# What is a bit?

## CS143: lecture 13

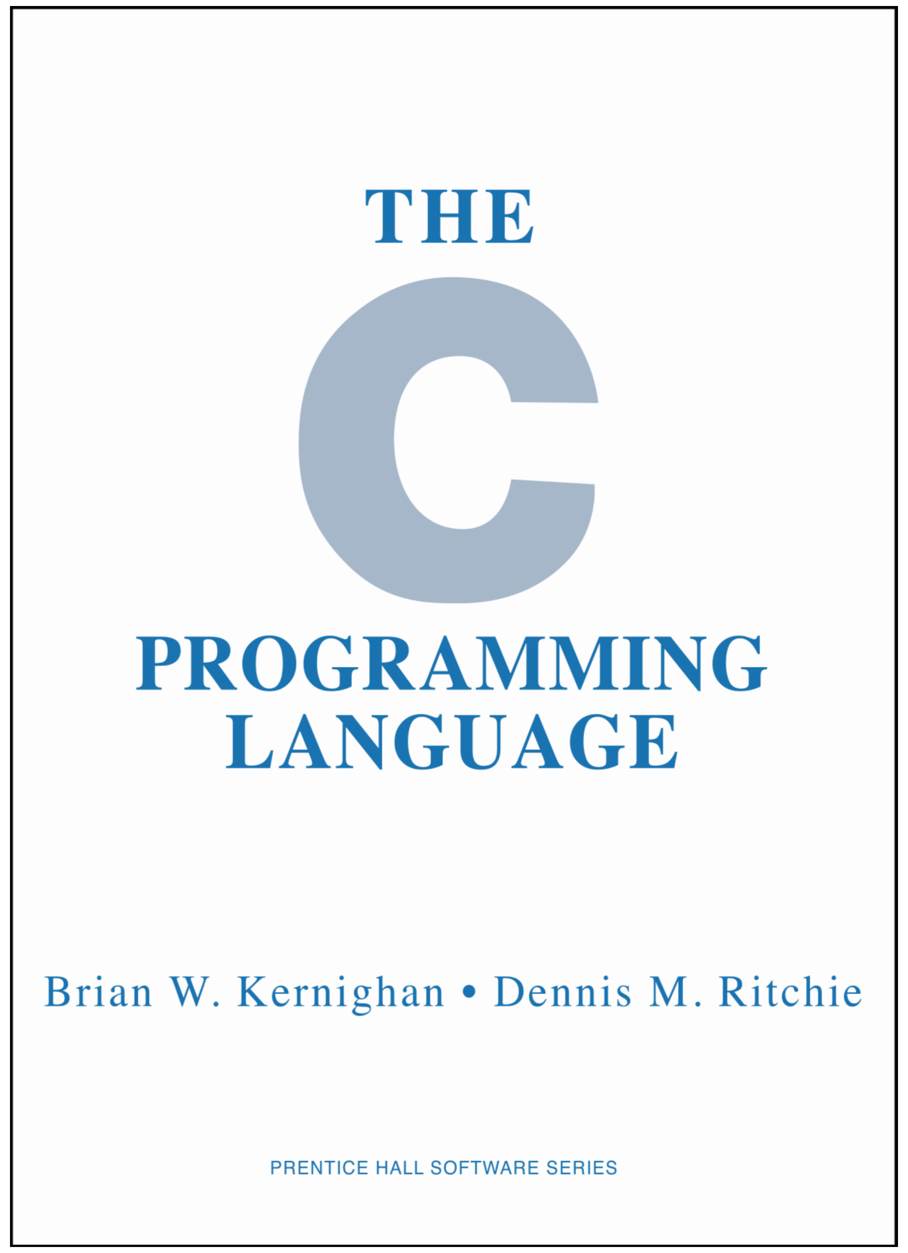
Byron Zhong, July 17

# Information

## Which one contains more information?

The Joseph Regenstein Library

This book:



THE

C

PROGRAMMING LANGUAGE

Brian W. Kernighan • Dennis M. Ritchie

PRENTICE HALL SOFTWARE SERIES

# Information

## How about...

Eckhart Library

The John Crerar Library

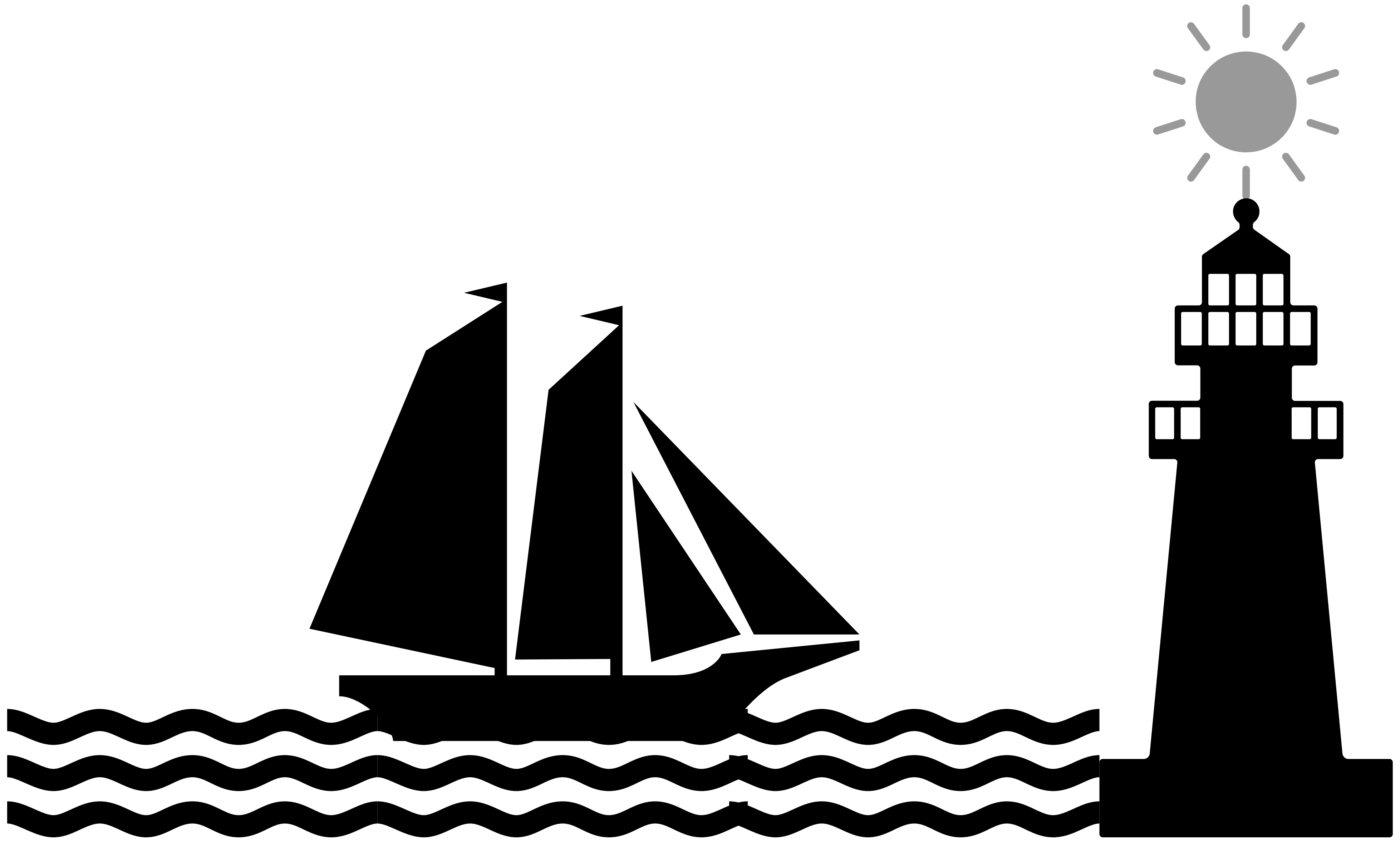
# Information

- We can weigh objects
- We can determine the volume of solid objects or liquids
- We can measure the height of walls in this room

- Can we measure information?
- Can we distinguish more information from less?
- What is the unit of information?

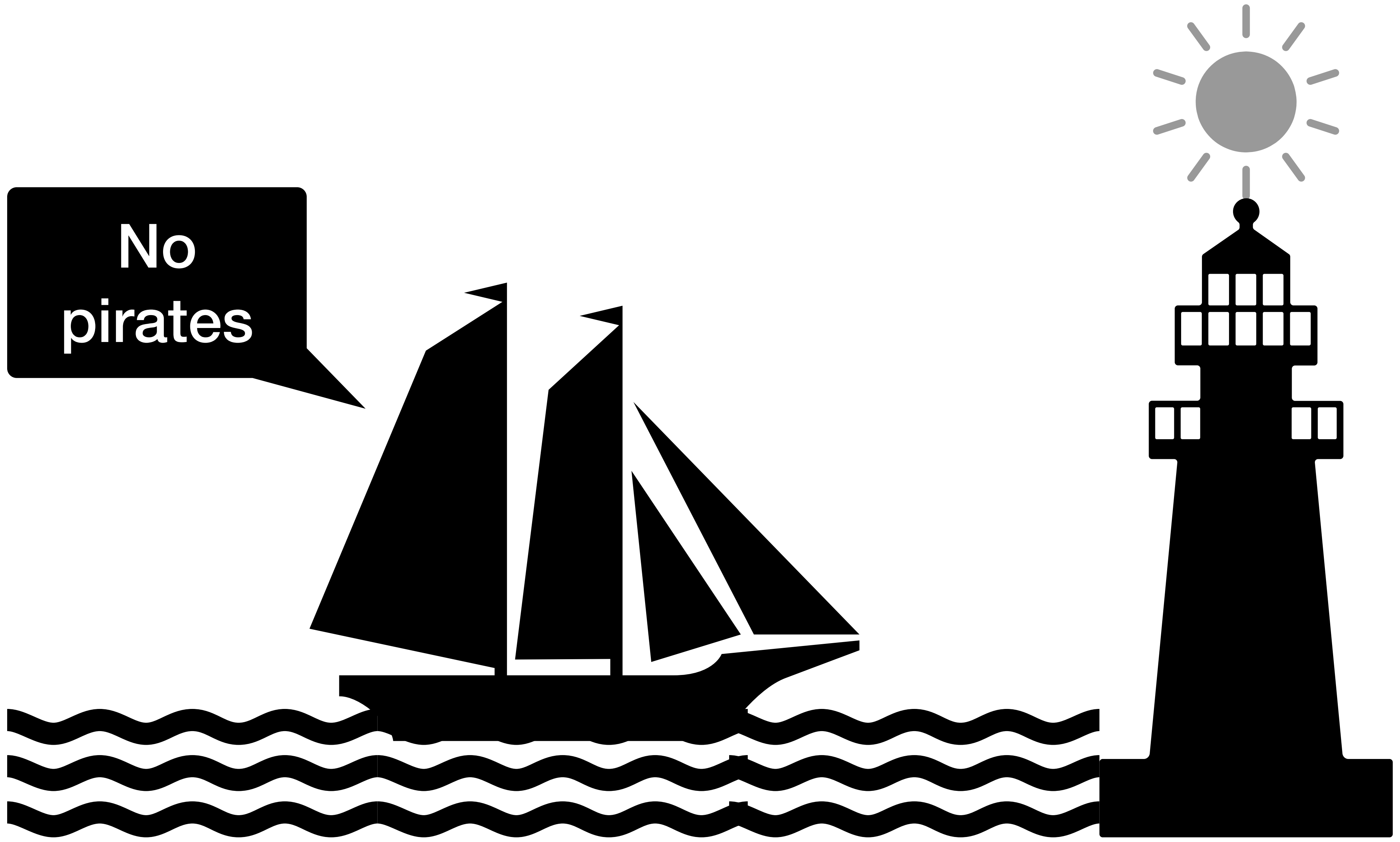
# Information

- Example: a lighthouse operator and the captain wanted to communicate if there are pirates ahead. What is the shortest message the lighthouse can send?



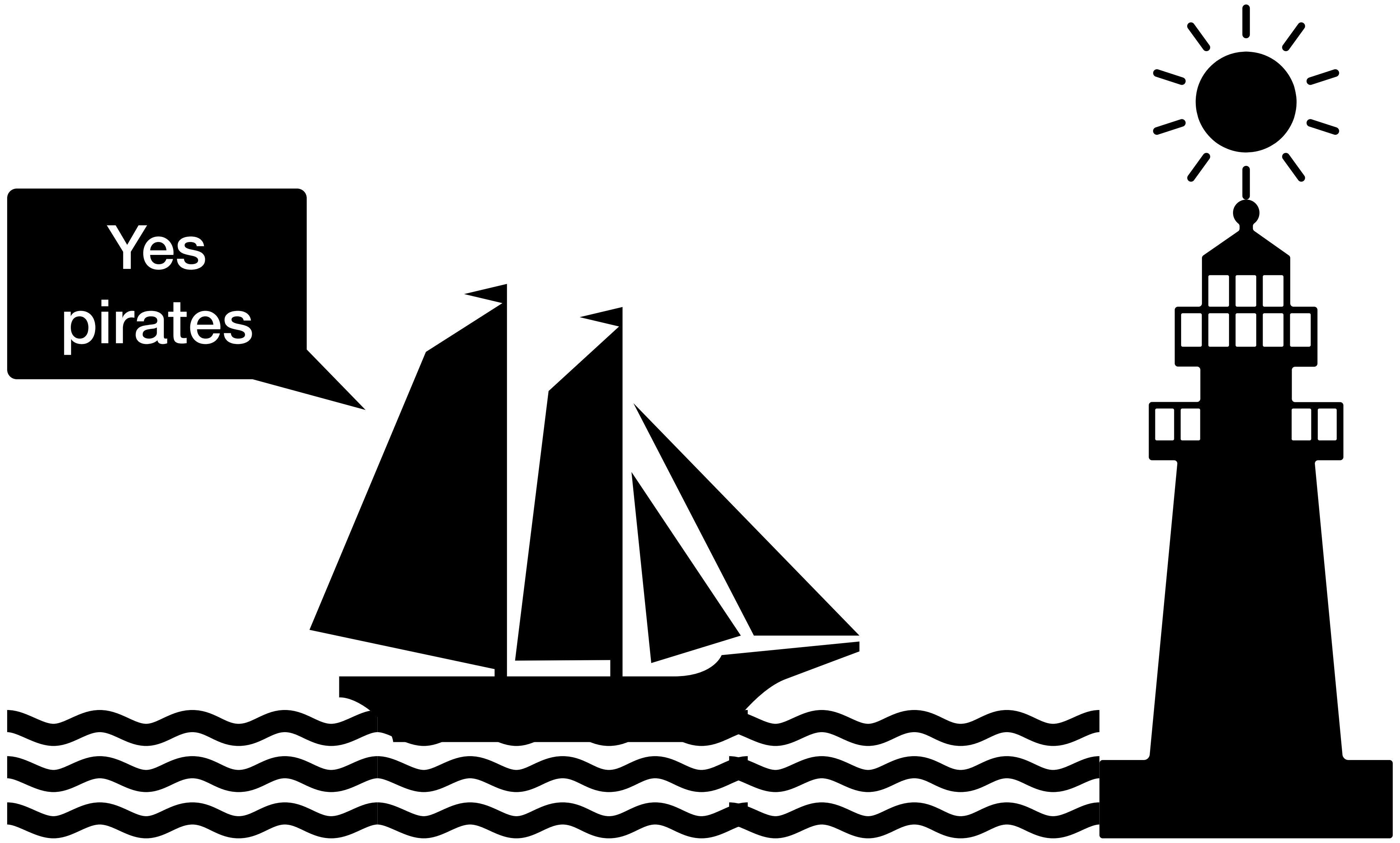
# Information

- Example: a lighthouse operator and the captain wanted to communicate if there are pirates ahead. What is the shortest message the lighthouse can send?
- They agree ahead of time:
  - light off -> no pirates



