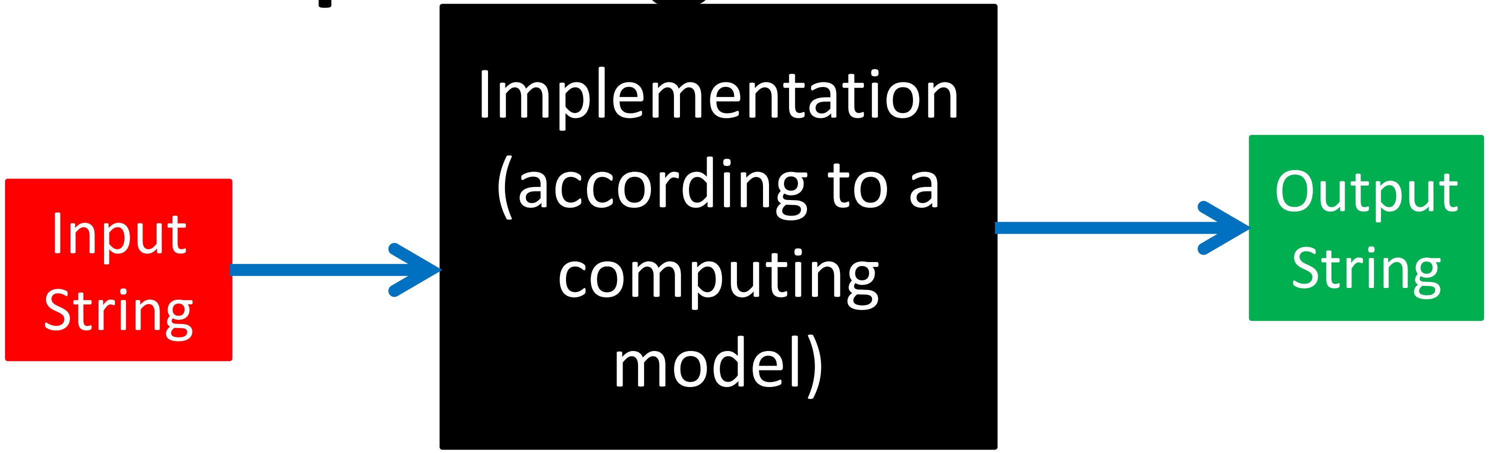
- TA Office hours starting this week (see webpage soon for when/where)
- Course registration survey was due Thursday
  - Didn’t complete it? No problem! Just do it soon
- Exercise 0_2 due today
  - Didn’t complete it? No problem (this time)! Just request an extension
- Exercise 0_3 due Thursday
  - Pick one of python/java
  - Decided to enroll late and need to catch up? No problem! Just request an extension if you can’t get it in on time
- Exercise due/release dates will be more “batched” going forward, staggered this time to test out the submission system
- First Quiz
  - Released Friday, due Tuesday

# Last Class




- What is a String?
  - A finite sequence of characters (an element of $\Sigma^*$)
- What are we implementing?
  - A function mapping strings to strings ($\Sigma^* \rightarrow \Sigma^*$)

# Computing a Function




- To define a computing model (which will implement functions):
  - Define how to receive an input
  - Define how to produce an output
  - Define the steps taken to convert input into output

