Intro to Coding with Python– How Computers Work

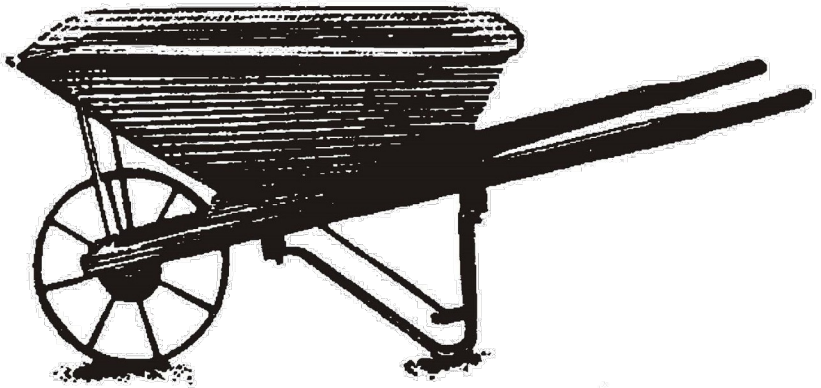
Dr. Ab Mosca (they/them)

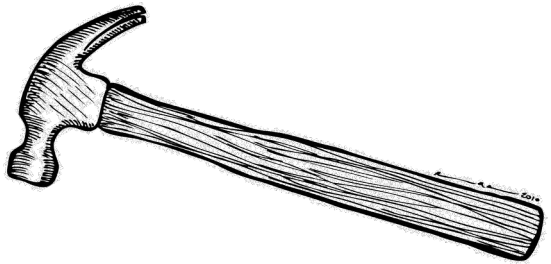
Slides based off slides courtesy of Jordan Crouser (https://jcrouser.github.io/)

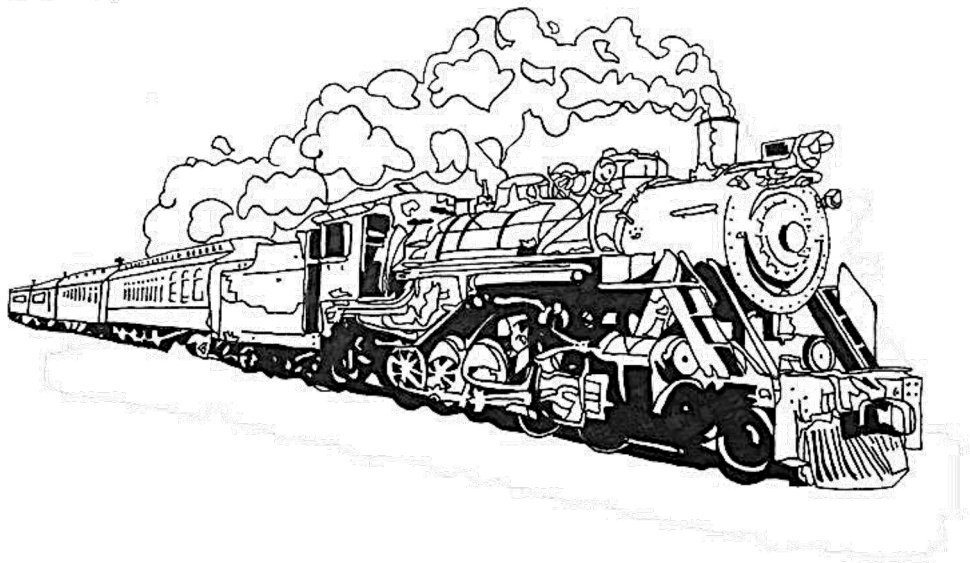
Plan for Today

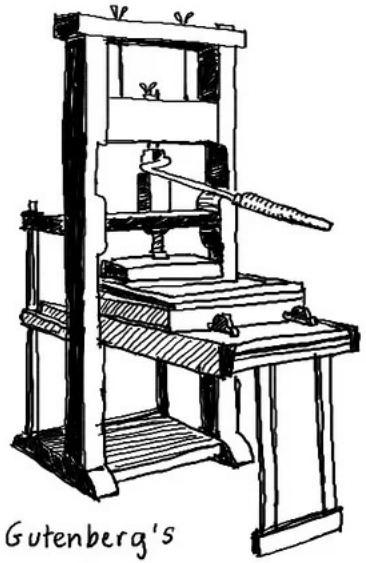
- 4 key components to computers
- Computer *hardware* crash course
- Boolean logic

A little history…





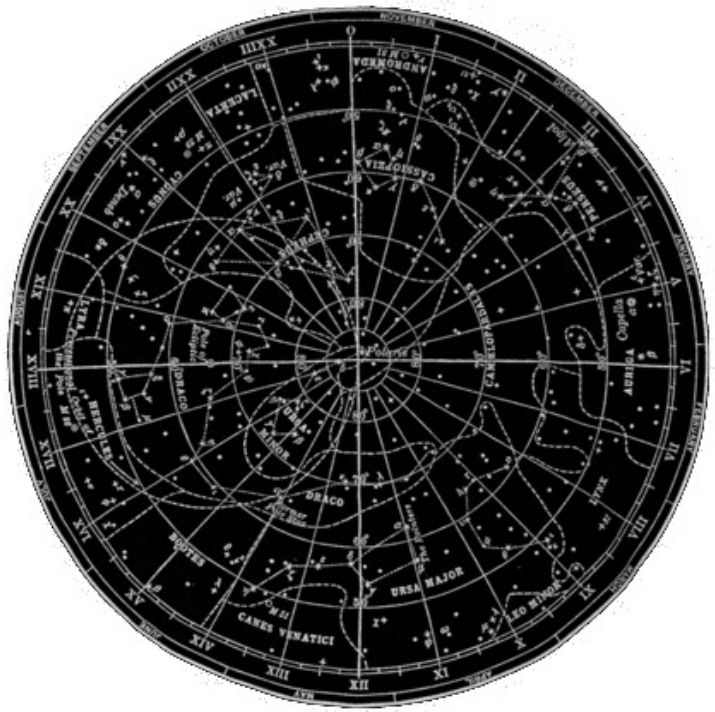




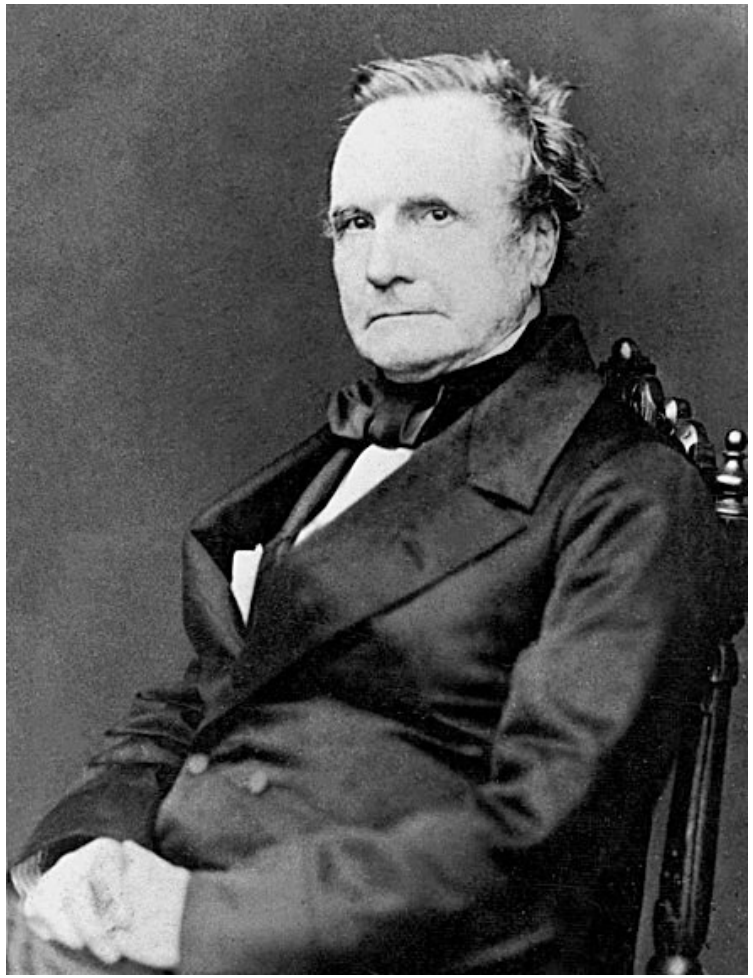
This section based on examples from "What Makes a Computer, a Computer?" by May-Li Khoe and Nat Brown