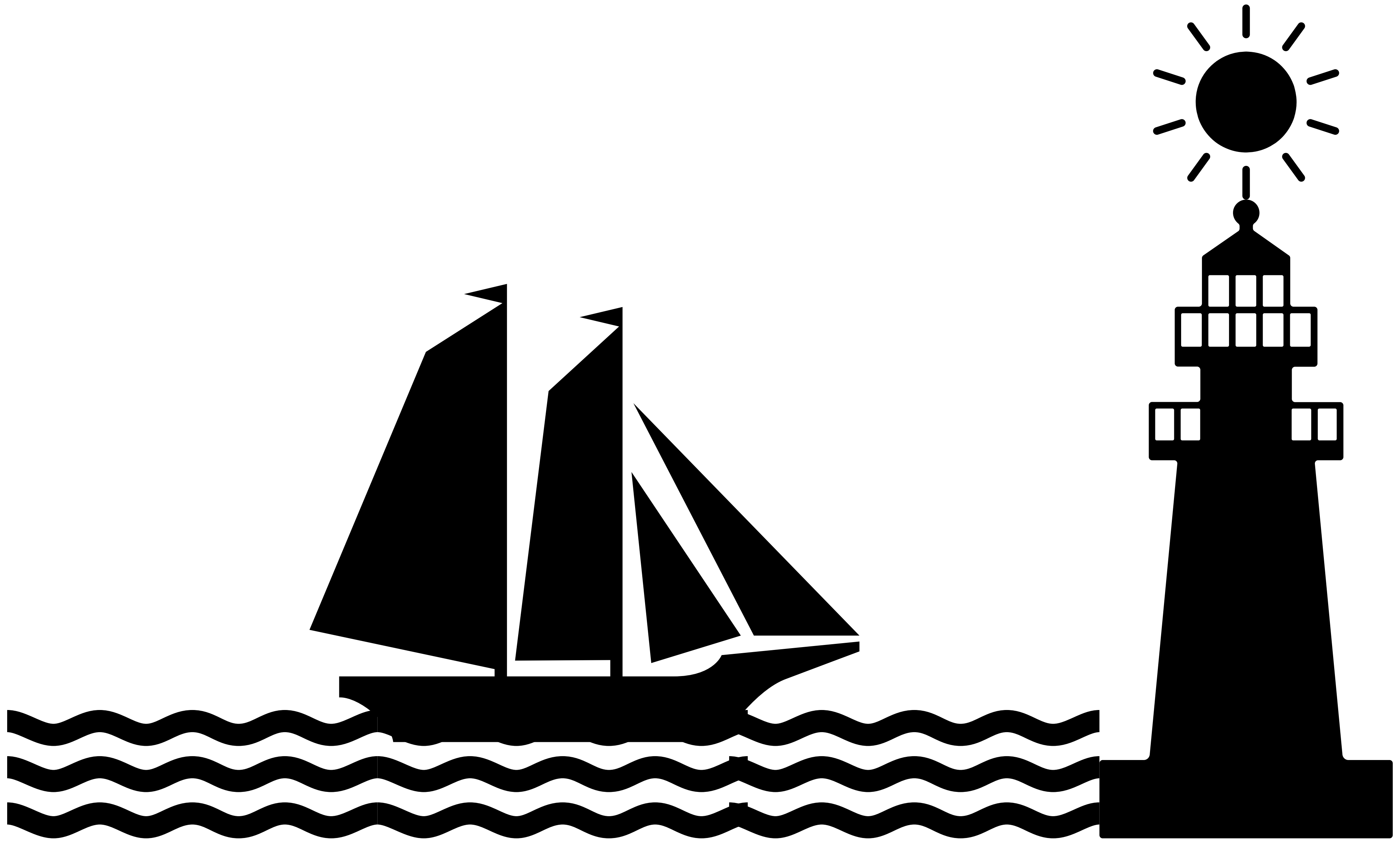
# Information

- Example: a lighthouse operator and the captain wanted to communicate if there are pirates ahead. What is the shortest message the lighthouse can send?
- They agree ahead of time:
  - light off -> no pirates
  - light on -> yes pirates



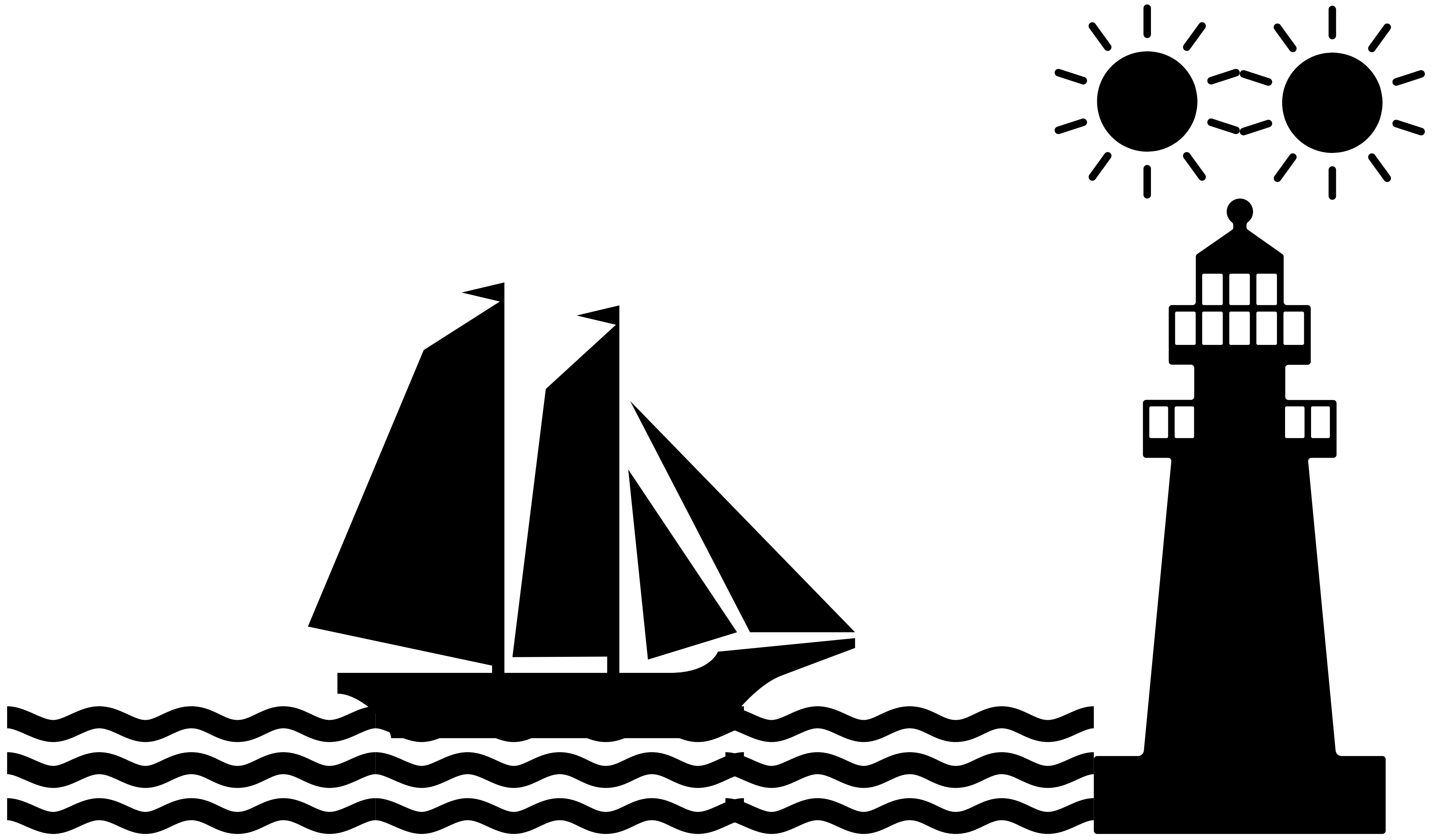
# Information

- Example: a lighthouse operator and the captain wanted to communicate if there are pirates ahead *and if there is an iceberg.*



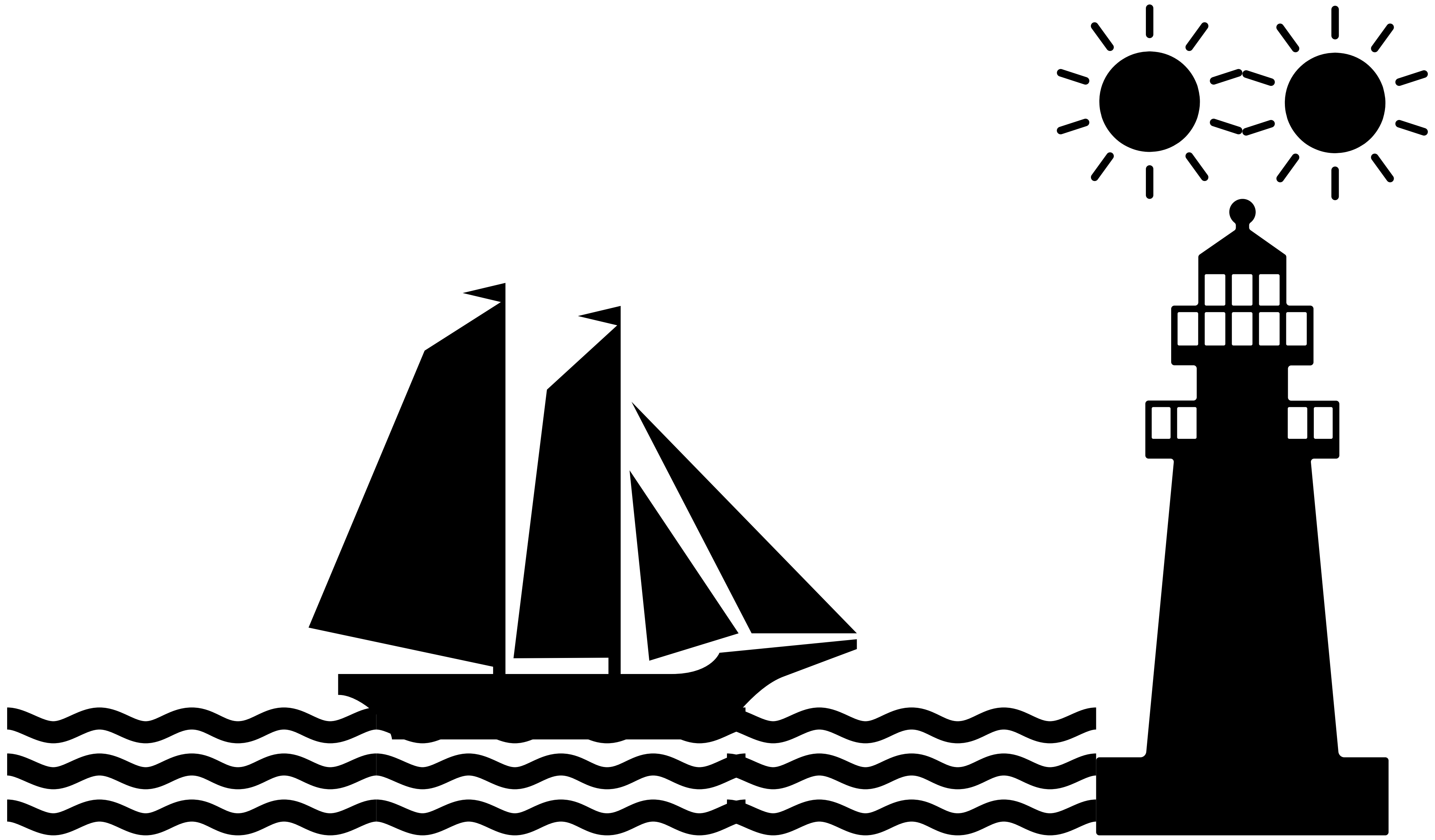
# Information

- Example: a lighthouse operator and the captain wanted to communicate if there are pirates ahead *and if there is an iceberg.*



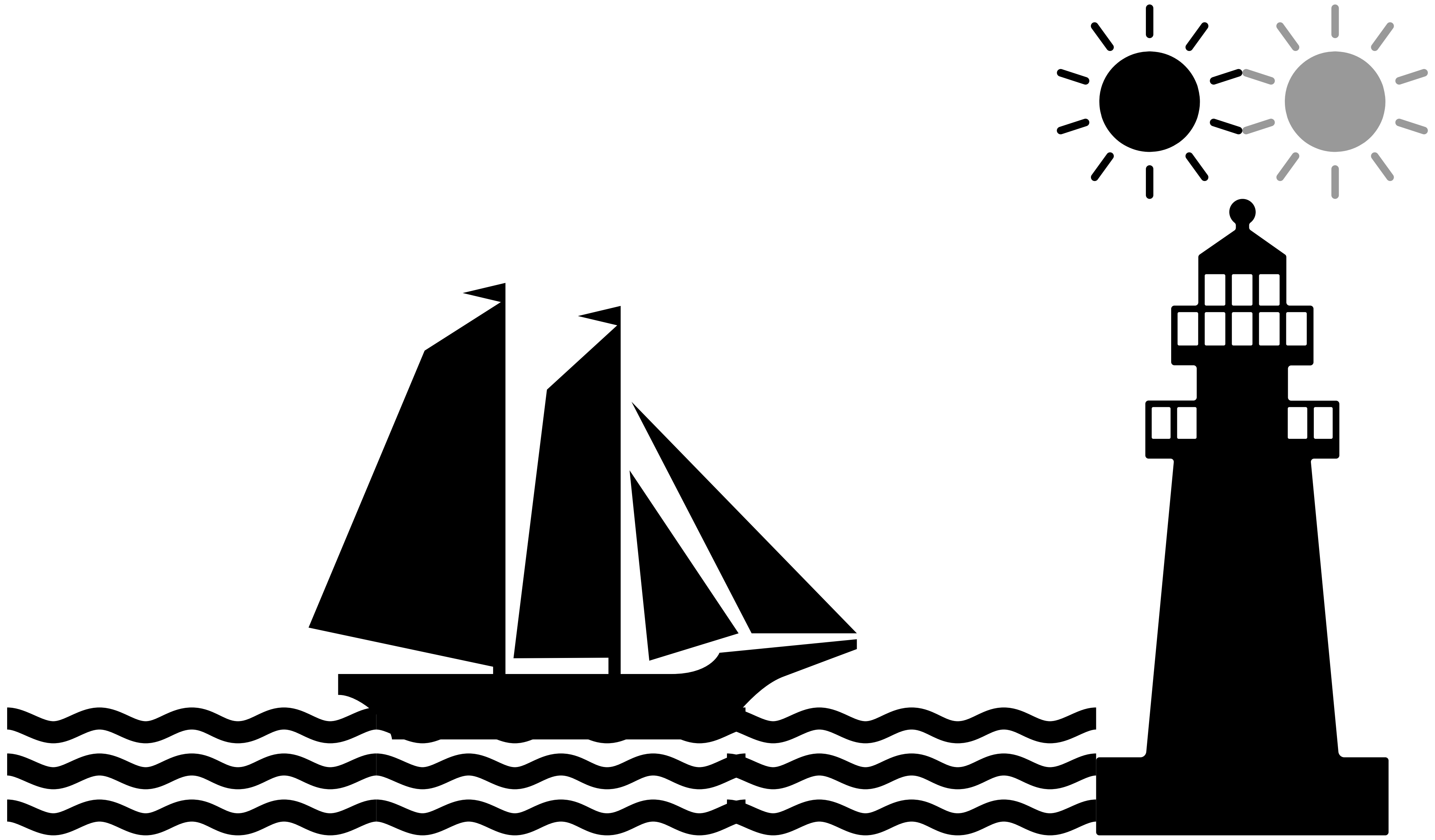
# Information

- Example: a lighthouse operator and the captain wanted to communicate if there are pirates ahead *and if there is an iceberg.*
- Agree in advance:
  - On, On -> pirates and icebergs