# Implementing a Function

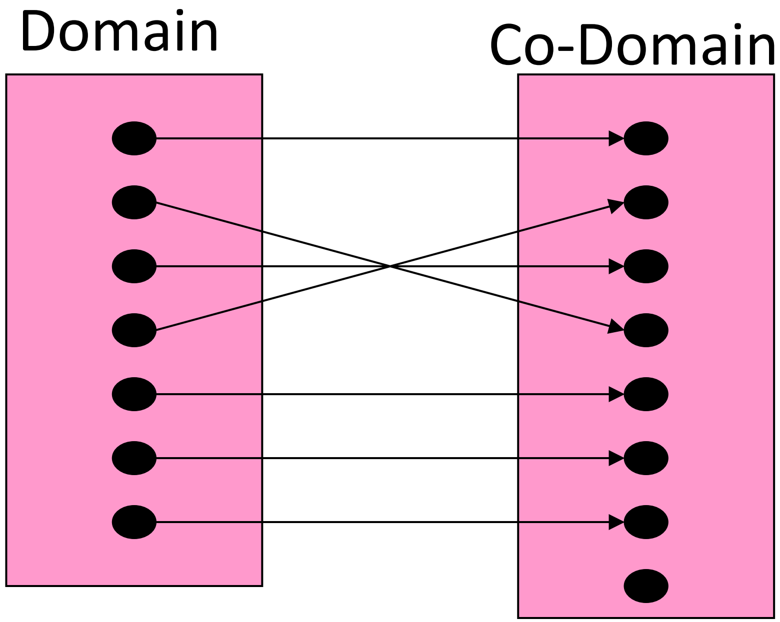
- Examples of ways to implement a function:

- Properties we want of implementations:

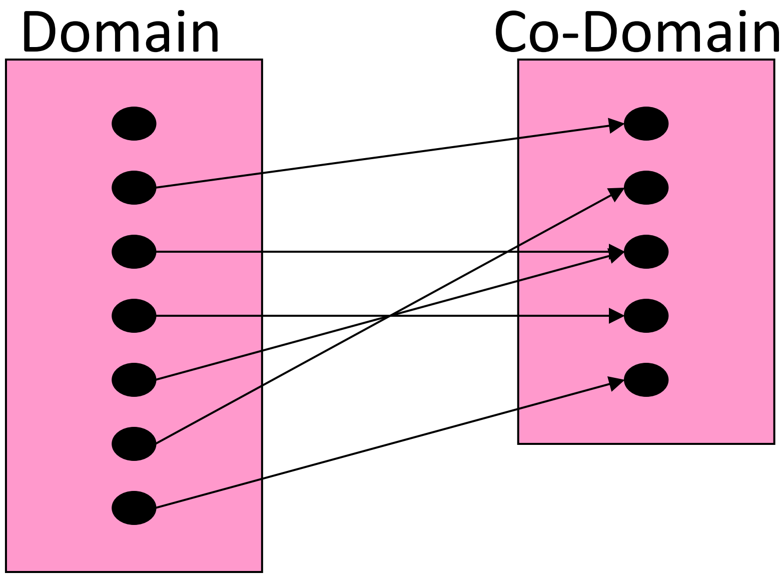
# Are all functions computable?

- How could we approach this question?