A little history…

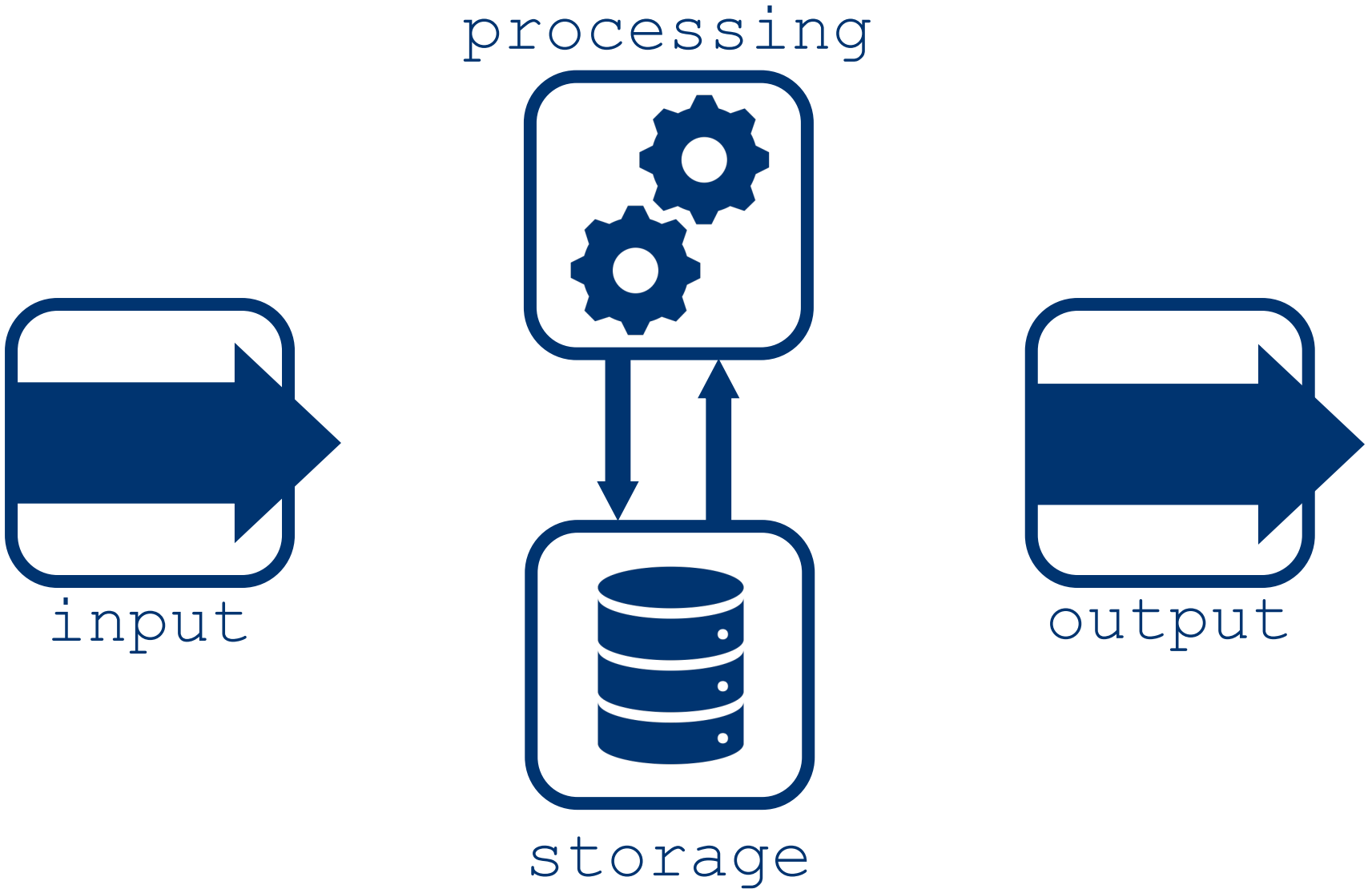
$$(x+3)^2 = 4$$



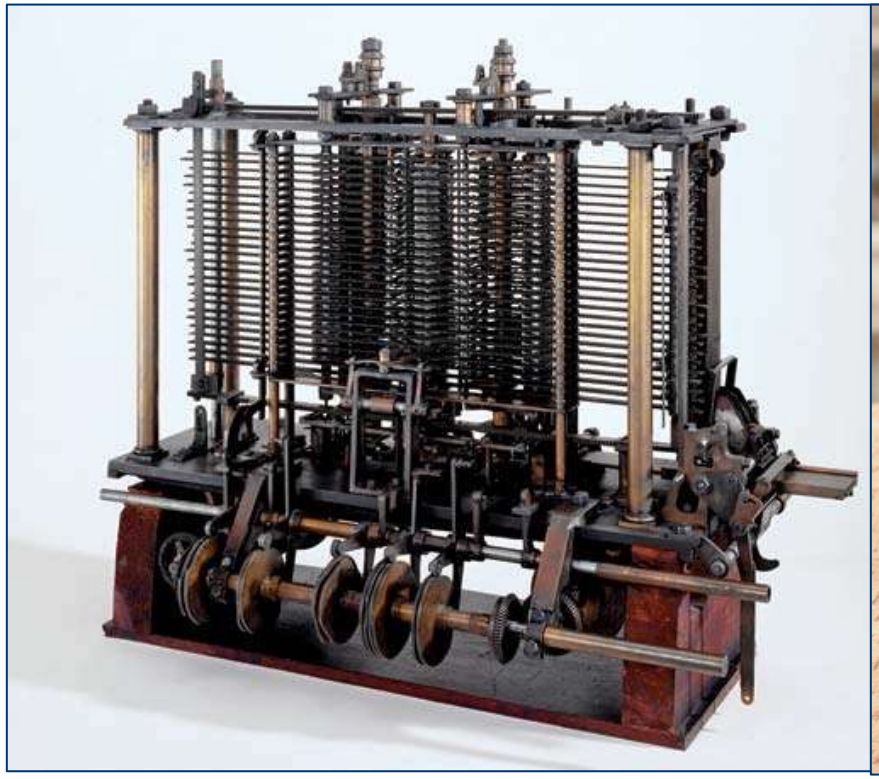
A little history…



4 basic tasks

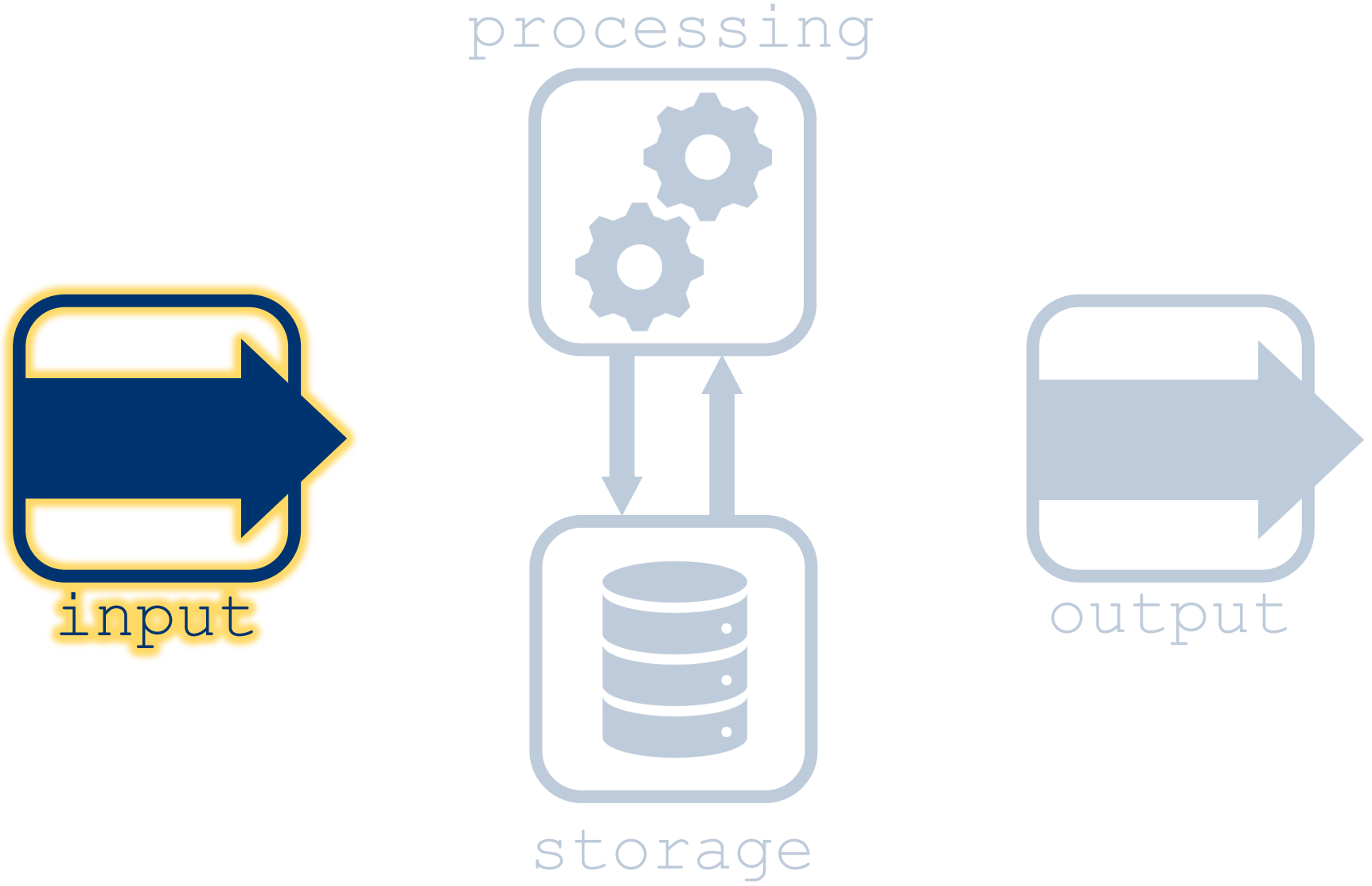


