

A decorative graphic on the left side of the slide, consisting of a network of white lines and small circles on a dark blue background, resembling a circuit board or a neural network.

DEFINING “COMPUTATION”

DISCRETE MATHEMATICS AND THEORY 2

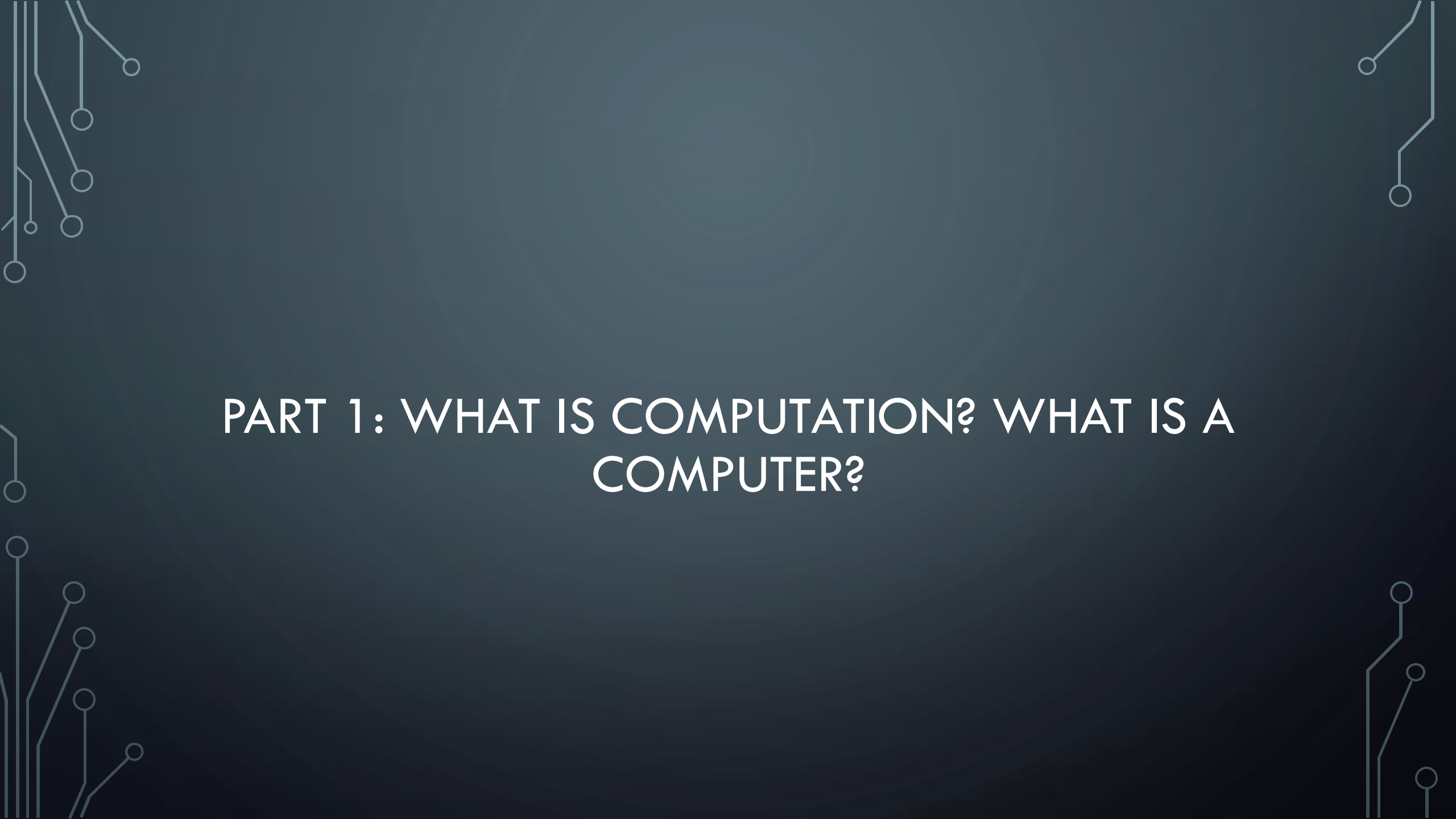
MARK FLORYAN

GOALS!

1. What exactly is “computation”. What exactly do we mean by “computation”? Let’s define these things clearly and explicitly.

2. What is the **birds-eye view** of theory of computation (for our course!). Are there different ways to “compute” that have strengths and weaknesses?

3. What “Math” do we need in order to start thinking about computation properly, and why?

The background is a dark blue gradient. In the corners, there are white line-art illustrations of circuit boards or neural networks, with lines connecting to small circles.