# 1-1, Injective Functions




INJECTIVE FUNCTION

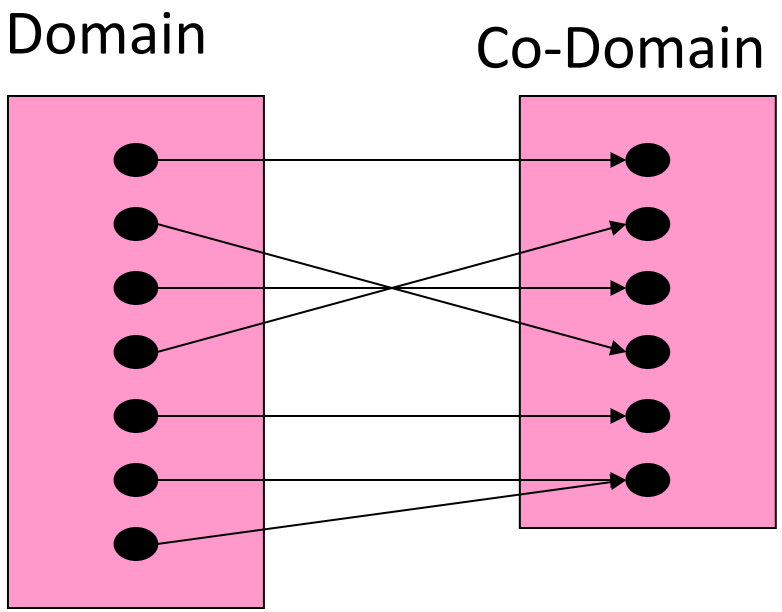



NON-INJECTIVE  FUNCTION

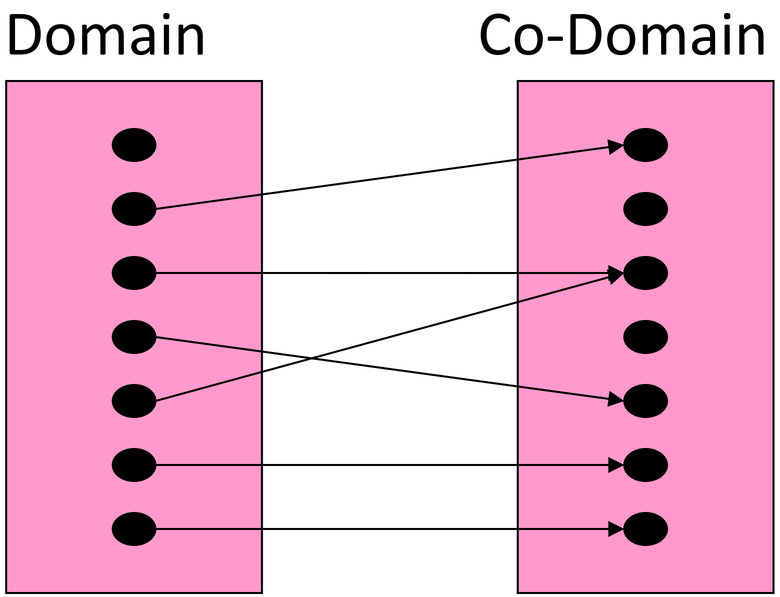
Nothing in Co-Domain “receives” two things

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

# Onto, Surjective Functions




SURJECTIVE FUNCTION

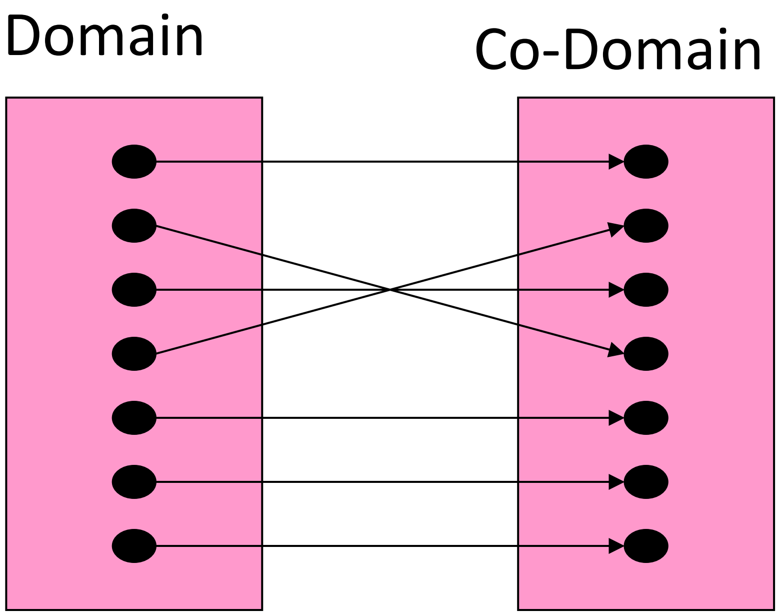



NON-SURJECTIVE FUNCTION

Everything in Co-Domain “receives” something

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

# 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

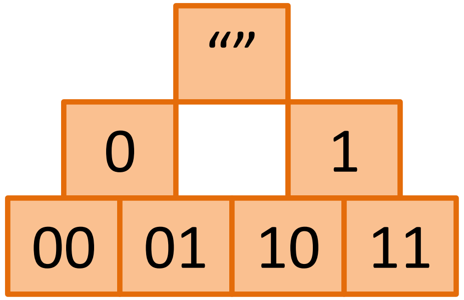
$$|D| = |C|$$

# 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 $k \in \mathbb{N}$

# How many length-$n$ binary strings

- How many binary strings are there of length $n$?
  - $|\{0,1\}^n| = 2^n$
- How to prove this?
  - Induction
  - Bijection with $[2^n] = \{0,1,\ldots,2^{n-1}\}$