Then and now



Early 19th century

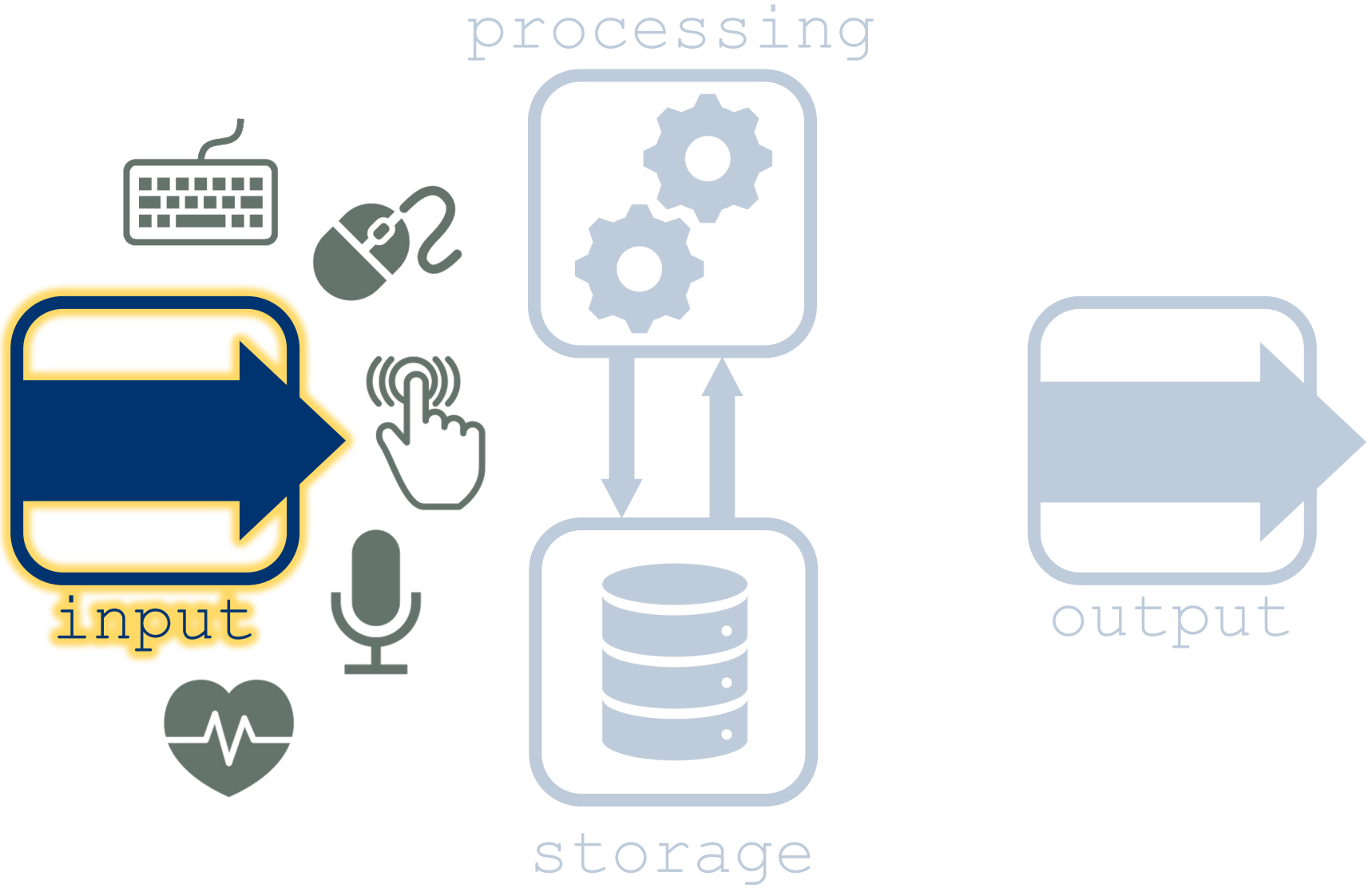
Input



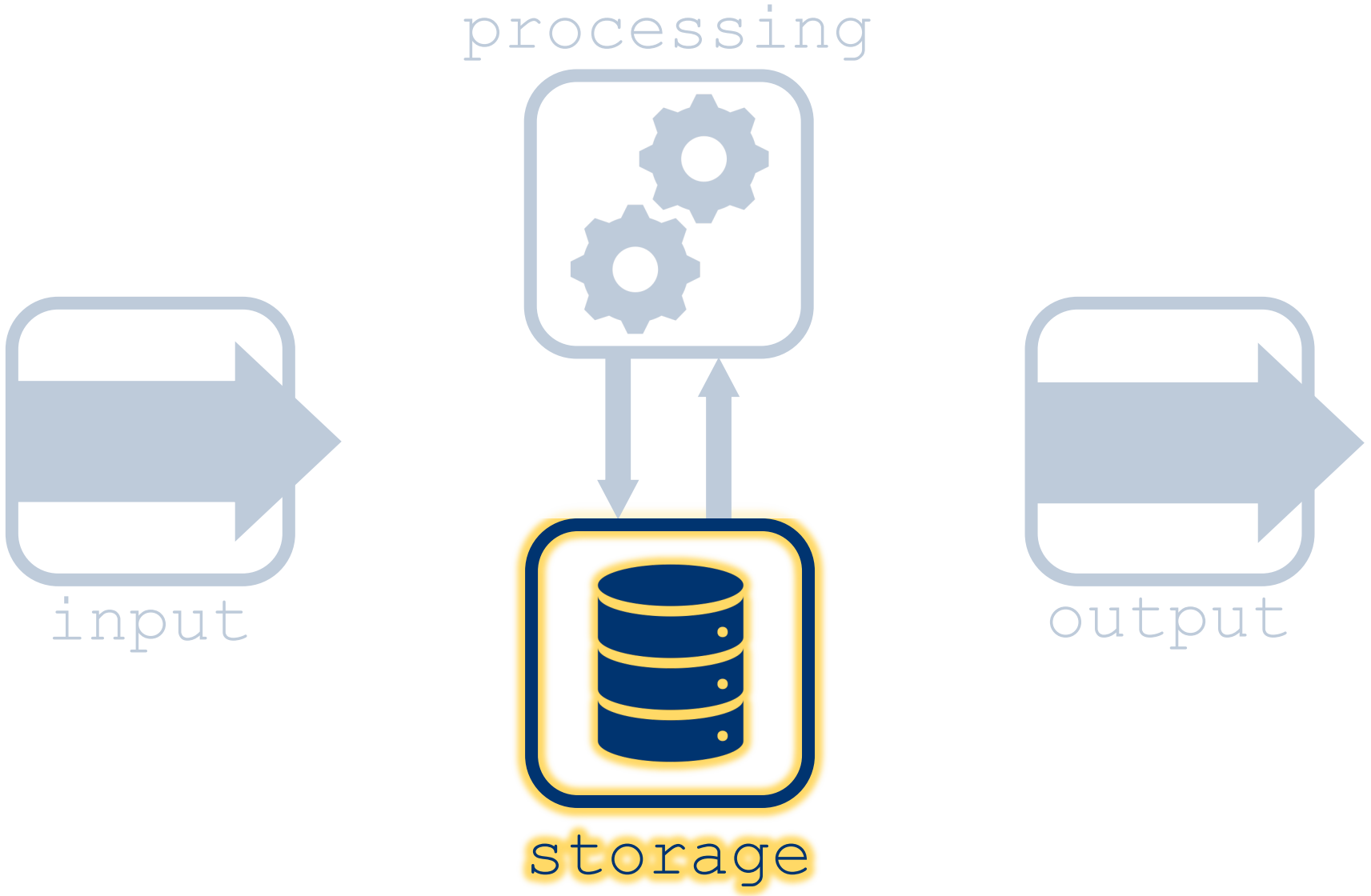
Input

What are some ways that you get information **into** a computer?