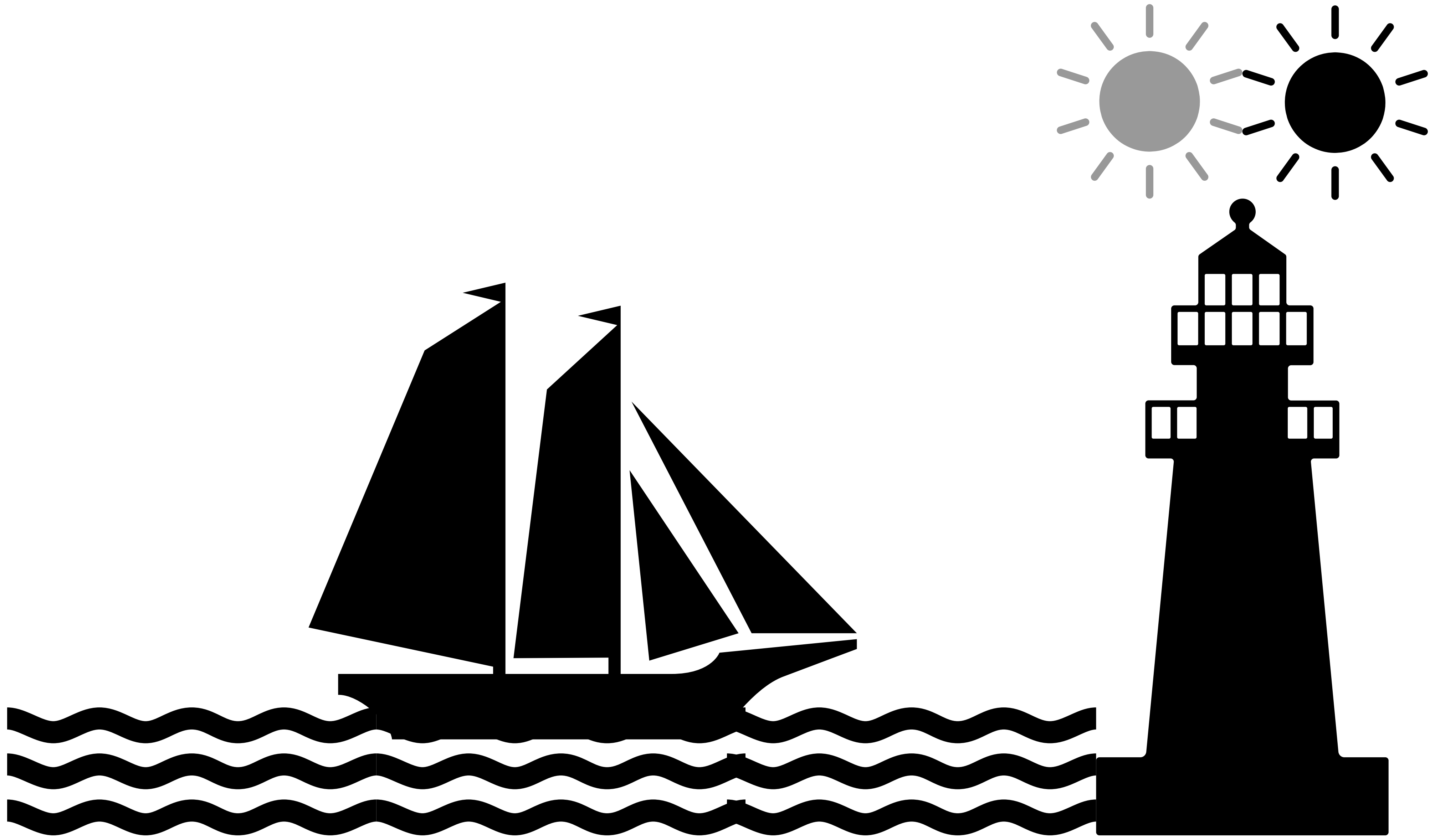
# Information

- Example: a lighthouse operator and the captain wanted to communicate if there are pirates ahead *and if there is an iceberg.*
- Agree in advance:
  - On, On -> pirates and icebergs
  - On, Off -> only pirates



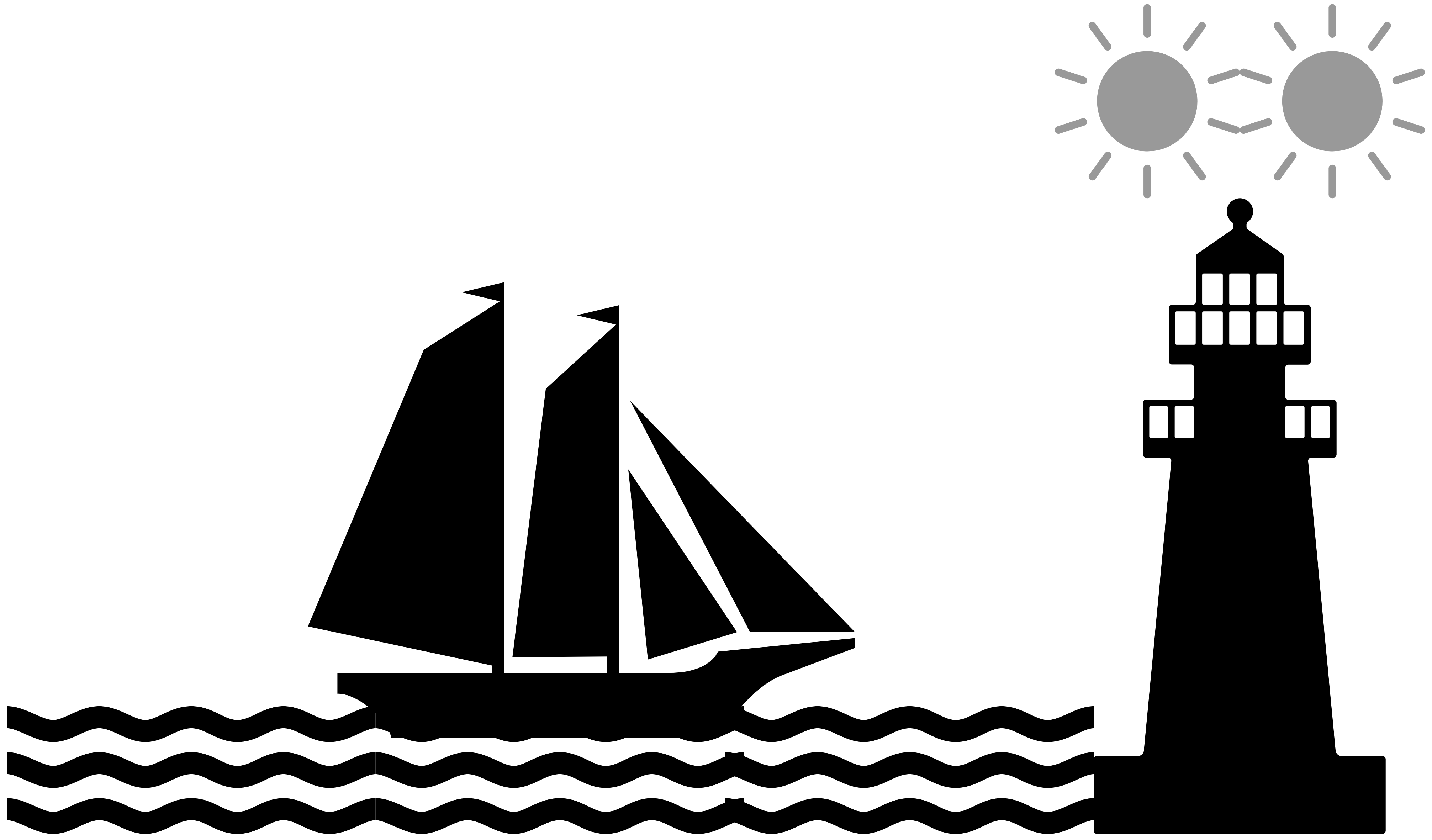
# Information

- Example: a lighthouse operator and the captain wanted to communicate if there are pirates ahead *and if there is an iceberg.*
- Agree in advance:
    - On, On -> pirates and icebergs
    - On, Off -> only pirates
    - Off, On -> only iceberg

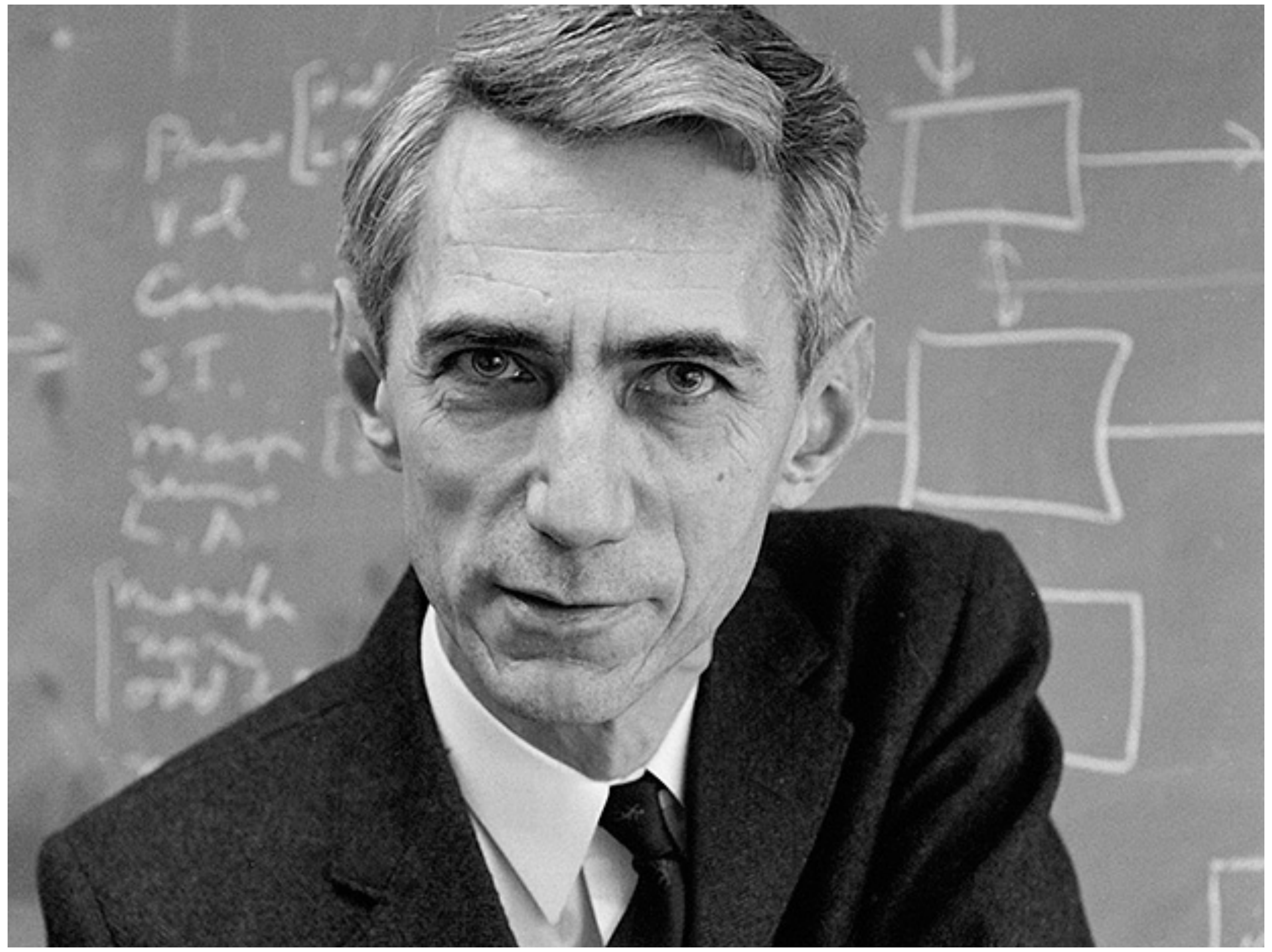


# Information

- Example: a lighthouse operator and the captain wanted to communicate if there are pirates ahead *and if there is an iceberg.*
- Agree in advance:
    - On, On -> pirates and icebergs
    - On, Off -> only pirates
    - Off, On -> only iceberg
    - Off, Off -> neither



# Information



- Claude Shannon (1916 - 2001):
- *A Mathematical Theory of Communication* (1948)

# Information

- Insight: whenever two parties communicate, each message is answering one or more yes-or-no questions.

- We call each answer *a bit*.

- Information can be measured in bits.

- This is quite profound: communication is choosing among possibilities!

# Information

- Just saying "yes" or "no" isn't enough
  - We don't know what *questions* they answer
- We have to agree ahead of time what the choices are.
- The more choices we have to make, the more answers we'll have to communicate.

# Information

## Why "yes" or "no"?

☀

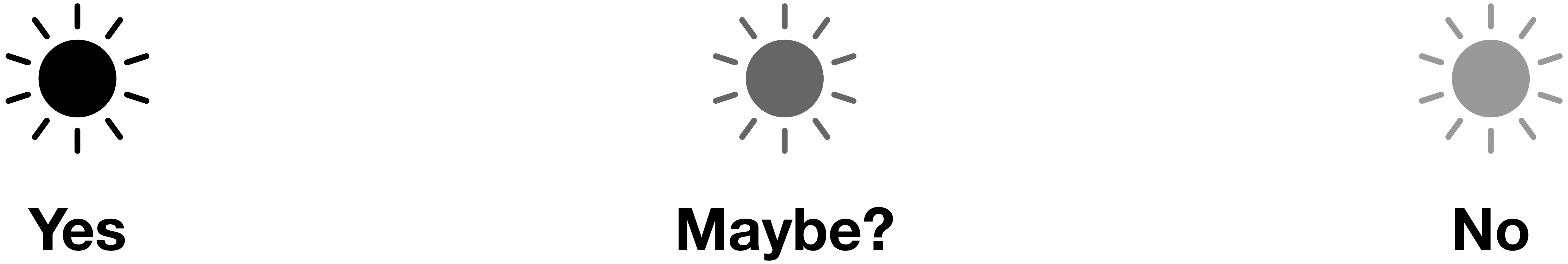
Yes

☀

No

# Information

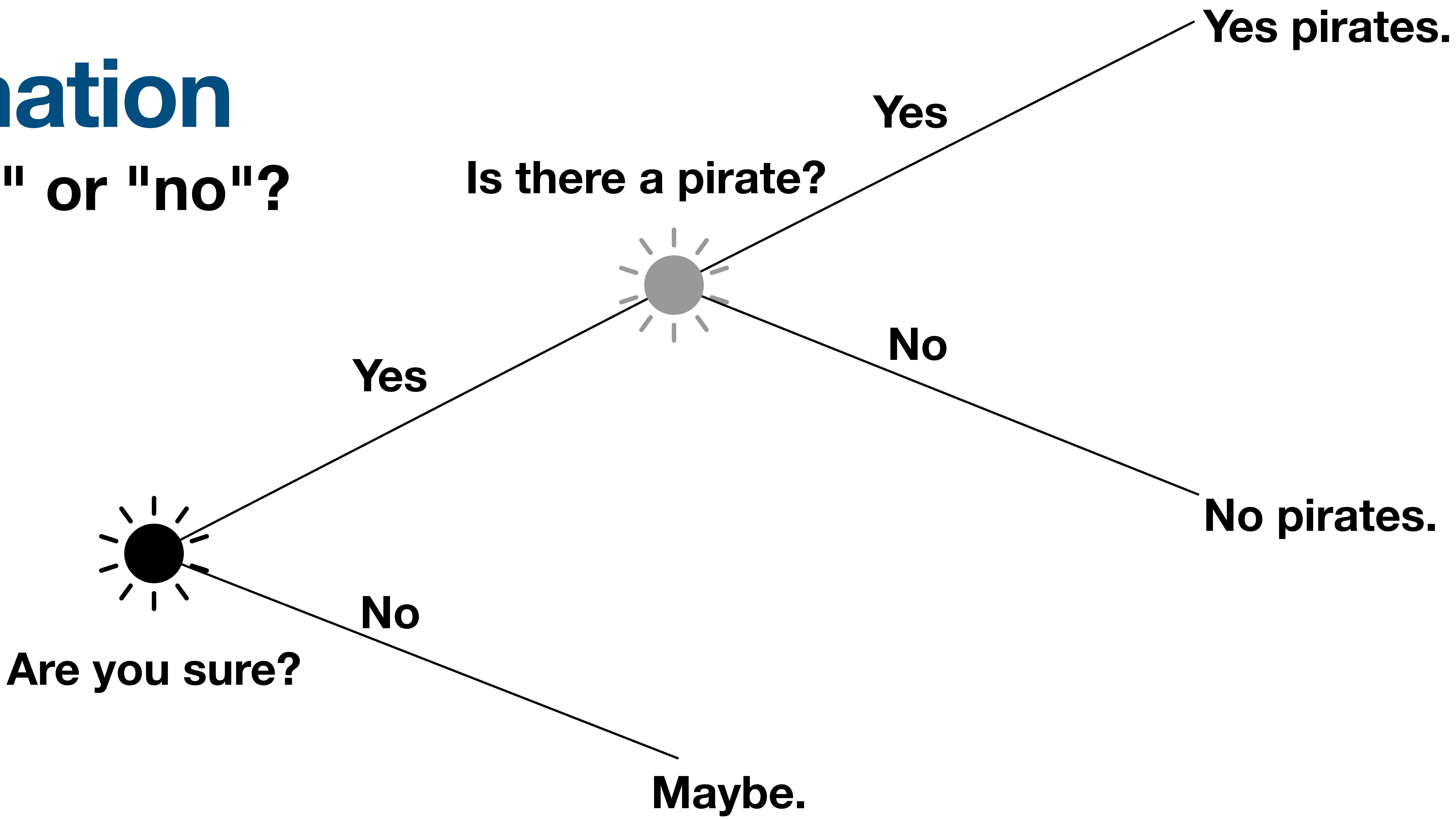
## Why "yes" or "no"?