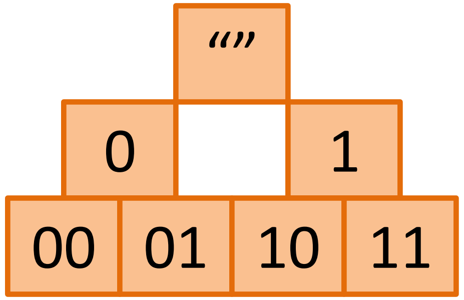


# Principle of Induction

- If something is true for $x = 0$ (base case),
- And when it's true for $x = n$ (inductive hypothesis), it must be true for $x = n + 1$ (inductive step)
- It must be that it's true for all $x \in \mathbb{N}$

# $|\{0,1\}^n| = 2^n$ via induction

- Base case: $|\{0,1\}^0| = 1$
  - Proof: $\{0,1\}^0 = \{""\}$
- Inductive hypothesis:
  - $|\{0,1\}^{n-1}| = 2^{n-1}$
- Inductive step:
  - For each binary string of length $n-1$, we can make two different binary strings of length $n$ by concatenating either a 0 or a 1.
  - Therefore $|\{0,1\}^n| = 2 \cdot |\{0,1\}^{n-1}| = 2 \cdot 2^{n-1} = 2^n$



# $|\{0,1\}^n| = 2^n$ via bijection

- Proof idea:
  - Find a bijection $f_n: \{0,1\}^n \leftrightarrow [2^n]$

| $10^5$ | $10^4$ | $10^3$ | $10^2$ | $10^1$ | $10^0$ |
|---|---|---|---|---|---|
| 3 | 0 | 1 | 2 | 2 | 0 |

| $2^5$ | $2^4$ | $2^3$ | $2^2$ | $2^1$ | $2^0$ |
|---|---|---|---|---|---|
| 0 | 0 | 1 | 1 | 0 | 1 |

- Given $b \in \{0,1\}^n$, what is $f_n(b) \in [2^n]$?
  - $f_n(b) = \sum_{i=0}^{n-1} b_i \cdot 2^i$
  - E.g. $1101 = 1 \cdot 2^0 + 0 \cdot 2^1 + 1 \cdot 2^2 + 1 \cdot 2^3 = 13$

- Given $x \in [2^n]$, what is $f_n^{-1}(x) \in \{0,1\}^n$?
  - Do $n$ times: if $x$ is even, make $0$ the next bit of $b$, otherwise make it 1. Let $x = \left\lfloor \frac{x}{2} \right\rfloor$

# Calculating binary of 13

- $13$ is odd, so last bit is $1$
  - $x = \left\lfloor \frac{13}{2} \right\rfloor = 6$
- 6 is even, so next bit is $0$
  - $x = \left\lfloor \frac{6}{2} \right\rfloor = 3$
- 3 is odd, so next bit is 1
  - $x = \left\lfloor \frac{3}{2} \right\rfloor = 1$
- 1 is odd, so next bit is 1

$b =$

| | | | |
|---|---|---|---|
| 1 | 1 | 0 | 1 |

# How many binary strings of any length?

- If we don’t limit the length, how many strings are there?
  - $\infty$?
- What naturals can/can’t we represent
- What does it mean to “represent”?
  - A surjective mapping from a set of strings onto a set
  - Ideally a bijection, but not necessary
- Are there things we can’t represent?