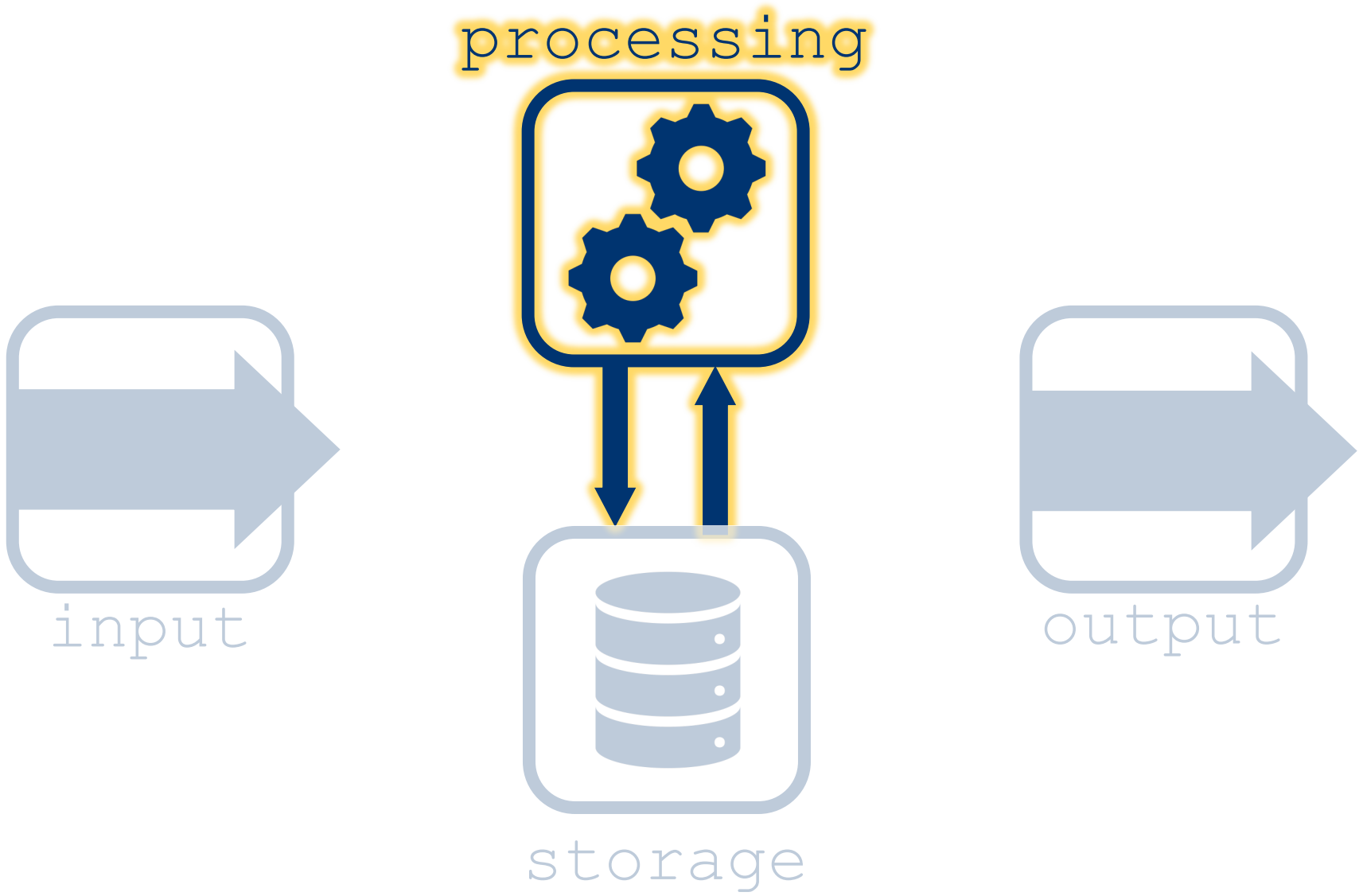
Input



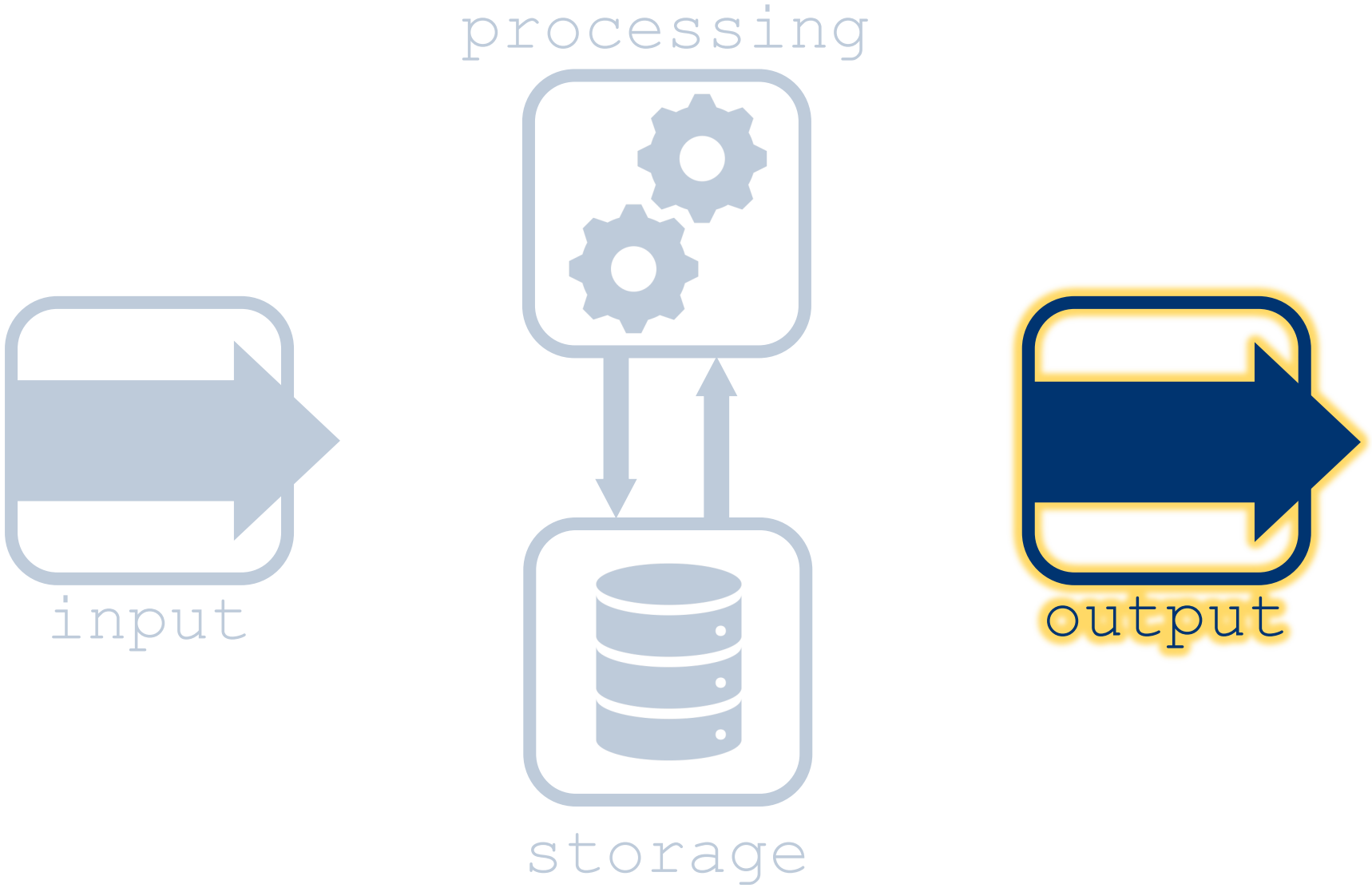
Storage



Processing



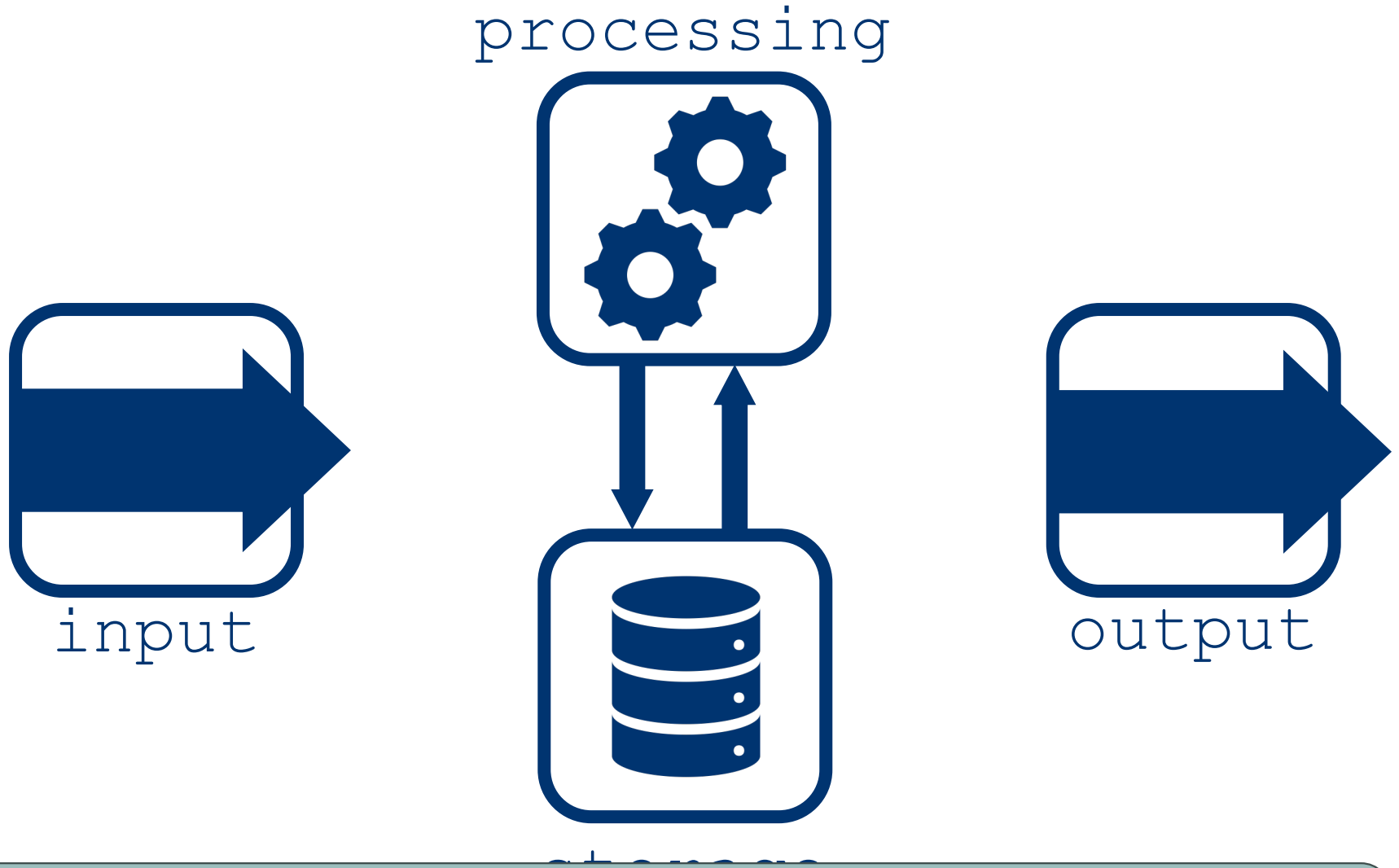
Output



Output

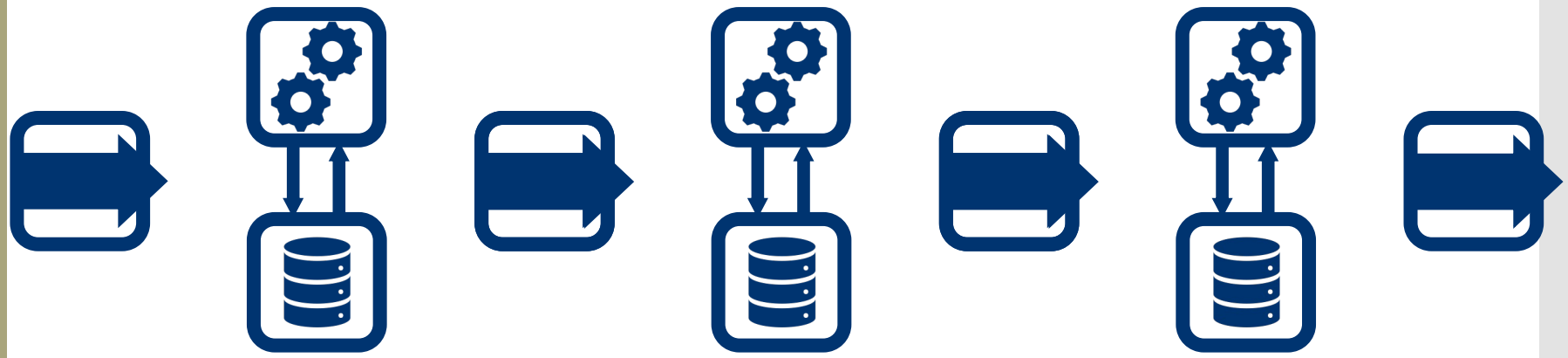