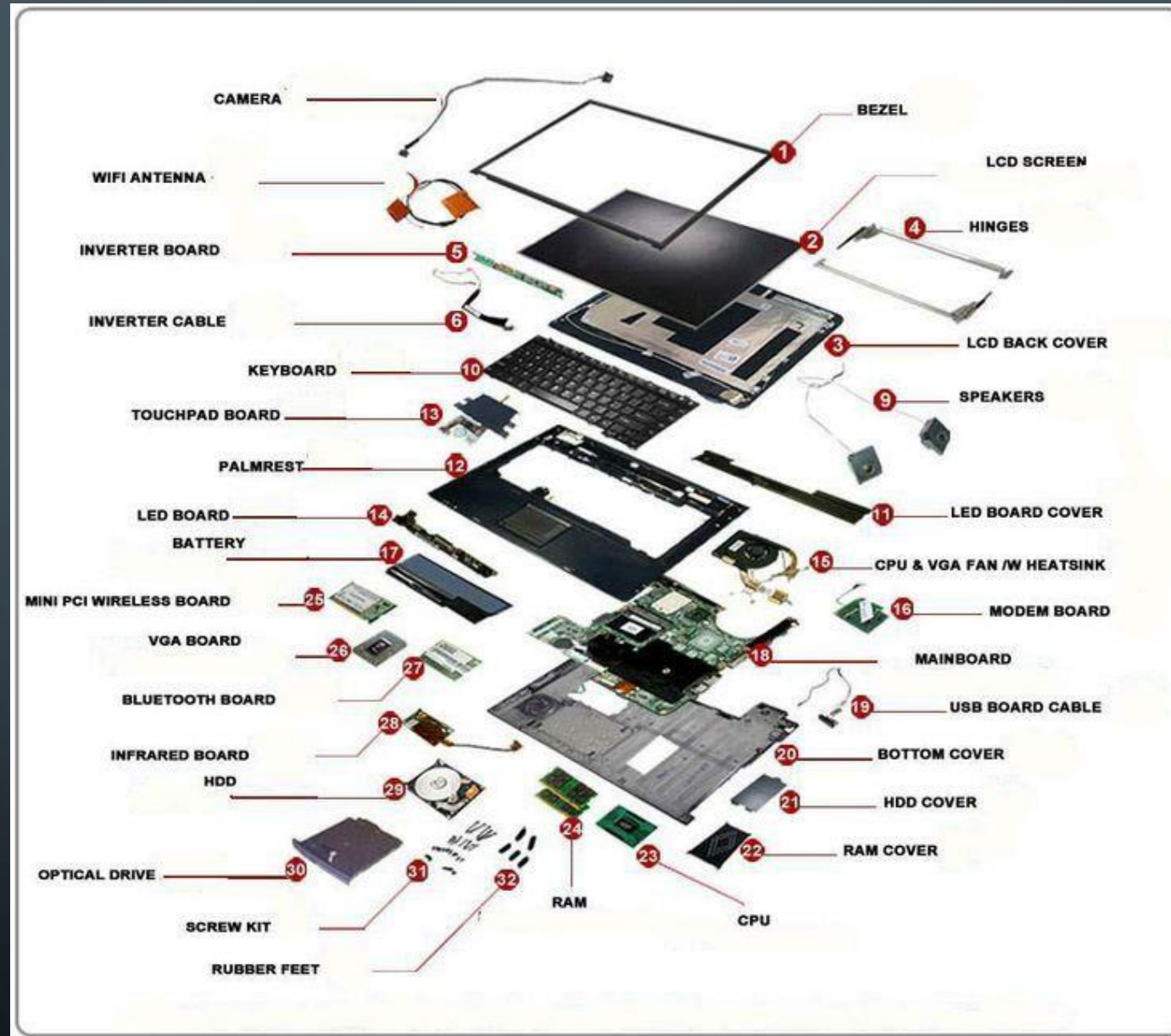
PART 1: WHAT IS COMPUTATION? WHAT IS A COMPUTER?

WHAT IS A COMPUTER?

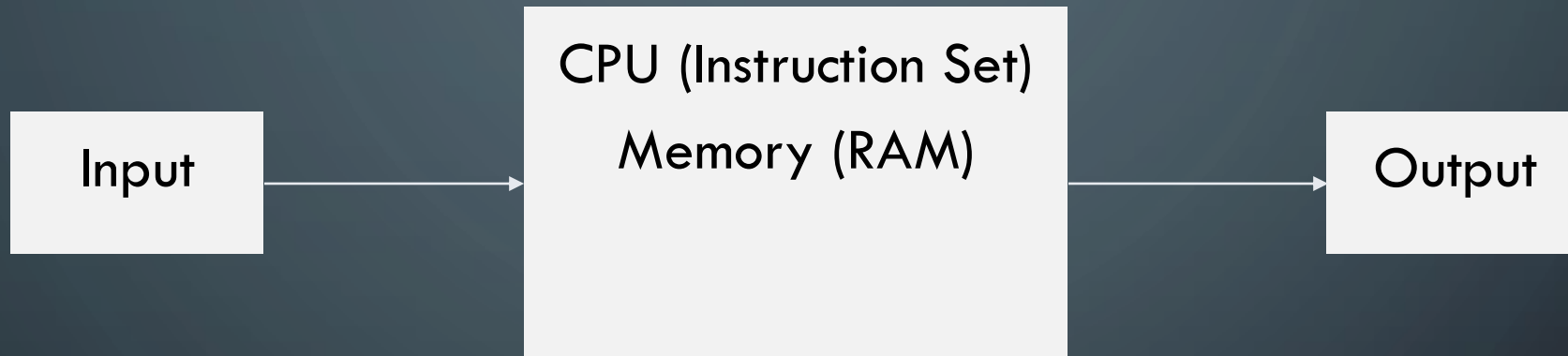
Class discussion:

What is a computer? What do you think?