# Representing $\mathbb{N}$ with binary strings

- For a binary string $b$, if $f_n(b) = x$, then $f_n(0b) = x$
  - Leading zeros don't change the value
  - Our procedure above gives an onto mapping from binary strings to the natural numbers
  - We can represent all natural numbers with binary strings

# Countablility and Uncountability

- A set $S$ is countable if $|S| \leq |\mathbb{N}|$
  - If $|S| = |\mathbb{N}|$, then $S$ is "countably infinite"
- A set $S$ is countable if there is an onto (surjective) function from $\mathbb{N}$ to $S$

# $\{0,1\}^*$ is countable

- We showed $|\{0,1\}^*| \geq |\mathbb{N}|$
- Countable if $|\{0,1\}^*| \leq |\mathbb{N}|$
- Need to "represent" strings with naturals
- Idea: build a "list" of all strings, represent each string by its index in that list

# Listing all strings

- $\{0,1\}^0 = \{\underset{0}{""}\}$
- $\{0,1\}^1 = \{\underset{1}{0},\underset{2}{1}\}$
- $\{0,1\}^2 = \{\underset{3}{00},\underset{4}{01},\underset{5}{10},\underset{6}{11}\}$
- $\{0,1\}^3 = \{\underset{7}{000},\underset{8}{001},\underset{9}{010},\underset{10}{011},\underset{11}{100},\underset{12}{101},\underset{13}{110},\underset{14}{111}\}$

# How Many Python/Java programs?

- How do we represent Java/Python programs?
- How many things can we represent using that method?

# How many functions $\Sigma^* \to \Sigma^*$?

- Short answer: Too many!
  - Uncountable
  - $|\{f \mid f : \Sigma^* \to \Sigma^*\}| > |\mathbb{N}|$
- Conclusion: Some functions cannot be computed by any java/python program
- How to prove this?

# Uncountably many functions

- If we show a subset of $\{f \mid f: \Sigma^* \to \Sigma^*\}$ is uncountable, then $\{f \mid f: \Sigma^* \to \Sigma^*\}$ is uncountable too
- Consider just the "yes/no" functions (decision problems): $\{f \mid f: \{0,1\}^* \to \{0,1\}\}$
- The right-hand column is an infinite binary string that represents that function

| $b$ | $f(b)$ |
|---|---|
| "" | 1 |
| 0 | 0 |
| 1 | 0 |
| 00 | 1 |
| 01 | 1 |
| 10 | 1 |
| 11 | 1 |
| 000 | 0 |
| 001 | 0 |

# $|\{0,1\}^{\infty}| > |\mathbb{N}|$

- Idea:
  - show there is no way to "list" all finited binary strings
  - Any list of binary strings we could ever try will be missing elements of $\{0,1\}^{\infty}$

# $|\{0,1\}^\infty| > |\mathbb{N}|$

Attempt at mapping $\mathbb{N}$ to $\{0,1\}^\infty$

| | $b_0$ | $b_1$ | $b_2$ | $b_3$ | $b_4$ | $b_5$ | $b_6$ |
|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |
| 3 | 1 | 1 | 0 | 1 | 1 | 0 | 1 |
| 4 | 1 | 0 | 1 | 1 | 0 | 1 | 0 |
| 5 | 1 | 0 | 0 | 1 | 1 | 1 | 0 |
| 6 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| … | | | | | | | |
| | 0 | 1 | 0 | 0 | 1 | 0 | 0 |

A string that our attempt missed

Derive by selecting each $b_i$ as the opposite of the $b_i$ from row $i$

# $|\{0,1\}^\infty| > |\mathbb{N}|$ proof summary

- Assume towards reaching a contradiction that $\{0,1\}^\infty$ is countable
- This means we can find a bijection $f: \mathbb{N} \to \{0,1\}^\infty$
- Using $f$, we can find $s \in \{0,1\}^\infty$ which is not in the range of $f$:
  - let bit $i$ of $s$ be the opposite of bit $i$ of $f(i)$
  - This is missing from the range because it must be different from every output (at the position indexed by the input)

# Conclusion

- There are countably many strings
  - And therefore binary strings, programs, etc.
- We can’t write down (or compute) all things from an uncountable set
- There are uncountably many functions
- Some functions can’t be implemented

# Other countable/uncountable sets

- Countable sets:
  - Integers
  - Rational numbers
- Uncountable Sets:
  - Real numbers