- CS answer: "yes" or "no" is the most reduced form.

# Information

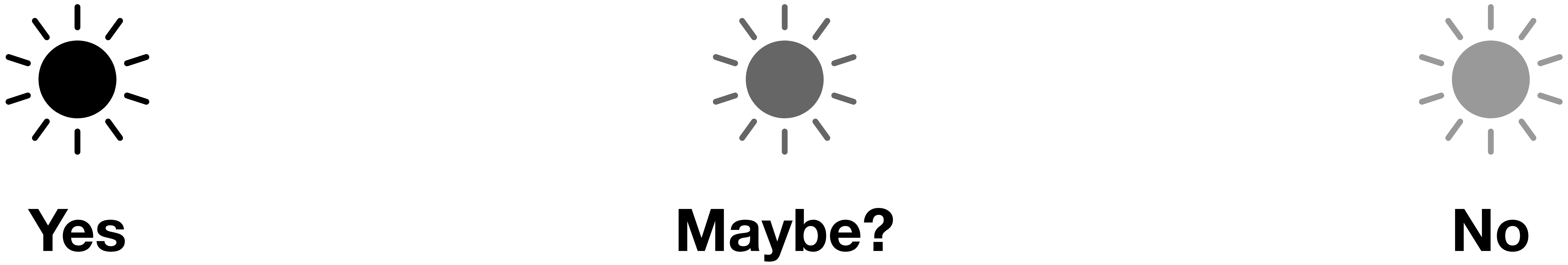
## Why "yes" or "no"?



- CS answer: "yes" or "no" is the most reduced form.

# Information

## Why "yes" or "no"?



Yes

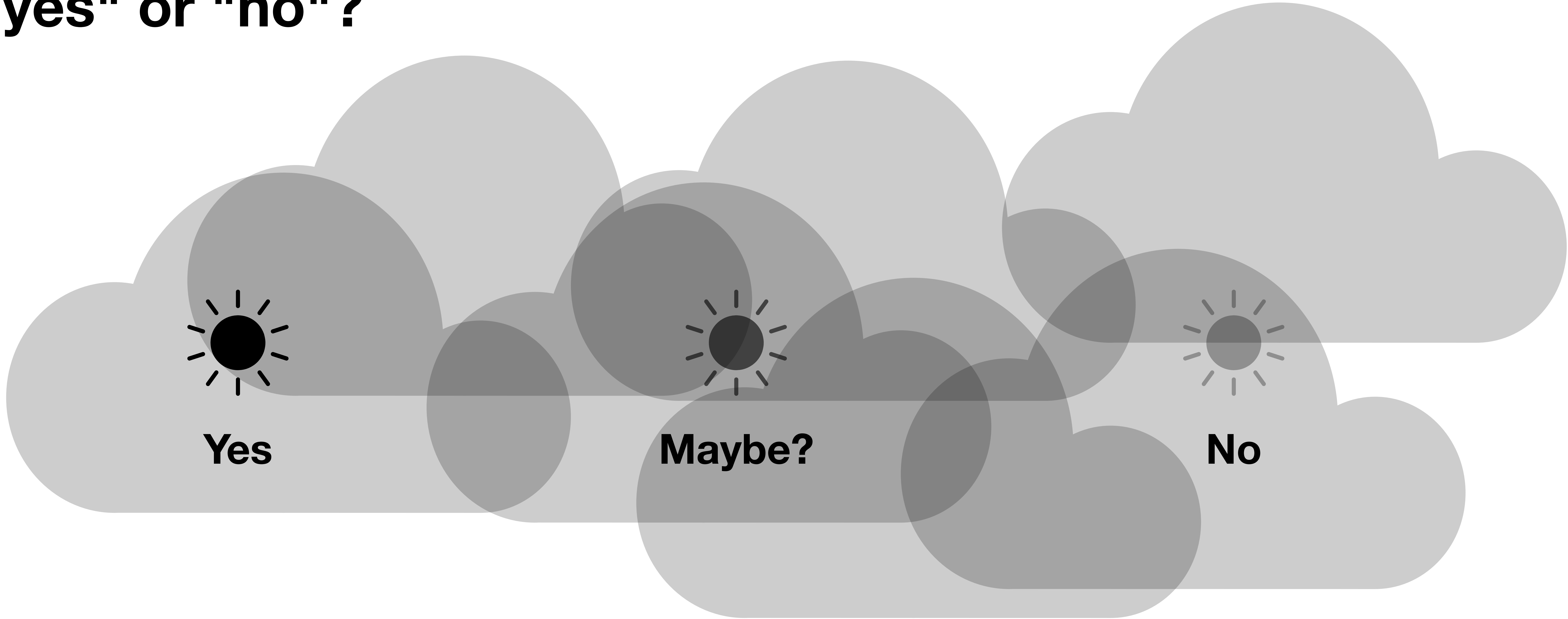
Maybe?

No

- EE answer: the two options can survive a lot of noise.

# Information

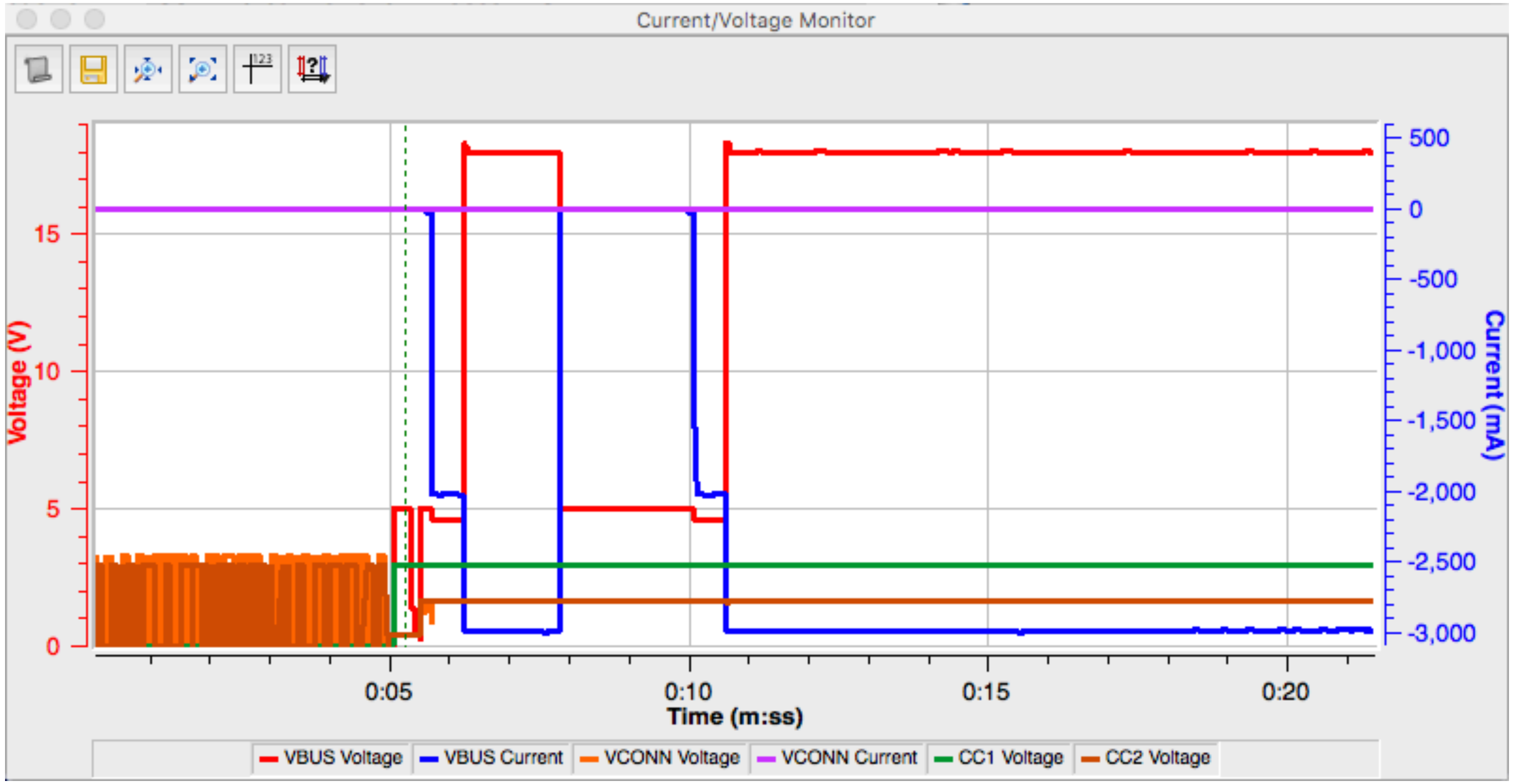
## Why "yes" or "no"?



- EE answer: the two options can survive a lot of noise.

# Information

## Why "yes" or "no"?

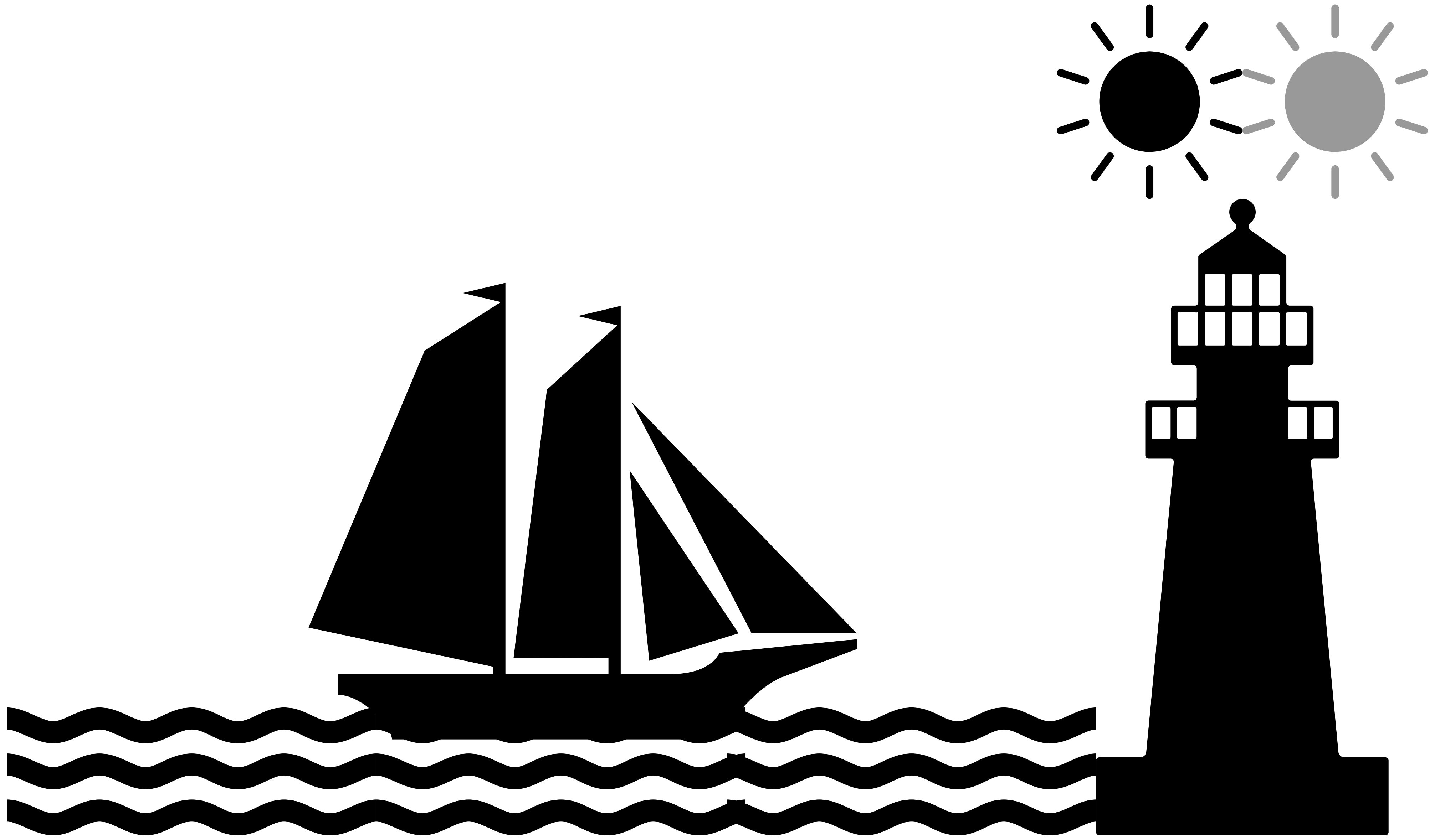


- EE answer: the two options can survive a lot of noise.