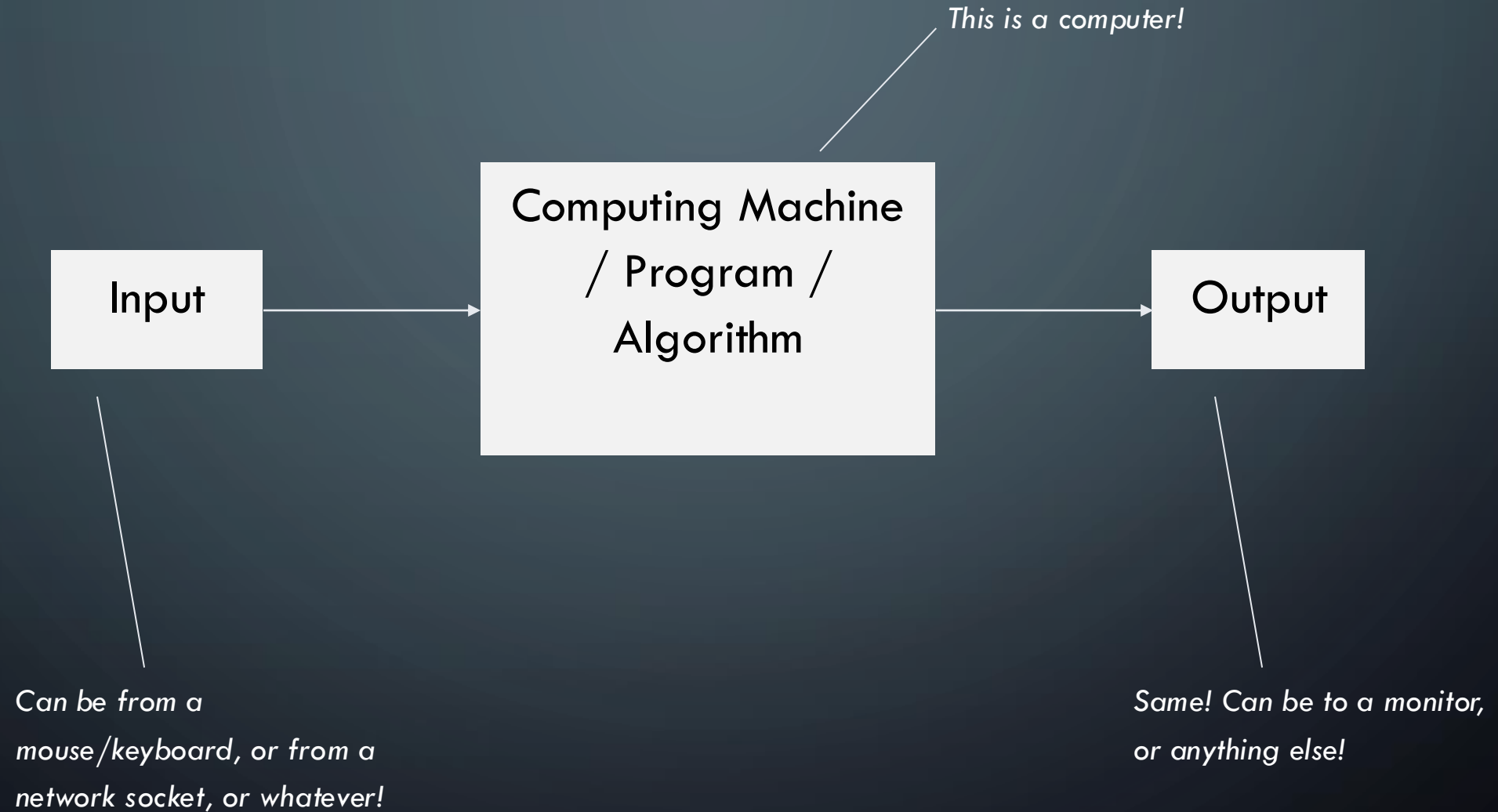
DISCUSSION: WHAT PARTS ARE NECESSARY AT MINIMUM?