What are some ways that you get information **out** of a computer?

4 basic tasks



Think of an example of a computer you interact with daily. What are the input and output of your example?

Networks



Looking Under the Hood

Hard Disk Drive (old)

Hard Disk Drive (old)

Solid State Drive

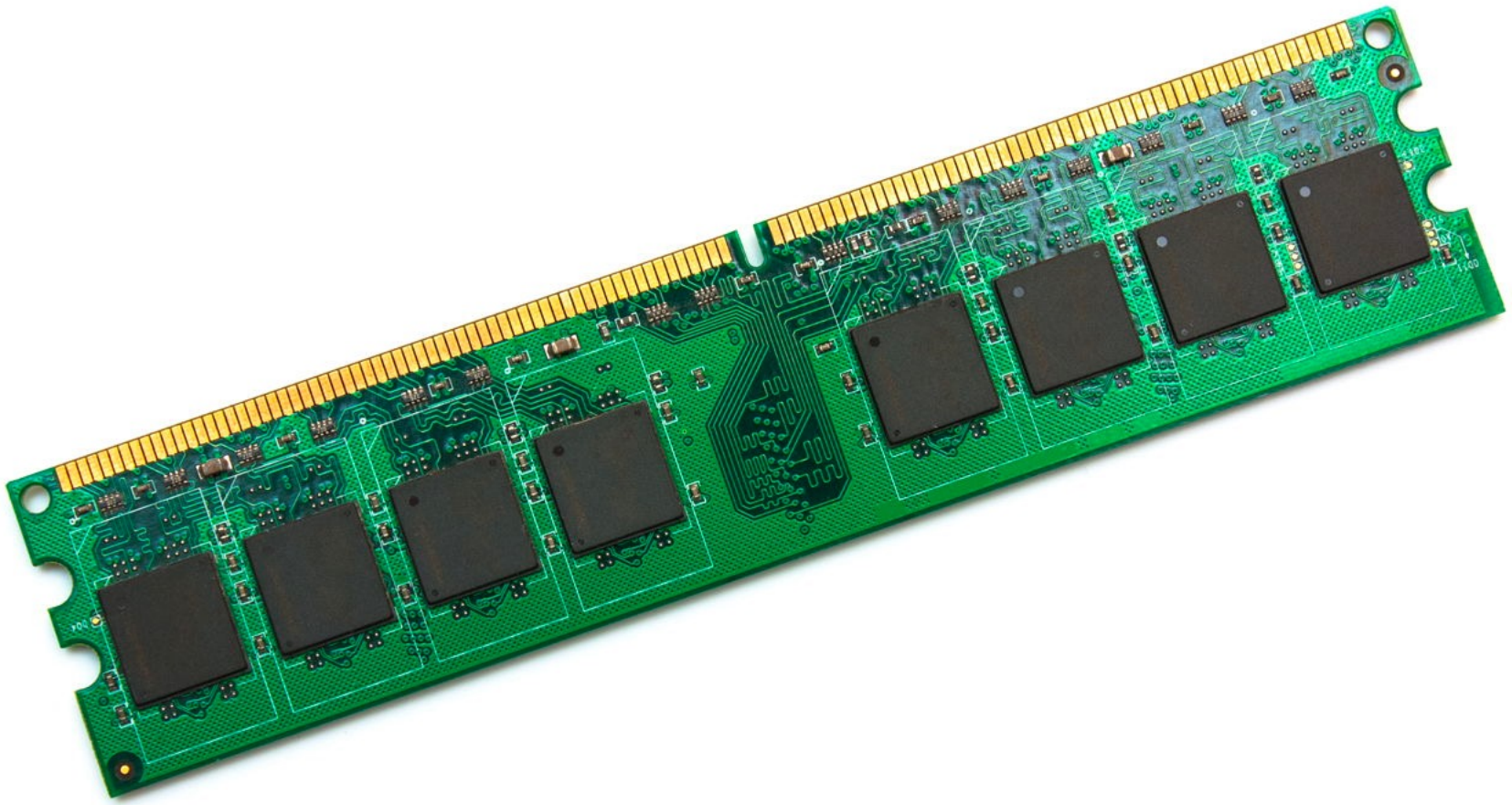
https://www.next7it.com/insights/standard-computer-components/

CPU (Central Processing Unit)

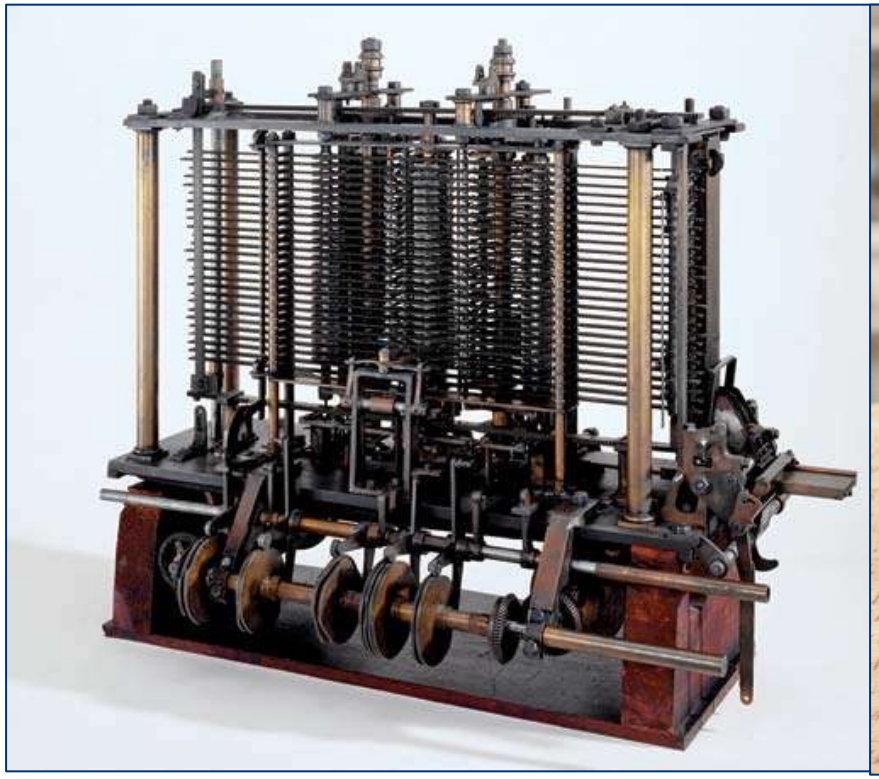
CPU (Central Processing Unit)