# Information

## Protocol

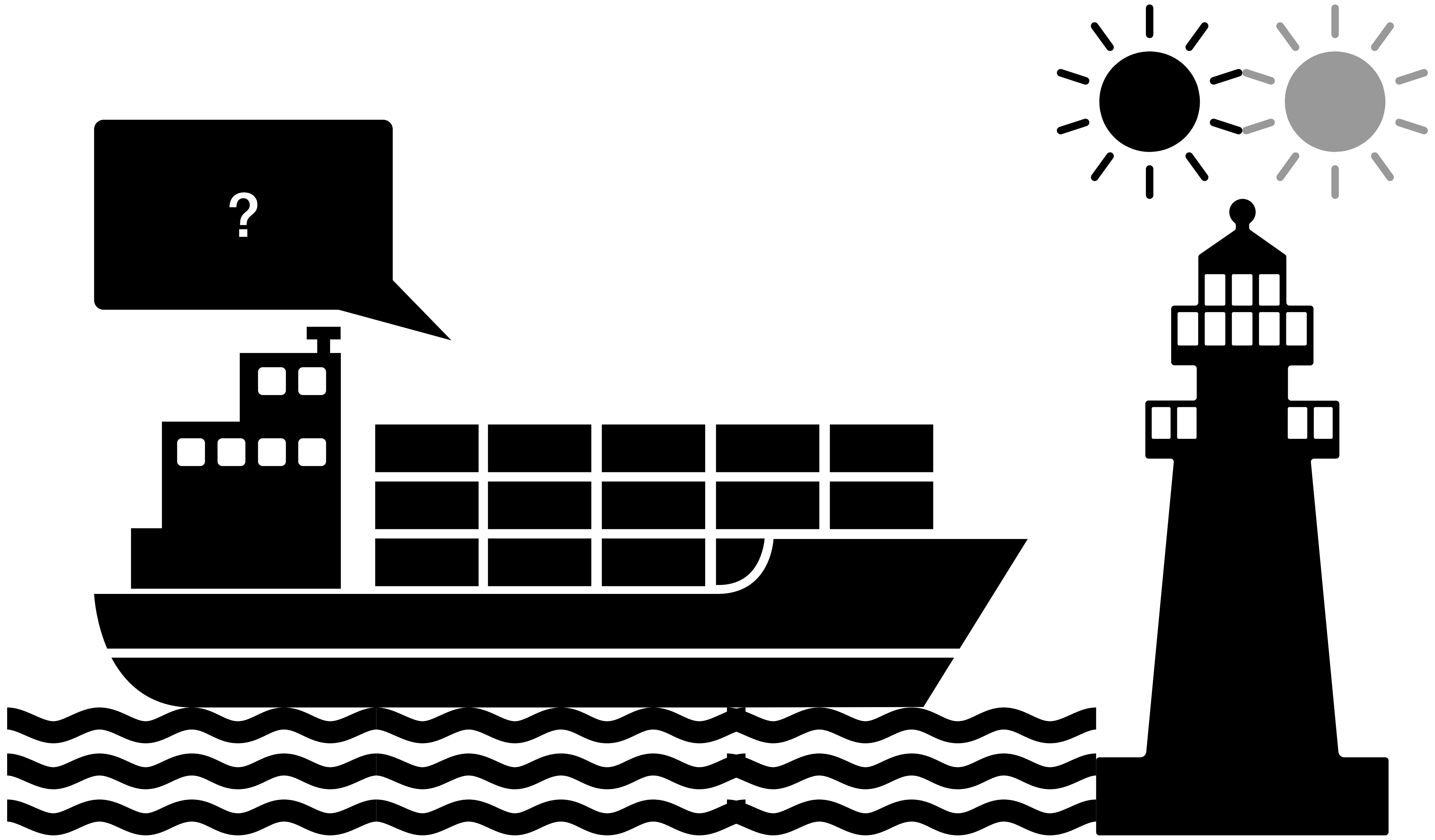
- The captain and the lighthouse *must* agree on the choices in advance.
  - How many choices?
  - Which light answers which?



# Information

## Protocol

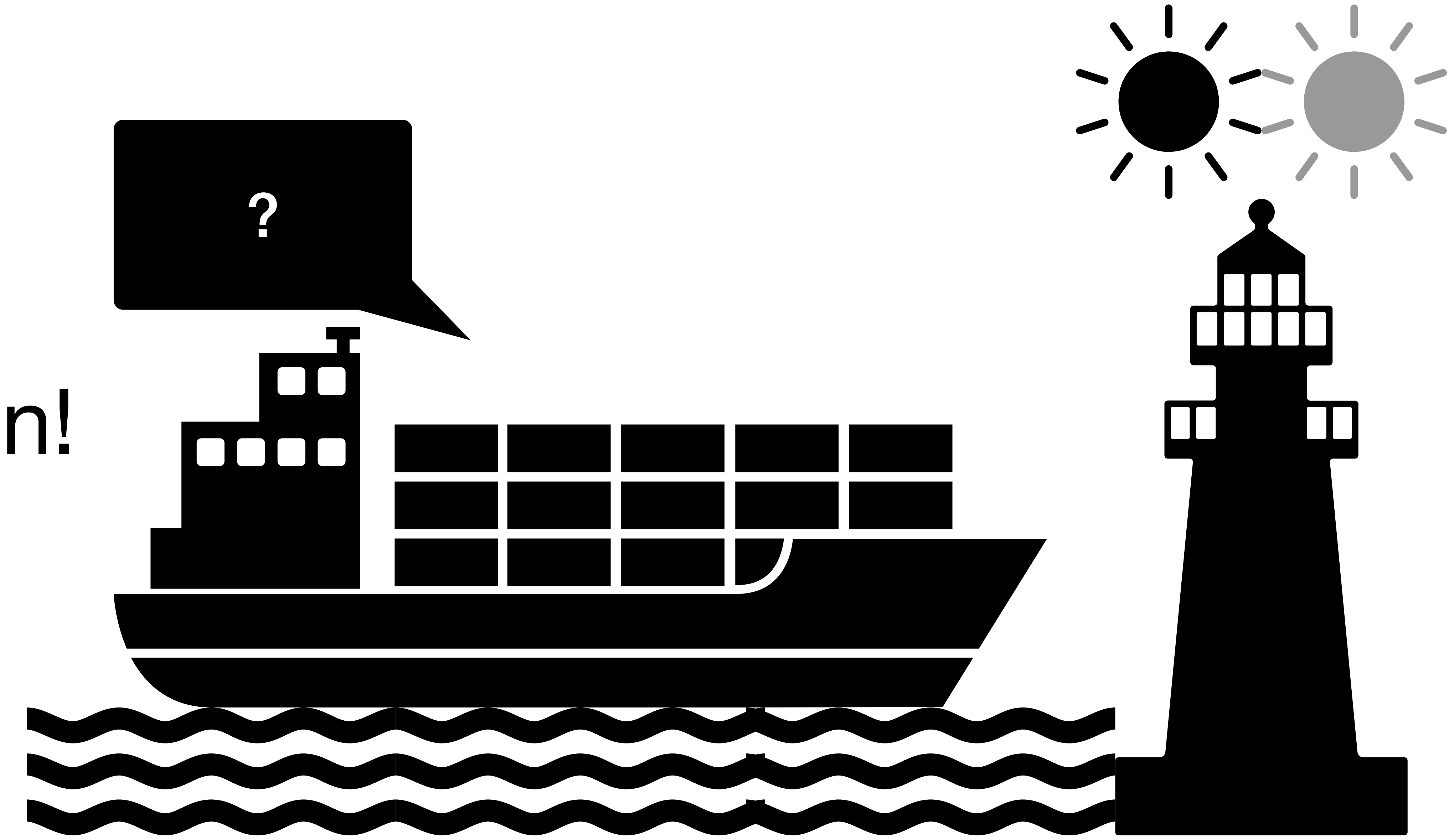
- The captain and the lighthouse *must* agree on the choices in advance.
  - How many choices?
  - Which light answers which?



# Information

## Protocol

- The captain and the lighthouse *must* agree on the choices in advance.
    - How many choices?
    - Which light answers which?
- If you don't know the context, you don't know what the bits mean!
- This is why we need file format, types, protocols, ...



# Information

How many bits do you need to send Rick Astley's *Never Gonna Give You Up*?

It depends on how many choices there are!

# Information

How many bits do you need to send Rick Astley's *Never Gonna Give You Up*?

It depends on how many choices there are!

# Information

How many bits do you need to send Rick Astley's *Never Gonna Give You Up*?

It depends on how many choices there are!

# Information

How many bits do you need to send Rick Astley's *Never Gonna Give You Up*?

☀ ☼ ☀

☀ ☀ ☀

☼ ☼ ☀

☼ ☀ ☼

☼ ☀ ☀

It depends on how many choices there are!

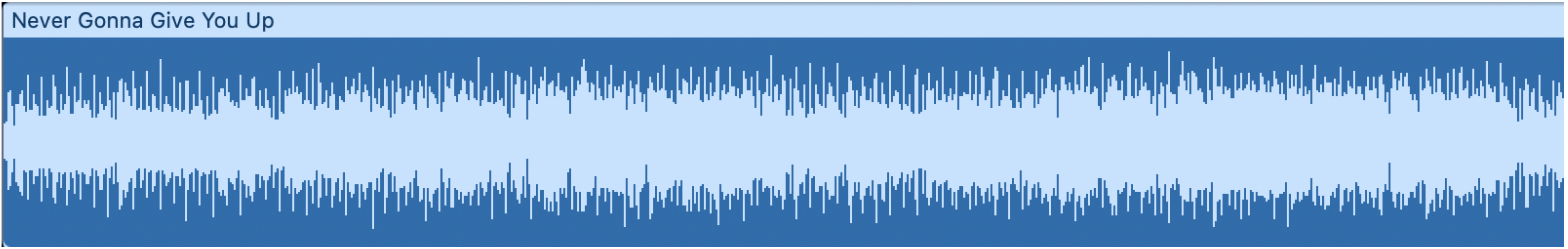
# Information

How many bits do you need to send Rick Astley's *Never Gonna Give You Up*?

What if you want to choose between all music?

# Information

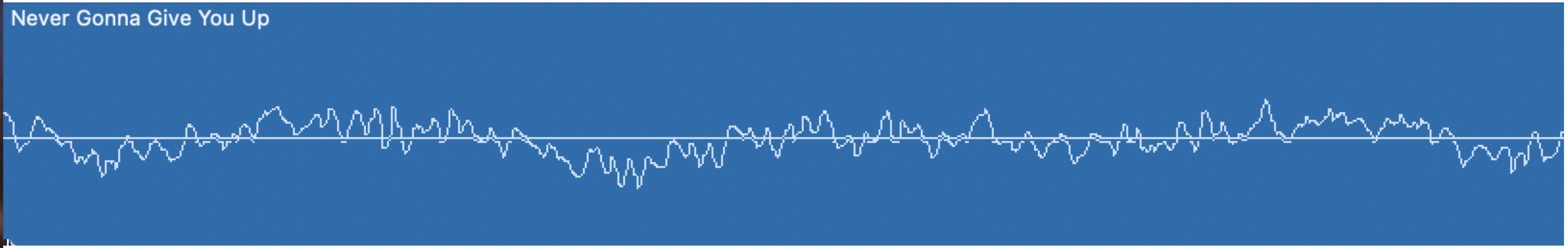
How many bits do you need to send Rick Astley's *Never Gonna Give You Up*?



What if you want to choose between all music?

# Information

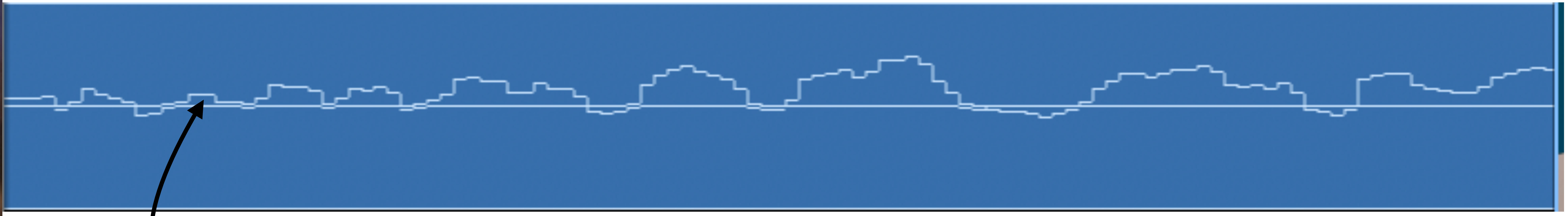
How many bits do you need to send Rick Astley's *Never Gonna Give You Up*?



What if you want to choose between all music?

# Information

How many bits do you need to send Rick Astley's *Never Gonna Give You Up*?

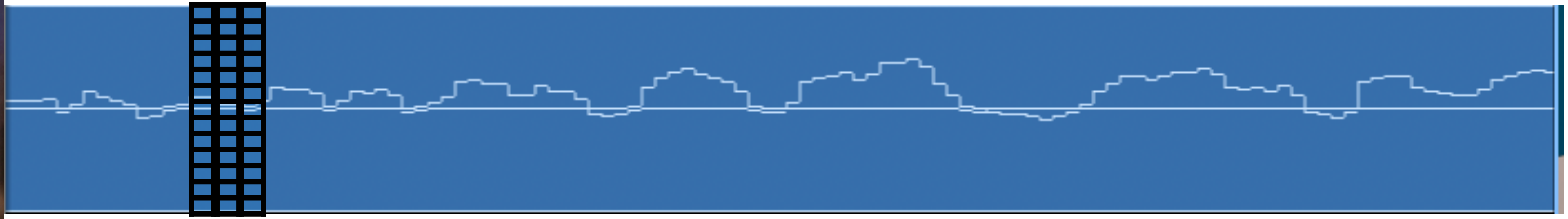


The microphone samples the current sound wave many times a second

What if you want to choose between all music?

# Information

How many bits do you need to send Rick Astley's *Never Gonna Give You Up*?



Each sample is one of several choices of magnitude

What if you want to choose between all music?

# Information

How many bits do you need to send Rick Astley's *Never Gonna Give You Up*?

- CD quality:
  - 16 bit per sample (65536 choices of magnitudes!)
  - 44100 Hz from 2 channels.
  - 16 * 44100 * 2 = 1,411,200 bits/second
- Length: 214 seconds
  - 301,996,800 bits!

What if you want to choose between all music?

# Information

## Bits

- A bit is an answer to a yes-no question
- We can model information as a series of yes-no questions
- In order to communicate in bits, we must know:
    - what the questions are;
    - which bit answers which question.
- If we have $n$ bits, we have $2^n$ choices to communicate.
- If we have $n$ choices to communicate, we need $\lceil \log_2 n \rceil$ bits.

# Binary

## Let's talk numbers

- To communicate numbers, just like other information, we can encode numbers as bits!