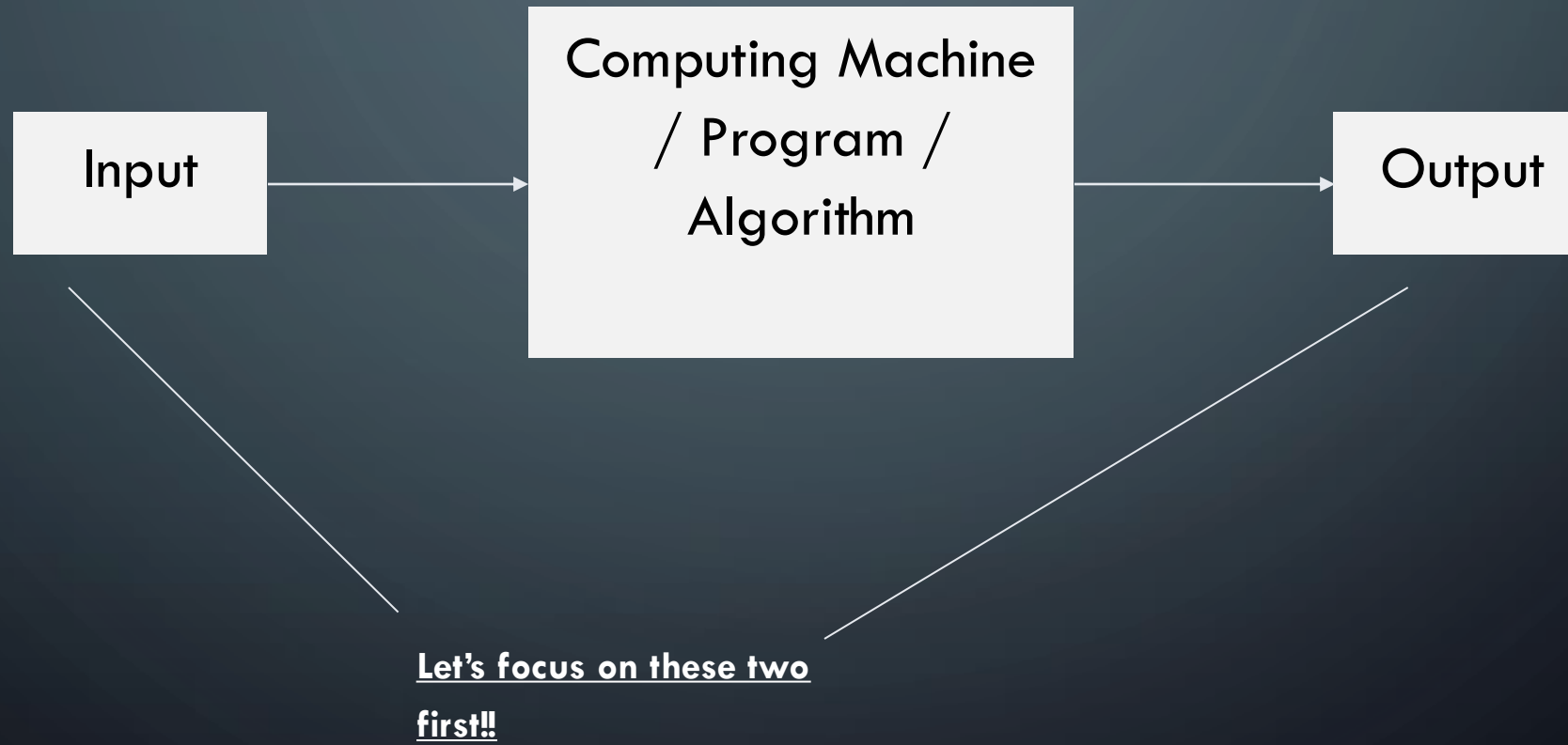
MY ANSWER



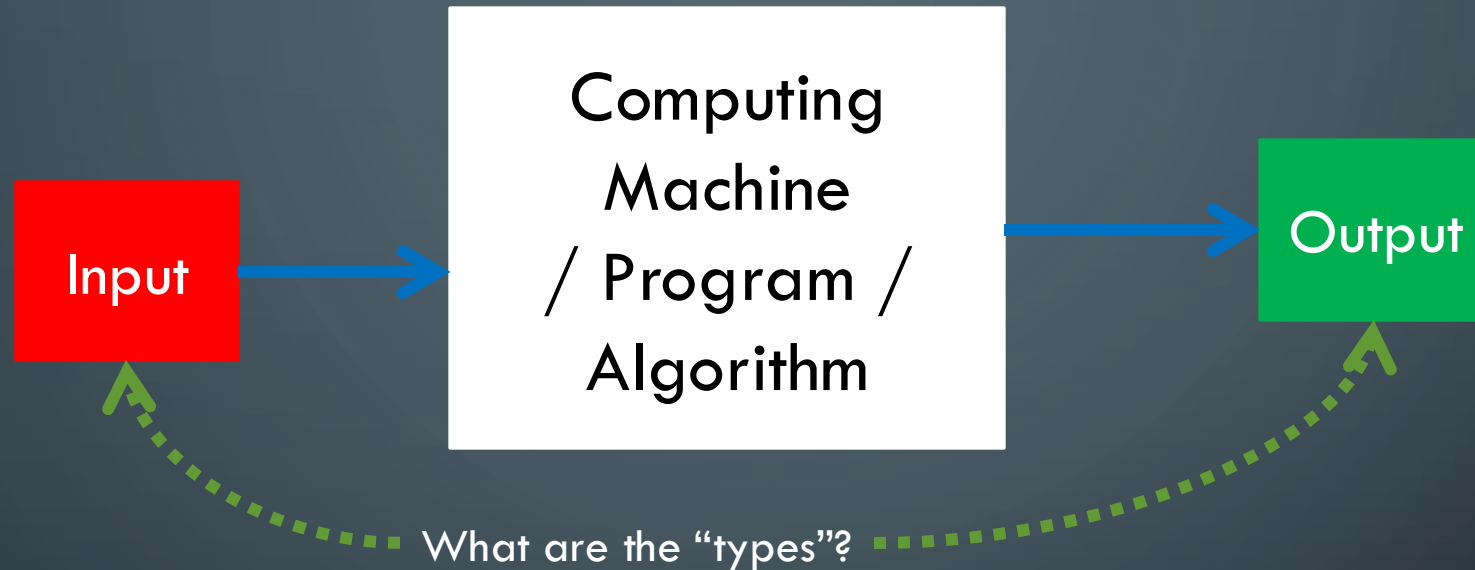
MY ANSWER



DEFINING INPUT AND OUTPUT



DEFINING INPUT AND OUTPUT

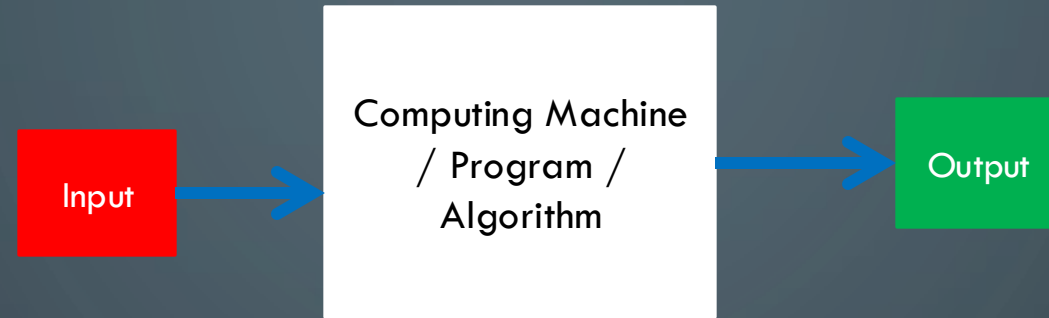


ON COMPUTABLE NUMBERS, WITH AN APPLICATION TO
THE ENTSCHEIDUNGSPROBLEM

By A. M. TURING.