RAM (Random Access Memory)

RAM (Random Access Memory)



Then and now



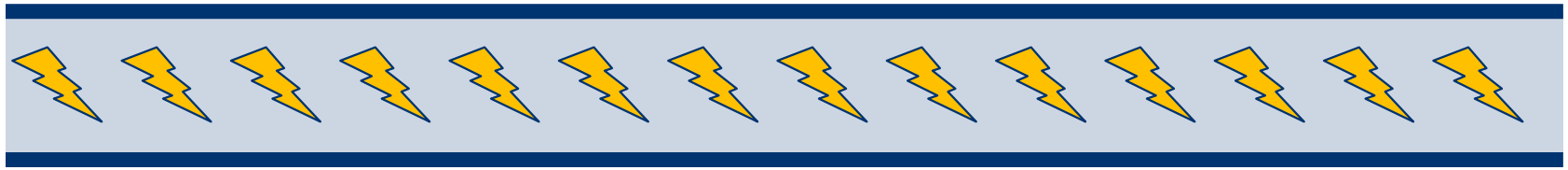
Early 19th century

Discussion

How is information represented using **electricity**?

One wire: a "bit"

"off"

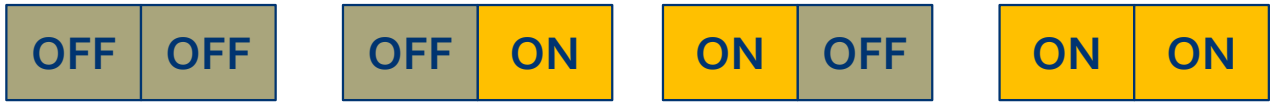


"on"

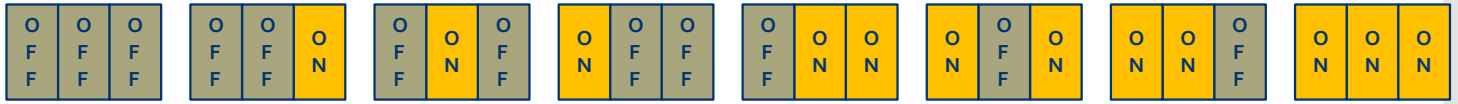
Multiple bits

- 1 bit:

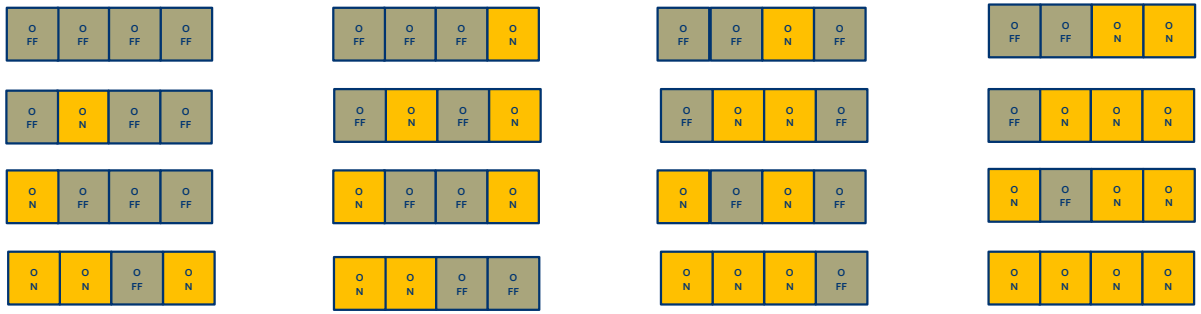
- 2 bits:



- 3 bits:

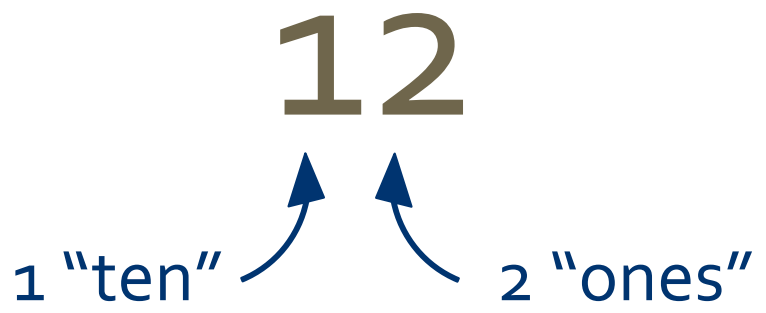


- 4 bits:



Using bits to represent numbers

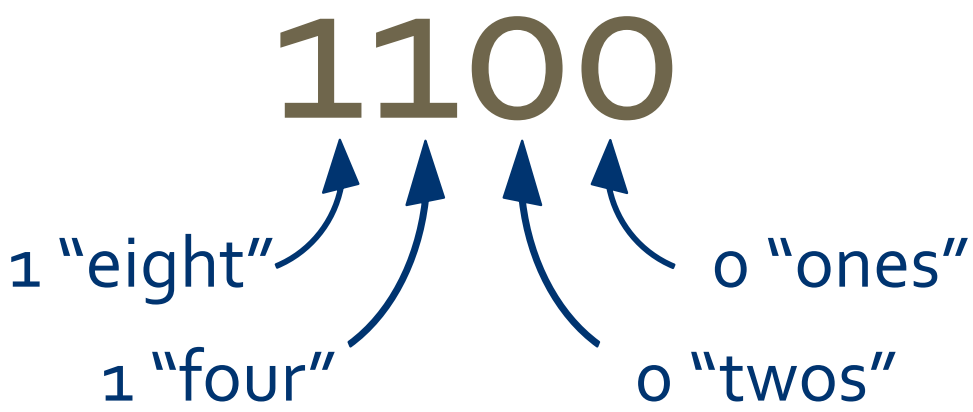
- In **base-10**, each place represents a power of 10, and each digit can take on a value from 0 to 9:



$$(1 * 10^1) + (2 * 10^0) = 12$$

Using bits to represent numbers

- In **base-2 ("binary")**, each place represents a power of 2, and each digit can take on a value of either 0 or 1:



$$(1 * 2^3) + (1 * 2^2) + (0 * 2^1) + (0 * 2^0) = 12$$

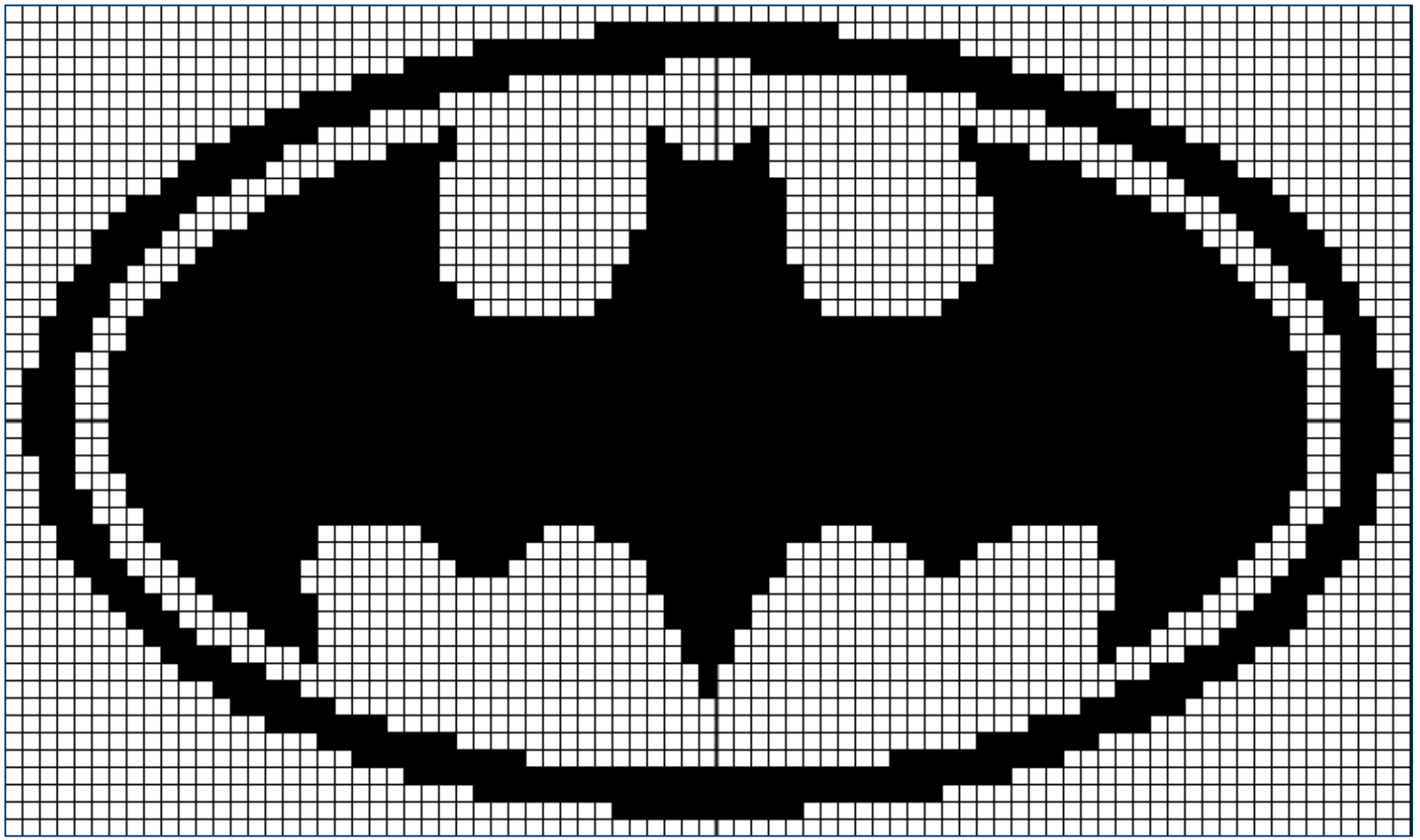
How much can we represent?

- With 8 bits*, we can represent the numbers 0 to 255
  - *8 bits is called a "byte"
- With 32 bits, we can represent numbers **> 4 billion**
- With 266 bits, we can represent **more unique numbers** than there are believed to be **atoms in the universe**

How much can we represent?

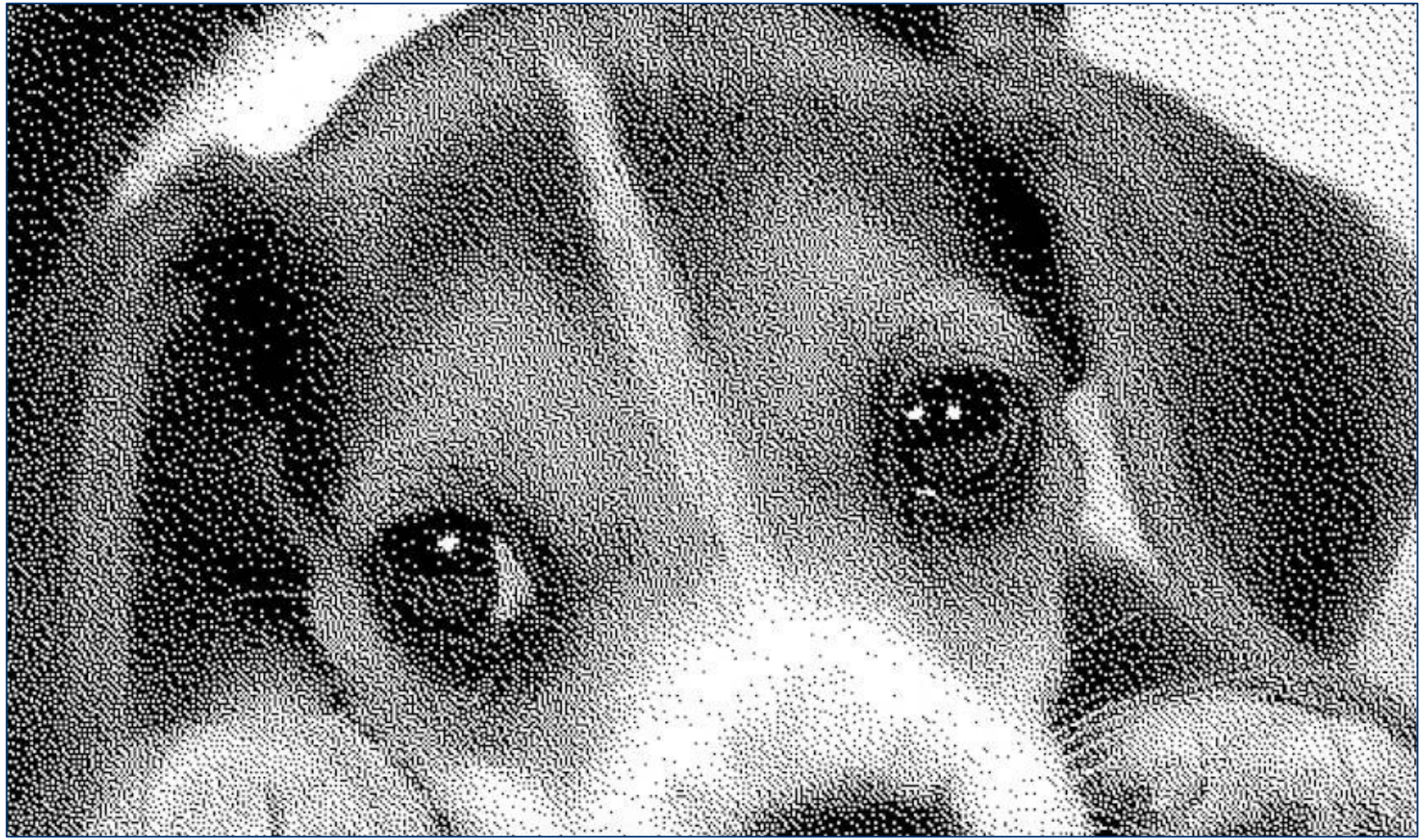
Besides numbers, what other things could we represent with binary?

Binary images?



[0,1]

Greyscale images?



[0:255]

What about color?

([0:255], [0:255], [0:255])

The tradeoff

- Smartphone camera (8 megapixels): 3296 x 2472 pixels
  - each requires 4 bytes to represent RGB + opacity
  - 32,590,848 bits ≈ 4MB (1MB = 1024 bytes, 1 byte = 8 bits)
- HD video: 1920 x 1080 pixels, 30fps
  - 5 minutes of video = 300 seconds = 9000 frames
  - 2,073,600 bits per frame ≈ 2.33GB (1GB = 1024MB)

The good news

- “High level” programming languages like Python mean we don’t have to write in “low level” binary
- Instead, we write statements like:

```
print(“hello”)
```