# Binary

## Let's talk numbers

| Number | Encoding |
|---|---|
| 0 | |
| 1 | |
| 2 | |
| 3 | |
| 4 | |
| 5 | |
| 6 | |
| 7 | |

# Binary

## Let's talk numbers

| Number | Encoding |
|---|---|
| 0 | ○ ○ ○ |
| 1 | ○ ○ ● |
| 2 | ○ ● ○ |
| 3 | ○ ● ● |
| 4 | ● ○ ○ |
| 5 | ● ○ ● |
| 6 | ● ● ○ |
| 7 | ● ● ● |

# Binary

## Let's talk numbers

| Number | Encoding |
|---|---|
| **0** | [no, no, no] |
| **1** | [no, no, yes] |
| **2** | [no, yes, no] |
| **3** | [no, yes, yes] |
| **4** | [yes, no, no] |
| **5** | [yes, no, yes] |
| **6** | [yes, yes, no] |
| **7** | [yes, yes, yes] |

# Binary

## Let's talk numbers

Wait, we already have math behind this!

| Number | Encoding |
|---|---|
| 0 | [0, 0, 0] |
| 1 | [0, 0, 1] |
| 2 | [0, 1, 0] |
| 3 | [0, 1, 1] |
| 4 | [1, 0, 0] |
| 5 | [1, 0, 1] |
| 6 | [1, 1, 0] |
| 7 | [1, 1, 1] |

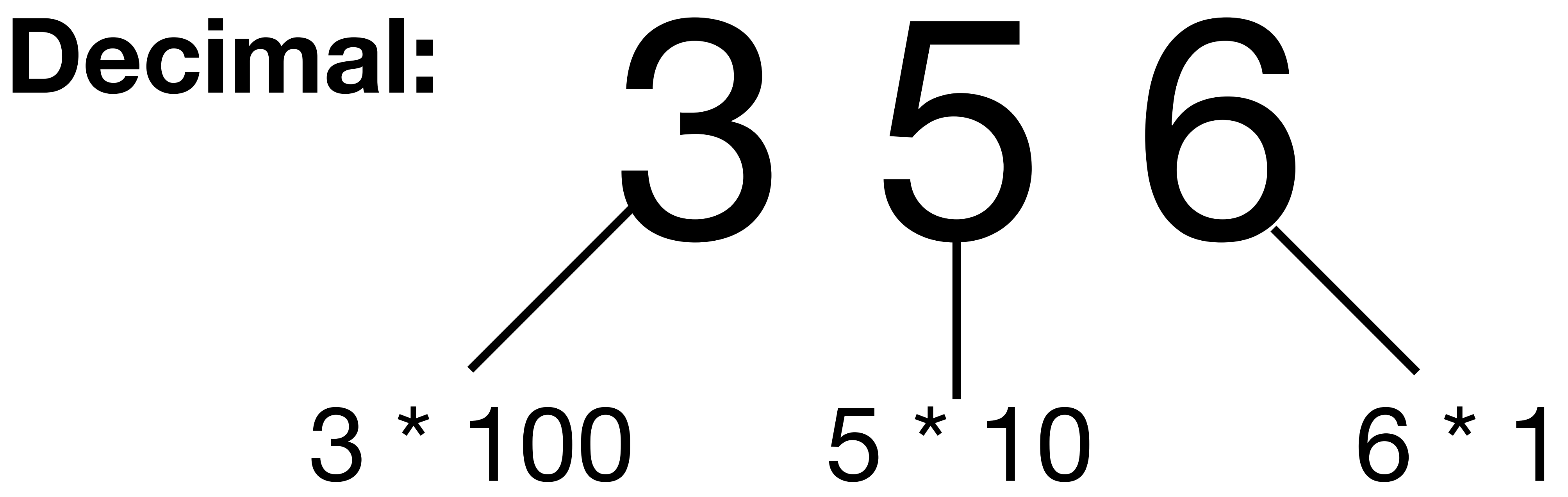
# Binary

## Decimal number with harder math

**Decimal:** 3 5 6

# Binary

## Decimal number with harder math



Decimal: 3 5 6

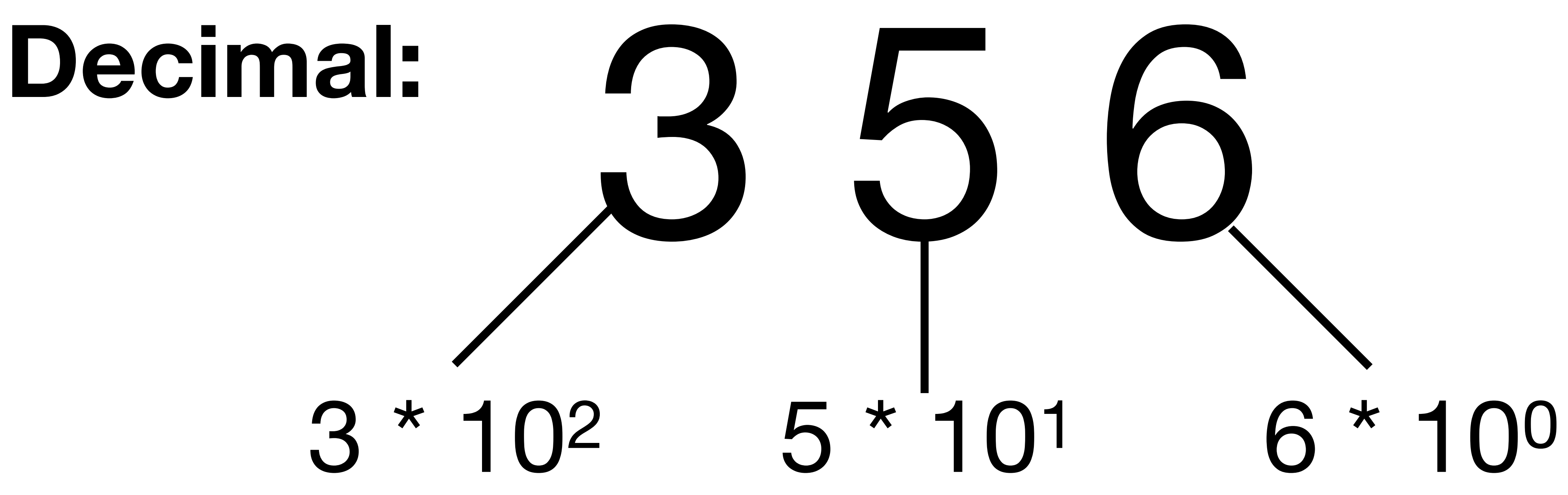
3 * 100

5 * 10

6 * 1

# Binary

## Decimal number with harder math

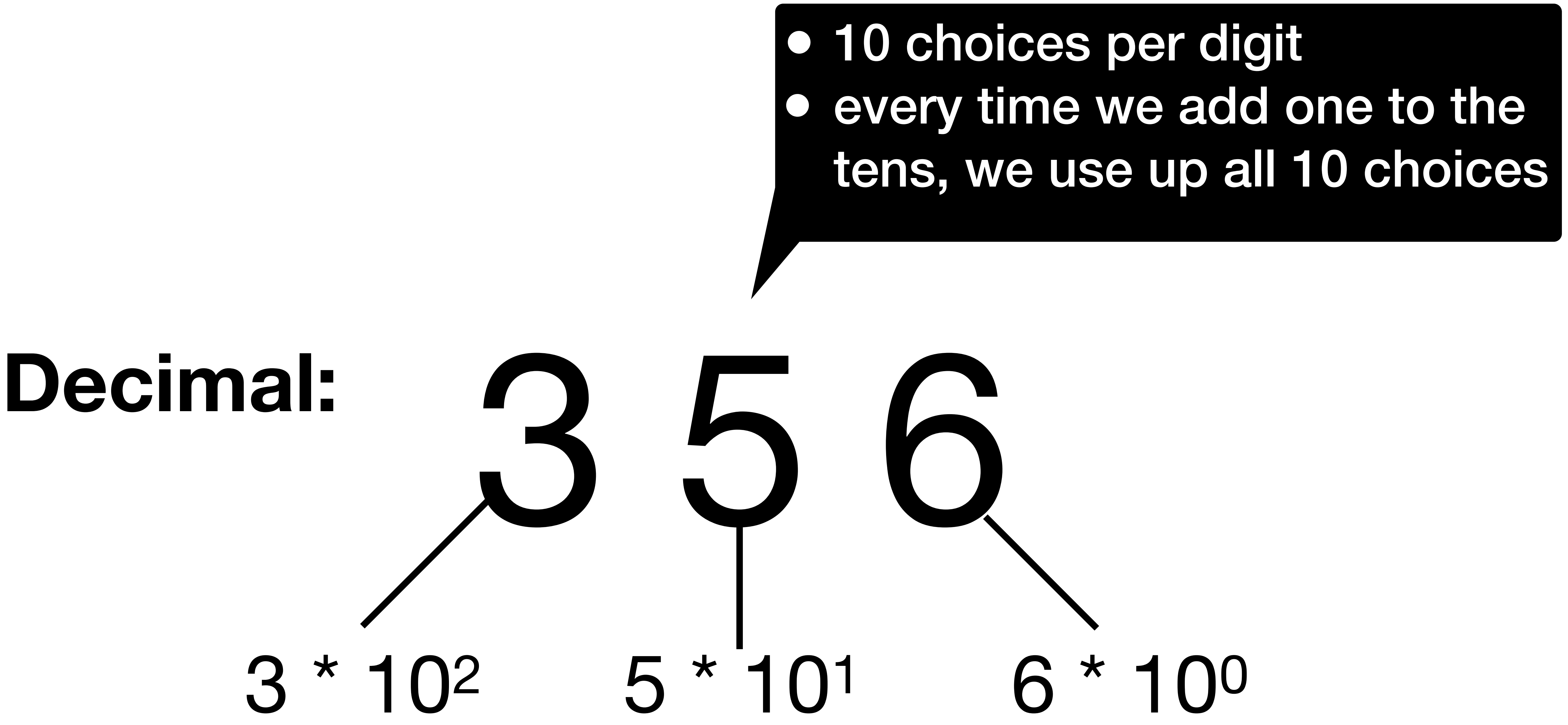


**Decimal:** 3 5 6

$3 * 10^2$ $5 * 10^1$ $6 * 10^0$

# Binary

## Decimal number with harder math



- 10 choices per digit
- every time we add one to the tens, we use up all 10 choices

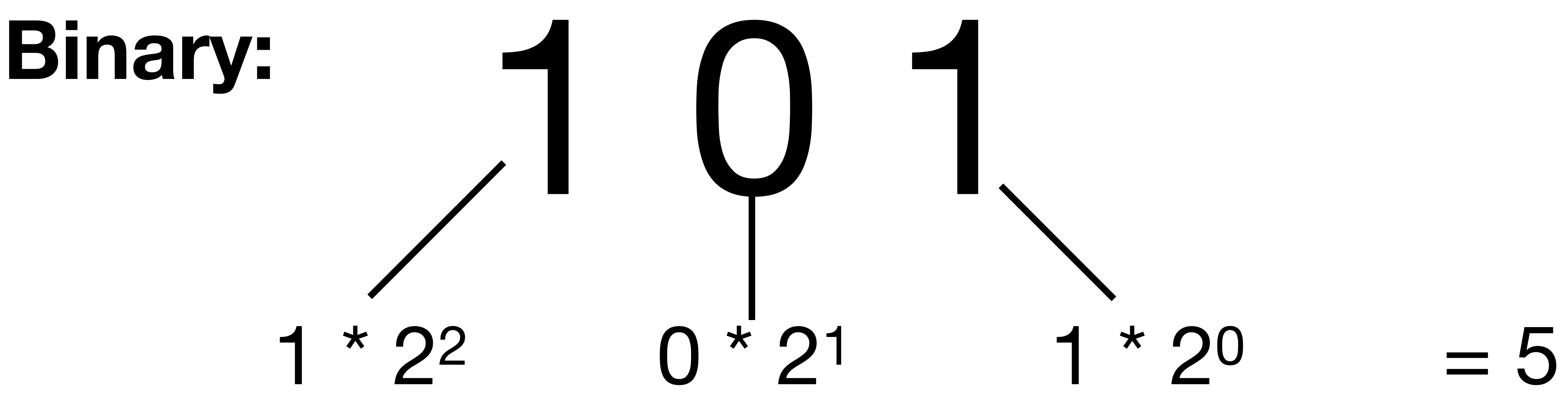
**Decimal:** 3 5 6

$3 * 10^2$ $5 * 10^1$ $6 * 10^0$

# Binary

## Decimal number with harder math

- 2 choices per digit
- every time we add one to the tens, we use up all 2 choices



**Binary:** 1 0 1

$1 * 2^2$    $0 * 2^1$    $1 * 2^0$    = 5

# Binary

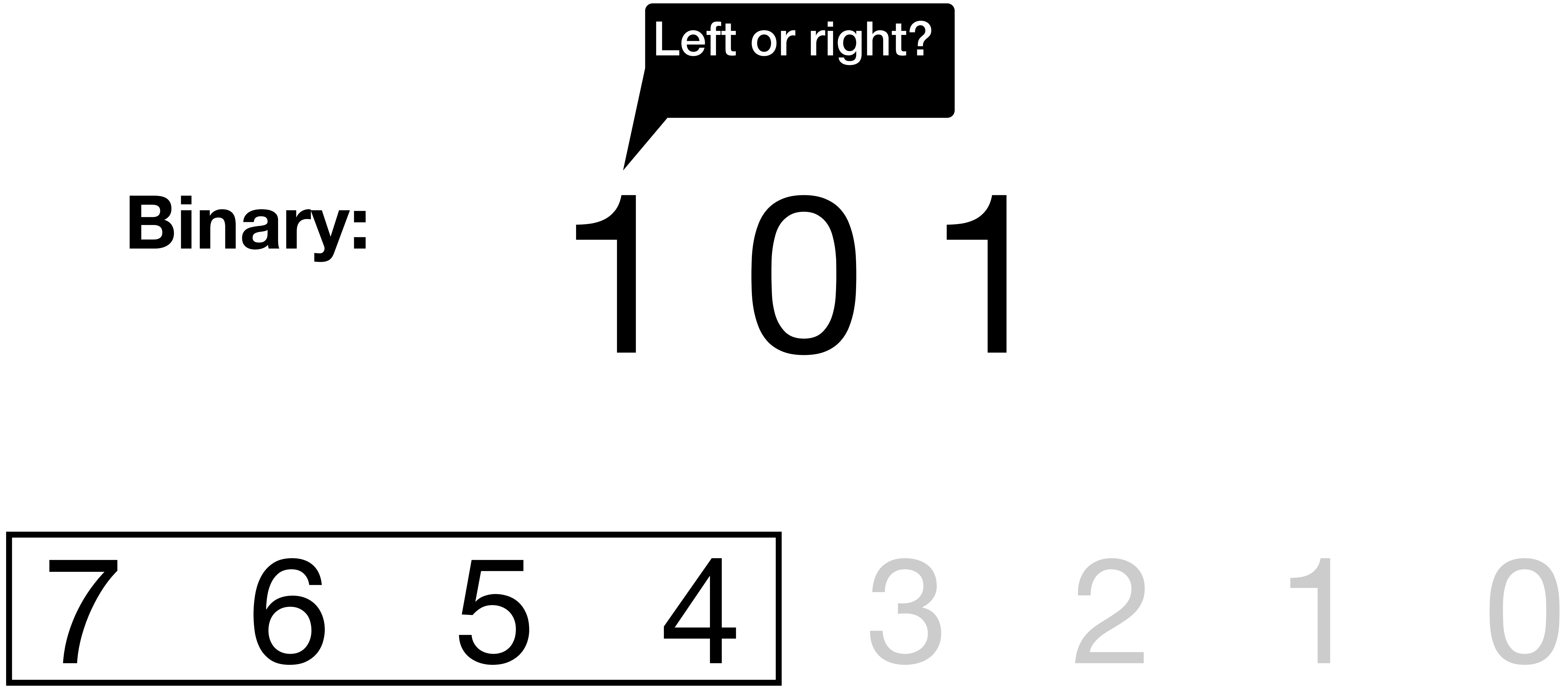
## Binary-searching the list of choices

**Binary:** 1 0 1

7 6 5 4 3 2 1 0

# Binary

## Binary-searching the list of choices



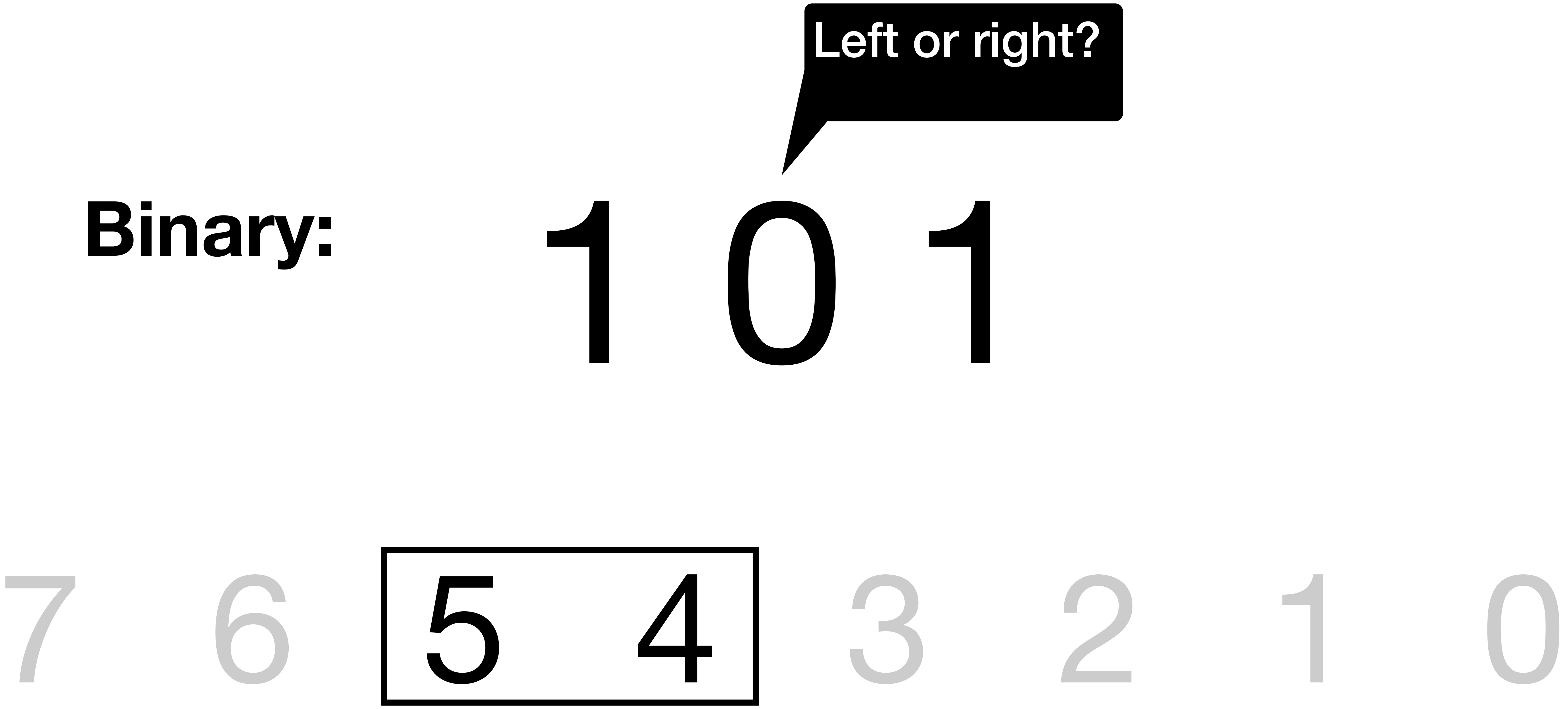
Left or right?