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

DEFINING INPUT AND OUTPUT



- Input and output are strings
- Black box is an implementation
- What are we implementing?
 - Functions: Transformations from a set of strings to another set of strings. More on this soon!

DEFINING INPUT AND OUTPUT

An **Alphabet** is a finite set of characters

Notation:

Σ (\Sigma in LaTeX)

Examples:

$\Sigma = \{0,1\}$

$\Sigma = \{a, b, c, d, \dots, z\}$

$\Sigma = \{0,1,2, \dots, 9\}$

Strings are built up from an **Alphabet**

If:

$\Sigma = \{0,1, a, b, c\}$

Valid Strings:

00abc10

cccccc

a11

DEFINING INPUT AND OUTPUT

We can define structure of a string in various ways. Examples:

Σ^n

The set of all length n strings over alphabet Σ

$\{0,1\}^3$

The set of 3-bit strings.

$$\{0,1\}^3 = \{(x_0, x_1, x_2) : x_0, x_1, x_2 \in \{0,1\}\}$$

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

Pop Quiz: If $|\Sigma| = m$, then $|\Sigma^n| = ?$

DEFINING INPUT AND OUTPUT

The $*$ operator refers to a string of any length, including 0

Σ^*

The set of all strings over alphabet Σ of any length (including 0)

$\{0,1\}^*$

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

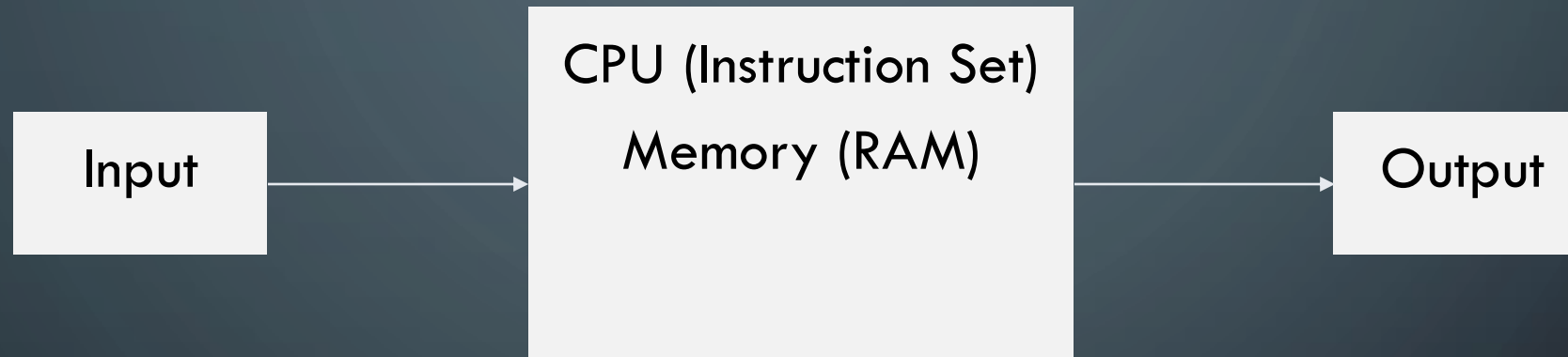
Note that:

- $\{0,1\}^3 = \{000, 001, 010, 011, \dots\}$
- *The set of three-bit strings*

Note that:

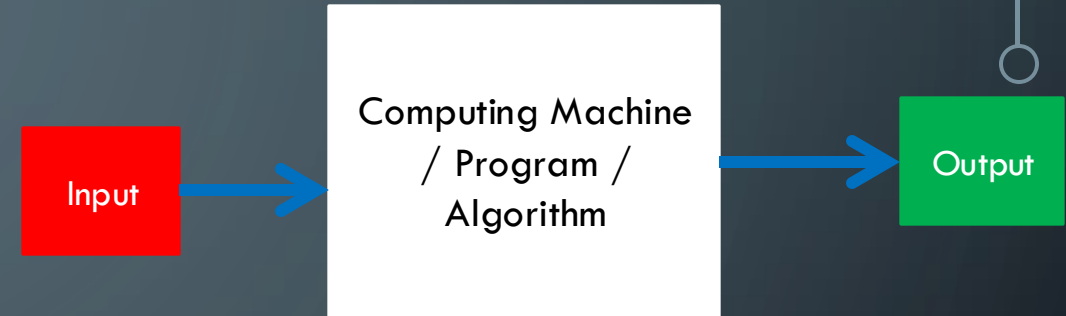
- $\{0,1\}^* = \{0,1\}^0 \cup \{0,1\}^1 \cup \{0,1\}^2 \cup \dots$
- $\{0,1\}^* = \bigcup_{n \in \mathbb{N}} \{0,1\}^n$

DEFINING COMPUTATION



Ok! What about this part?