Binary: 1 0 1

7 6 5 4 3 2 1 0

# Binary

## Binary-searching the list of choices



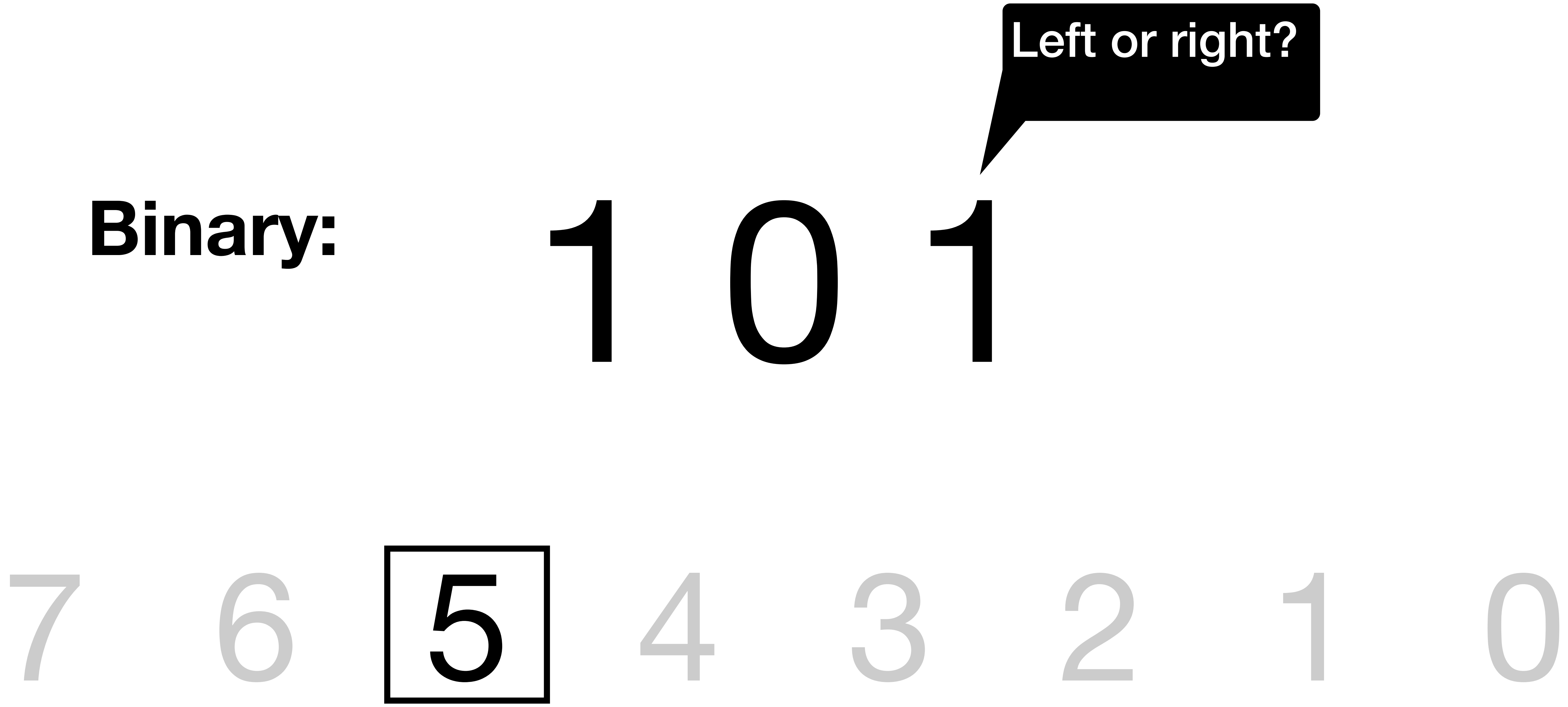
Binary: 1 0 1

Left or right?

7 6 $\boxed{5 \quad 4}$ 3 2 1 0

# Binary

## Binary-searching the list of choices



Binary: 1 0 1

Left or right?

7 6 5 4 3 2 1 0

# Binary

## Two ways

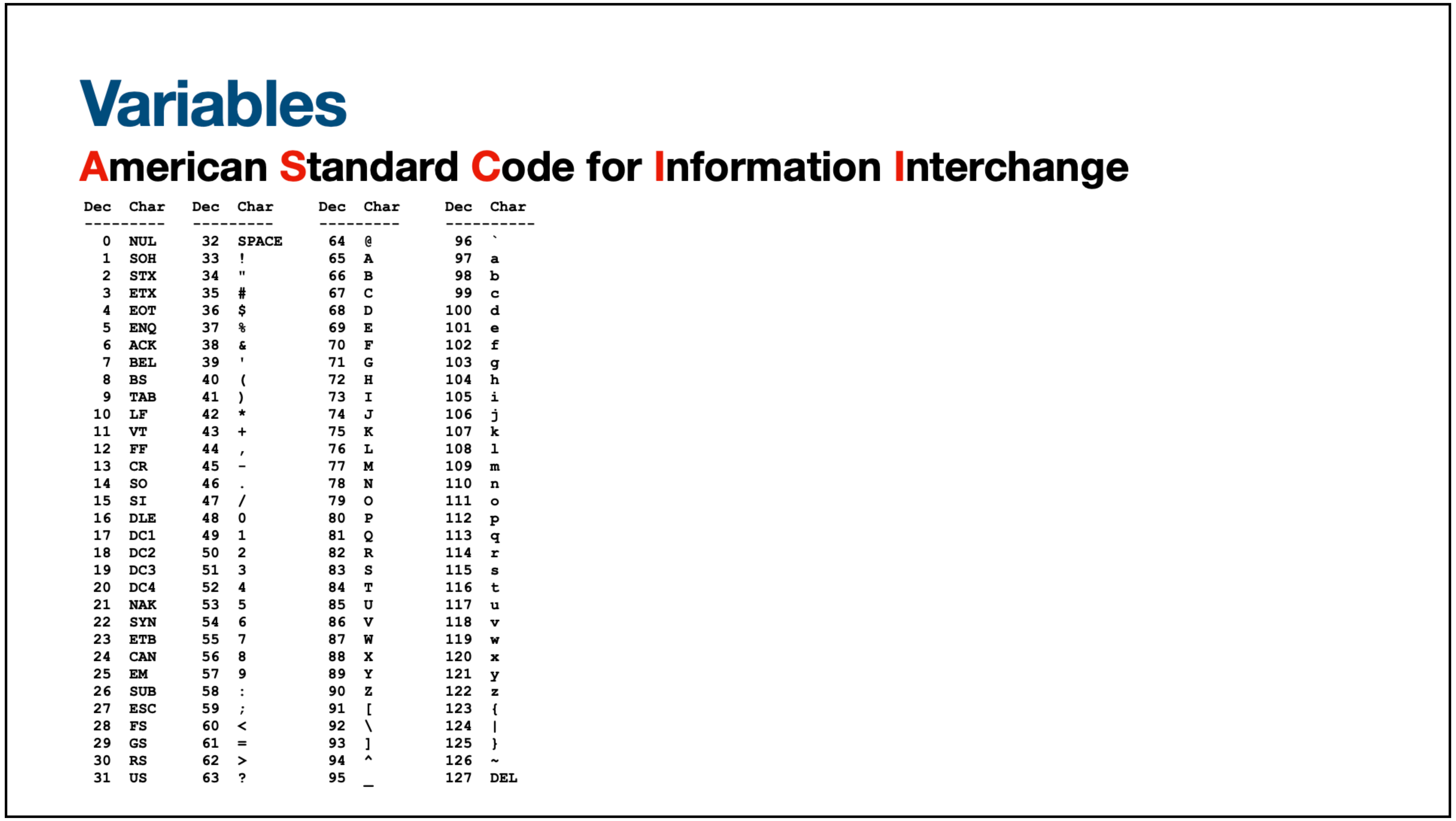
**Binary:** 1 0 1

$1 * 2^2$ $0 * 2^1$ $1 * 2^0$

7 6 $\boxed{5}$ 4 3 2 1 0

# Binary

## How about texts?



### Variables

**American Standard Code for Information Interchange**

| Dec | Char | Dec | Char | Dec | Char | Dec | Char |
|---|---|---|---|---|---|---|---|
| 0 | NUL | 32 | SPACE | 64 | @ | 96 | ` |
| 1 | SOH | 33 | ! | 65 | A | 97 | a |
| 2 | STX | 34 | " | 66 | B | 98 | b |
| 3 | ETX | 35 | # | 67 | C | 99 | c |
| 4 | EOT | 36 | $ | 68 | D | 100 | d |
| 5 | ENQ | 37 | % | 69 | E | 101 | e |
| 6 | ACK | 38 | & | 70 | F | 102 | f |
| 7 | BEL | 39 | ' | 71 | G | 103 | g |
| 8 | BS | 40 | ( | 72 | H | 104 | h |
| 9 | TAB | 41 | ) | 73 | I | 105 | i |
| 10 | LF | 42 | * | 74 | J | 106 | j |
| 11 | VT | 43 | + | 75 | K | 107 | k |
| 12 | FF | 44 | , | 76 | L | 108 | l |
| 13 | CR | 45 | - | 77 | M | 109 | m |
| 14 | SO | 46 | . | 78 | N | 110 | n |
| 15 | SI | 47 | / | 79 | O | 111 | o |
| 16 | DLE | 48 | 0 | 80 | P | 112 | p |
| 17 | DC1 | 49 | 1 | 81 | Q | 113 | q |
| 18 | DC2 | 50 | 2 | 82 | R | 114 | r |
| 19 | DC3 | 51 | 3 | 83 | S | 115 | s |
| 20 | DC4 | 52 | 4 | 84 | T | 116 | t |
| 21 | NAK | 53 | 5 | 85 | U | 117 | u |
| 22 | SYN | 54 | 6 | 86 | V | 118 | v |
| 23 | ETB | 55 | 7 | 87 | W | 119 | w |
| 24 | CAN | 56 | 8 | 88 | X | 120 | x |
| 25 | EM | 57 | 9 | 89 | Y | 121 | y |
| 26 | SUB | 58 | : | 90 | Z | 122 | z |
| 27 | ESC | 59 | ; | 91 | [ | 123 | { |
| 28 | FS | 60 | < | 92 | \ | 124 | \| |
| 29 | GS | 61 | = | 93 | ] | 125 | } |
| 30 | RS | 62 | > | 94 | ^ | 126 | ~ |
| 31 | US | 63 | ? | 95 | _ | 127 | DEL |

# Digitalization

- Software structures model *real world* objects and concepts:
  - Numbers
  - Texts
  - Images
  - Sound recordings
  - Students
  - Bank statements
  - etc.
- These aren't bits, but we agree on which choices we care about and which bit patterns to present them

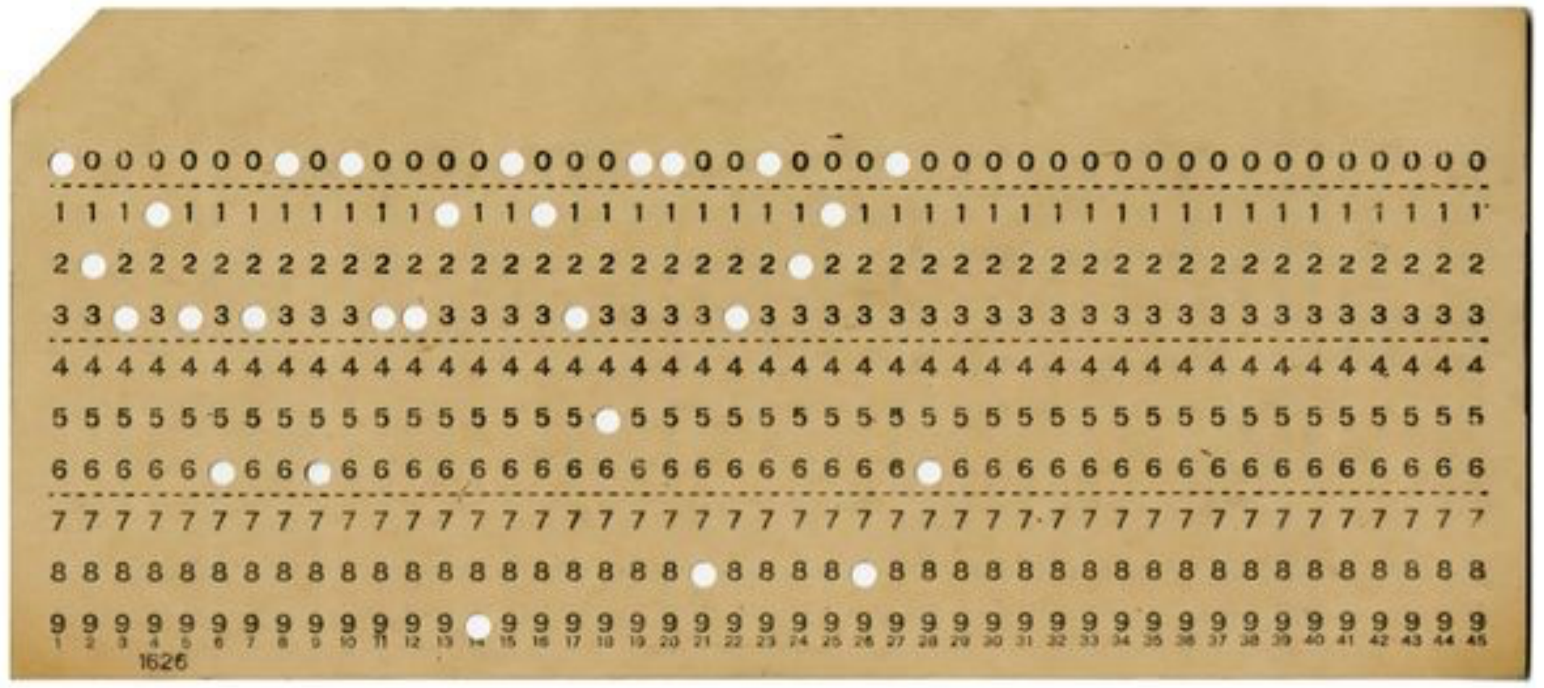
# Bits

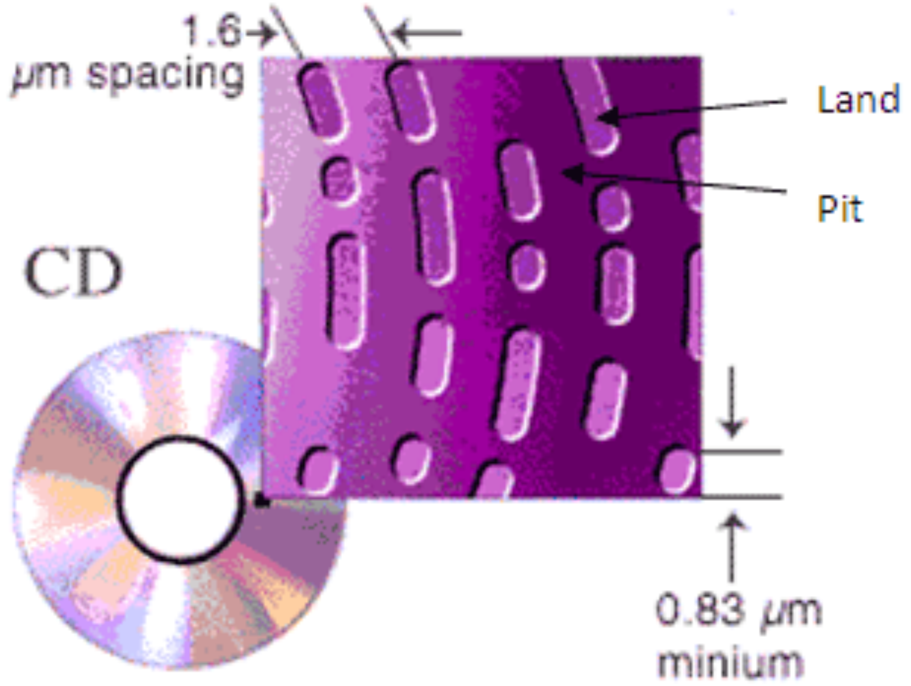
## In your machine

- We need hardware that can be one of two states

10001011011001010000111100000011010010

Relays
Punch cards
Tape
Optical disc (CD, DVD)
Transistors
...