COMPUTING A FUNCTION



- Definition of computation is based on functions
- 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

DEFINING COMPUTATION

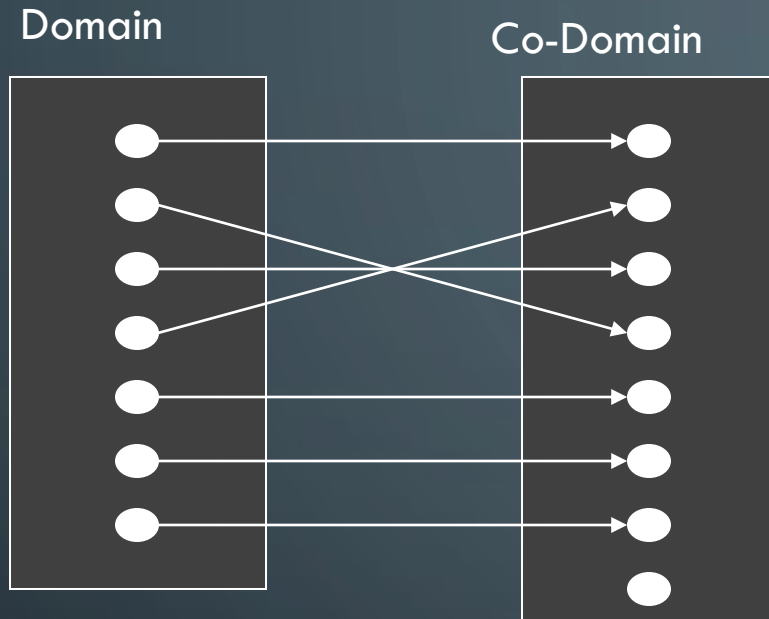
Function: a “mapping” from input to output

- $f: D \rightarrow C$
 - 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

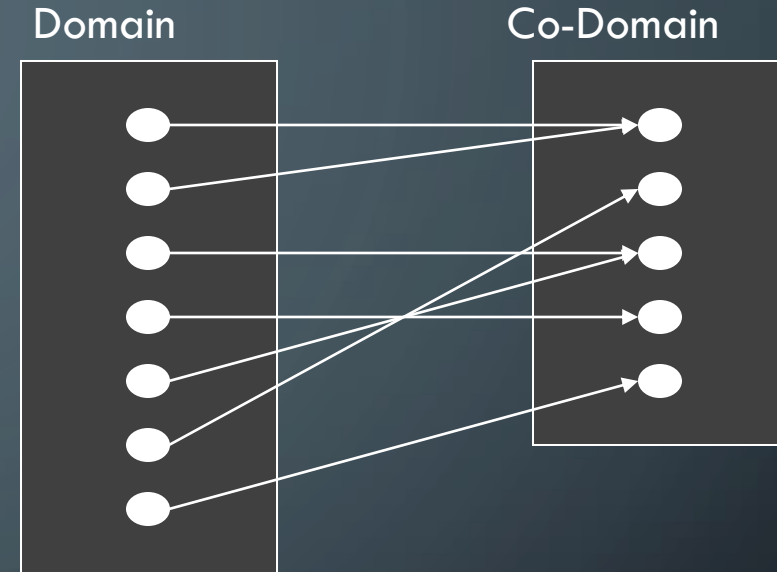
Finite function: a function with a finite domain

$f: D \rightarrow C$ is a finite function if D is finite. Otherwise it's an infinite function

INJECTIVE FUNCTIONS



INJECTIVE FUNCTION



NON-INJECTIVE FUNCTION

One-to-one (injective)

$$x \neq y \Rightarrow f(x) \neq f(y)$$

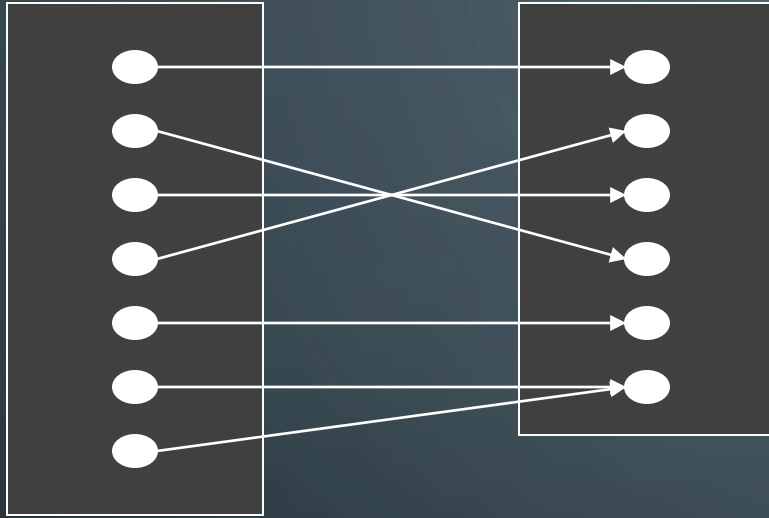
Different inputs yield different outputs

No two inputs share an output

ONTO, SURJECTIVE FUNCTIONS

Domain

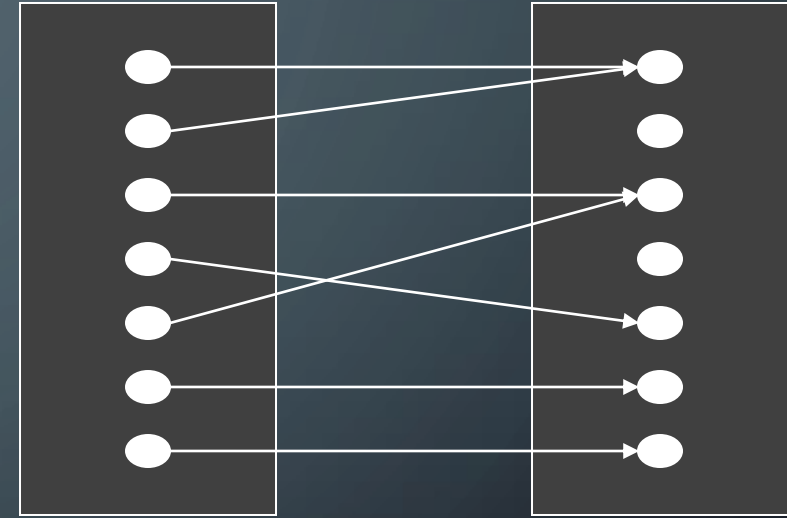
Co-Domain



SURJECTIVE FUNCTION

Domain

Co-Domain



NON-SURJECTIVE FUNCTION

Everything in Co-Domain “receives” something

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

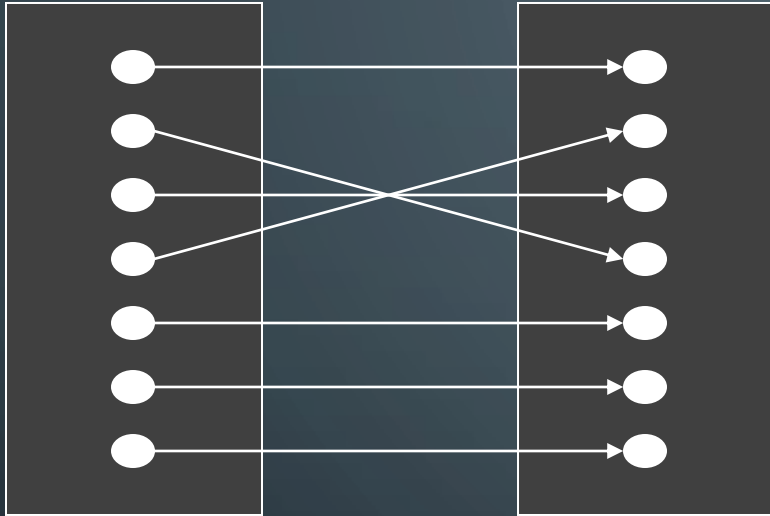
PROPERTIES OF FUNCTIONS

- 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 Correspondence (bijective)
 - Both one-to-one and surjective
 - Everything in C is mapped to by a unique element in D
 - All elements from domain and co-domain are perfectly “partnered”

BIJECTIVE FUNCTIONS

Domain

Co-Domain



BIJECTIVE FUNCTION

Because Onto:

Everything in Co-Domain “receives” something

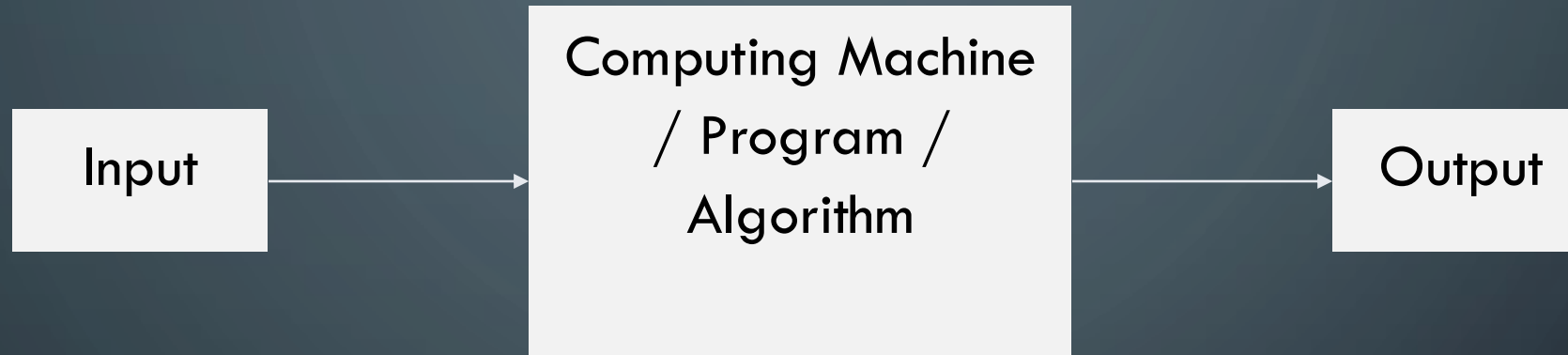
Because 1-1:

Nothing in Co-Domain “receives” two things

Conclusion:

Things in the Domain exactly “partner” with things in Co-Domain

OVERVIEW OF COMPUTATION!



Input / Output are Strings

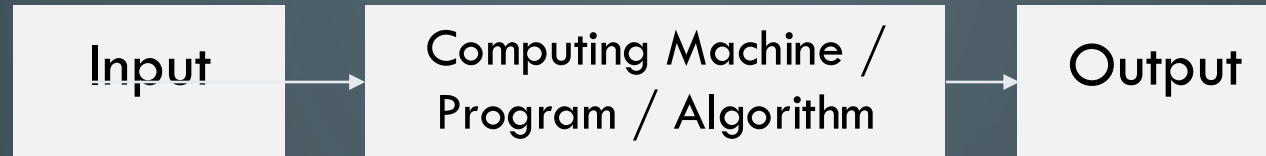
Computing Machine is something that implements a set of functions that we care about.

The background is a dark blue gradient. In the corners, there are white, stylized circuit-like lines with small circles at the ends, resembling a network or data flow diagram. These lines are more prominent in the top-left and bottom-left corners, and less so in the top-right and bottom-right corners.

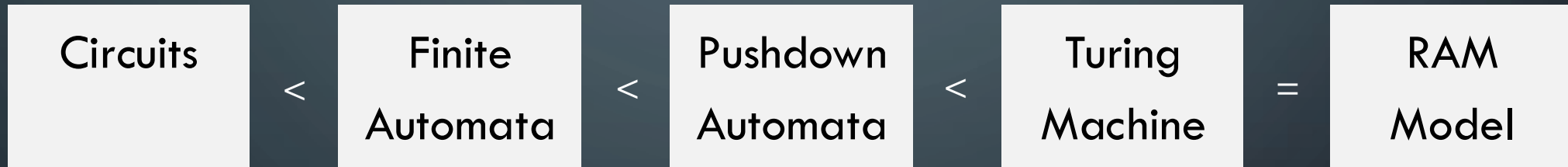
PART 2: BIRD'S EYE VIEW OF THEORY OF COMPUTATION?

OVERVIEW OF THEORY OF COMPUTATION

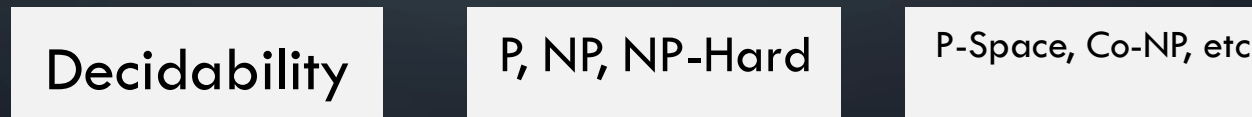
Defining Computation



Computational Models



Computational Complexity



The background is a dark blue gradient with a large, faint, light blue circle in the center. In the four corners, there are white line art elements resembling circuit traces or neural network connections, with small circles at the end of the lines.

PART 3: BACKGROUND MATH

OBJECTS THAT WE'LL NEED

- Sets
- Functions (already started looking at this)
- Strings (already looked at this briefly)

REVIEW: SETS

SETS

- Notation: $\{1,2,4\}$
- Order does not matter
 - $\{1,2,4\} = \{2,1,4\}$
- No Duplicates
 - $\{2,1,1\}$
- Equal when they have the same members

SEQUENCES/TUPLES

- Notation: $(1,2,4)$
- Order matters
 - $(1,2,4) \neq (2,1,4)$
- Duplicates allowed
 - $(2,1,1) \neq (2,1)$
- Equal when they have the same elements in the same order

REVIEW: SET OPERATORS

- $=$

- \in

- \subseteq

- \supseteq

- \subset

- \supset

- \cup

- \cap

- \setminus or $-$

- \times

- $\mathcal{P}(S)$

REVIEW: SET OPERATORS

- $=$ Set equality
- \in Set membership
- \subseteq Subset
- \supseteq Superset
- \subset Proper Subset
- \supset Proper Superset

- \cup Set Union
- \cap Set Intersection
- \setminus or $-$ Set Difference
- \times Cross Product
- $\mathcal{P}(S)$ Power Set

IMPORTANT SETS

\emptyset

\mathbb{N}

\mathbb{Z}

\mathbb{Q}

$\{0,1\}$

IMPORTANT SETS

\emptyset

Null Set; Set containing nothing

\mathbb{N}

Natural Numbers = $\{0, 1, 2, 3, \dots\}$

\mathbb{Z}

Integers = $\{\dots, -3, -2, -1, 0, 1, 2, 3, \dots\}$

\mathbb{Q}

Rational Numbers = $\left\{\frac{x}{y} \mid x, y \in \mathbb{Z}\right\}$

$\{0,1\}$

Set containing 0 and 1 (common alphabet for binary #s)

SET CARDINALITY

- The number of distinct members of a set.
 - How many things I could put on the left hand side of \in to make the statement true?

Pop Quiz:

- $|\{1,2,3\}| =$
- $|\{1\}| =$
- $|\emptyset| =$
- $|\{\emptyset\}| =$
- $\left|\left\{1,2,\{1,2\},\{1,2,\{1,2\}\}\right\}\right| =$
- $|\mathbb{N}| =$

SET BUILDER NOTATION

Version 1:

- $\{x \in S \mid P(x)\}$
- “The set of all members of S that make P true”
- $\{n \in \mathbb{N} \mid n^2 \leq 16\}$

Version 2:

- $\{f(x) \mid P(x)\}$
- “The set of all results of f when applied to members of our universe that make P true”
- $\{c^2 \mid c \in \mathbb{Z} \wedge c \leq 4\}$

COMPARING CARDINALITIES WITH FUNCTIONS

- 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 $[k] = \{n : n \in \mathbb{N} \wedge n < k\}$
- 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:

COMING UP!

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

REVIEW OF THIS DECK!

- A computer is any process that can take strings as input and produce strings as output.
- Computation of that machine / computer is a function (it maps inputs to outputs)
- Moving forward:
 - More math background
 - Different types of computers and the pros / cons of each