https://computerhistory.org/

# Bits

## In your machine

- Can we get a pointer to a single bit?

```
1000101101100101000011110000001101001011
```

# Bits

## In your machine

- Can we get a pointer to a single bit? *NO!*

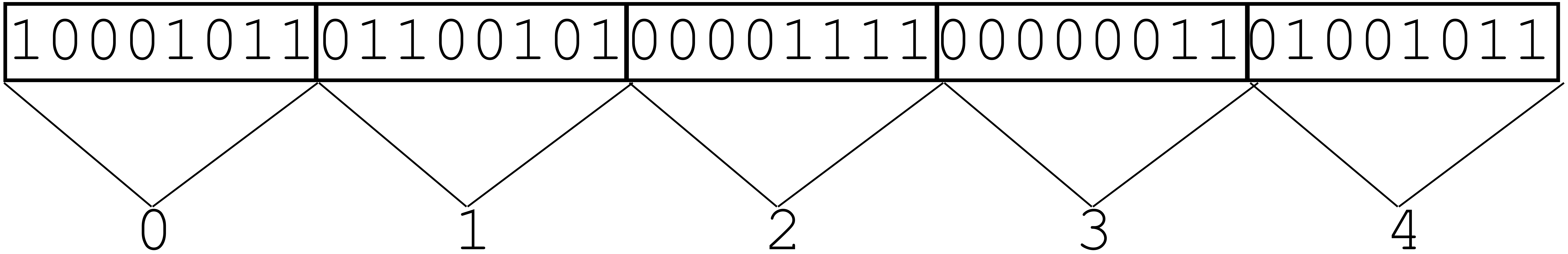
```
1000101101100101000011110000001101001011
```

- Too many addresses -- pointer sizes will be massive
- With some cleverness, you can retrieve a single bit (later!)

# Bits

## In your machine

- Can we get a pointer to a single bit? *NO!*



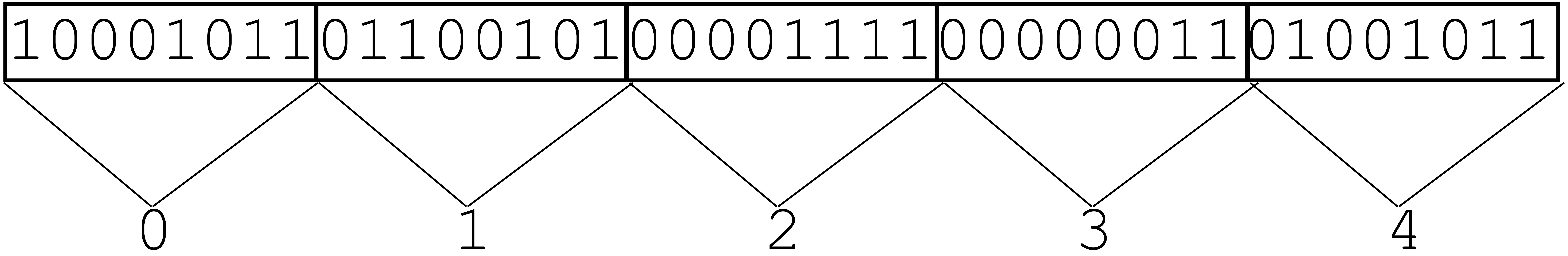
| | | | | |
|---|---|---|---|---|
| 10001011 | 01100101 | 00001111 | 00000011 | 01001011 |
| 0 | 1 | 2 | 3 | 4 |

- 8 bits is a *byte*.
- Memory is *byte-addressable*.

# Bits

## In your machine

- Why 8 bits?



10001011 01100101 00001111 00000011 01001011

0 1 2 3 4

- Big enough for an English character
- 8 is a power of 2 (later)

# Bits

## Know your powers of 2

| n | 2^n |
|---|---|
| 0 | 1 |
| 1 | 2 |
| 2 | 4 |
| 3 | 8 |
| 4 | 16 |
| 5 | 32 |
| 6 | 64 |
| 7 | 128 |

| n | 2^n |
|---|---|
| 8 | 256 |
| 9 | 512 |
| 10 | 1024 |
| 11 | 2048 |
| 12 | 4096 |
| 13 | 8192 |
| 14 | 16384 |
| 15 | 32768 |

| n | 2^n |
|---|---|
| 16 | 65,536 |
| 17 | 131,072 |
| 18 | 262,144 |
| 19 | 524,288 |
| 20 | 1,048,576 |
| 21 | 2,097,152 |
| 22 | 4,194,304 |
| 23 | 8,388,608 |

| n | 2^n |
|---|---|
| 24 | 16,777,216 |
| 25 | 33,554,432 |
| 26 | 67,108,864 |
| 27 | 134,217,728 |
| 28 | 268,435,456 |
| 29 | 536,870,912 |
| 30 | 1,073,741,824 |
| 31 | 2,147,483,648 |

| n | 2^n |
|---|---|
| 32 | 4,294,967,296 |
| 64 | 18,446,744,073,709,600,000 |
| | |
| | |
| | |
| | |
| | |
| | |

# Bits

## Know your powers of 2

| n | 2^n |
|---|---|
| 0 | 1 |
| 1 | 2 |
| 2 | 4 |
| 3 | 8 |
| 4 | 16 |
| 5 | 32 |
| 6 | 64 |
| 7 | 128 |

| n | 2^n |
|---|---|
| 8 | 256 |
| [illegible] | [illegible] |
| 10 | 1024 |
| 11 | 2048 |
| 12 | 4096 |
| 13 | 8192 |
| 14 | 16384 |
| 15 | 32768 |

| n | 2^n |
|---|---|
| 16 | 65,536 |
| 7 | [illegible] |
| 18 | 262,144 |
| 19 | 524,288 |
| 20 | 1,048,576 |
| 21 | 2,097,152 |
| 22 | 4,194,304 |
| 23 | 8,388,608 |

| n | 2^n |
|---|---|
| 24 | 16,777,216 |
| [illegible] | 33,554,432 |
| 26 | 67,108,864 |
| 27 | 134,217,728 |
| 28 | 268,435,456 |
| 29 | 536,870,912 |
| 30 | 1,073,741,824 |
| 31 | 2,147,483,648 |

| n | [illegible] |
|---|---|
| 32 | 4,294,967,296 |
| 64 | 18,446,744,073,709,600,000 |
| | |
| | |
| | |
| | |
| | |
| | |

`char`

`short`

`int, float`

`long, double,` pointers

# Bits

- Bits encode choices
- Bits have no inherent meanings; to read encoding, need to know the intended interpretation
- Bits can be stored in any devices that can be one of two states
- If the two states are 0 and 1, then binary number is an obvious encoding of integer numbers
- Modern machines has an address per byte (8 bits)

# Number Conversions

| n | 2^n |
|---|---|
| **0** | 1 |
| **1** | 2 |
| **2** | 4 |
| **3** | 8 |
| **4** | 16 |
| **5** | 32 |
| **6** | 64 |
| **7** | 128 |

**Decimal:** 1 9 9

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
|  |  |  |  |  |  |  |  |

# Number Conversions

| n | 2^n |
|---|---|
| **0** | 1 |
| **1** | 2 |
| **2** | 4 |
| **3** | 8 |
| **4** | 16 |
| **5** | 32 |
| **6** | 64 |
| **7** | 128 |

**Decimal:** 1 9 9

71

128 +

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| **1** | | | | | | | |

# Number Conversions

| n | 2^n |
|---|---|
| **0** | 1 |
| **1** | 2 |
| **2** | 4 |
| **3** | 8 |
| **4** | 16 |
| **5** | 32 |
| **6** | 64 |
| **7** | 128 |

**Decimal:** 1 9 9

7

128 + 64 +

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| **1** | **1** | | | | | | |

# Number Conversions

| n | 2^n |
|---|---|
| 0 | 1 |
| 1 | 2 |
| 2 | 4 |
| 3 | 8 |
| 4 | 16 |
| 5 | 32 |
| 6 | 64 |
| 7 | 128 |

**Decimal:** 1 9 9

7

128 + 64 +

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 |  |  |  |  |  |

# Number Conversions

| n | 2^n |
|---|---|
| **0** | 1 |
| **1** | 2 |
| **2** | 4 |
| **3** | 8 |
| **4** | 16 |
| **5** | 32 |
| **6** | 64 |
| **7** | 128 |

**Decimal:** 1 9 9

7

128 + 64 +

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| **1** | **1** | **0** | **0** | | | | |

# Number Conversions

| n | 2^n |
|---|---|
| **0** | 1 |
| **1** | 2 |
| **2** | 4 |
| **3** | 8 |
| **4** | 16 |
| **5** | 32 |
| **6** | 64 |
| **7** | 128 |

**Decimal:** 1 9 9

7

128 + 64 +

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| **1** | **1** | **0** | **0** | **0** |  |  |  |

# Number Conversions

| n | 2^n |
|---|---|
| 0 | 1 |
| 1 | 2 |
| 2 | 4 |
| 3 | 8 |
| 4 | 16 |
| 5 | 32 |
| 6 | 64 |
| 7 | 128 |

**Decimal:** 1 9 9

3

128 + 64 + 4 +

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 0 | 1 |  |  |

# Number Conversions

| n | 2^n |
|---|---|
| **0** | 1 |
| **1** | 2 |
| **2** | 4 |
| **3** | 8 |
| **4** | 16 |
| **5** | 32 |
| **6** | 64 |
| **7** | 128 |

**Decimal:** 1 9 9

1

128 + 64 + 4 + 2 +

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| **1** | **1** | **0** | **0** | **0** | **1** | **1** | |

# Number Conversions

| n | 2^n |
|---|---|
| 0 | 1 |
| 1 | 2 |
| 2 | 4 |
| 3 | 8 |
| 4 | 16 |
| 5 | 32 |
| 6 | 64 |
| 7 | 128 |

**Decimal:** 199

128 + 64 + 4 + 2 + 1

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 |

# Number Conversions

| n | 2^n |
|---|---|
| 0 | 1 |
| 1 | 2 |
| 2 | 4 |
| 3 | 8 |
| 4 | 16 |
| 5 | 32 |
| 6 | 64 |
| 7 | 128 |

**Decimal:** 200

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
|  |  |  |  |  |  |  |  |

# Number Conversions

| n | 2^n |
|---|---|
| **0** | 1 |
| **1** | 2 |
| **2** | 4 |
| **3** | 8 |
| **4** | 16 |
| **5** | 32 |
| **6** | 64 |
| **7** | 128 |

**Decimal:** 200

72

128 +

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| **1** | | | | | | | |

# Number Conversions

| n | 2^n |
|---|---|
| **0** | 1 |
| **1** | 2 |
| **2** | 4 |
| **3** | 8 |
| **4** | 16 |
| **5** | 32 |
| **6** | 64 |
| **7** | 128 |

**Decimal:** 200

8

128 + 64 +

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| **1** | **1** | | | | | | |

# Number Conversions

| n | 2^n |
|---|---|
| 0 | 1 |
| 1 | 2 |
| 2 | 4 |
| 3 | 8 |
| 4 | 16 |
| 5 | 32 |
| 6 | 64 |
| 7 | 128 |

**Decimal:** 200

0

128 + 64 + 8 +

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 1 |  |  |  |

# Number Conversions

| n | 2^n |
|---|---|
| 0 | 1 |
| 1 | 2 |
| 2 | 4 |
| 3 | 8 |
| 4 | 16 |
| 5 | 32 |
| 6 | 64 |
| 7 | 128 |

200

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |

# Number Conversions

| n | 2^n |
|---|---|
| **0** | 1 |
| **1** | 2 |
| **2** | 4 |
| **3** | 8 |
| **4** | 16 |
| **5** | 32 |
| **6** | 64 |
| **7** | 128 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| 199 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 |

+ 1

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| 200 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |

# Number Conversions

| n | 2^n |
|---|---|
| **0** | 1 |
| **1** | 2 |
| **2** | 4 |
| **3** | 8 |
| **4** | 16 |
| **5** | 32 |
| **6** | 64 |
| **7** | 128 |

**Decimal:** 20

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
|  |  |  |  |  |  |  |  |

# Number Conversions

| n | 2^n |
|---|---|
| 0 | 1 |
| 1 | 2 |
| 2 | 4 |
| 3 | 8 |
| 4 | 16 |
| 5 | 32 |
| 6 | 64 |
| 7 | 128 |

**Decimal:** 20

4

16 +

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | | | | |

# Number Conversions

| n | 2^n |
|---|---|
| 0 | 1 |
| 1 | 2 |
| 2 | 4 |
| 3 | 8 |
| 4 | 16 |
| 5 | 32 |
| 6 | 64 |
| 7 | 128 |

**Decimal:** 20

16 + 4

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |

# Number Conversions

| n | 2^n |
|---|---|
| **0** | 1 |
| **1** | 2 |
| **2** | 4 |
| **3** | 8 |
| **4** | 16 |
| **5** | 32 |
| **6** | 64 |
| **7** | 128 |

|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| 20 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |

|  | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| 200 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |

- x 10 can't convert nicely

# Number Conversions

| n | 2^n |
|---|---|
| 0 | 1 |
| 1 | 2 |
| 2 | 4 |
| 3 | 8 |
| 4 | 16 |
| 5 | 32 |
| 6 | 64 |
| 7 | 128 |

**Decimal:** 4 0

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
|  |  |  |  |  |  |  |  |

# Number Conversions

| n | 2^n |
|---|---|
| 0 | 1 |
| 1 | 2 |
| 2 | 4 |
| 3 | 8 |
| 4 | 16 |
| 5 | 32 |
| 6 | 64 |
| 7 | 128 |

**Decimal:** 40

8

32 +

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 |  |  |  |  |  |

# Number Conversions

| n | 2^n |
|---|---|
| **0** | 1 |
| **1** | 2 |
| **2** | 4 |
| **3** | 8 |
| **4** | 16 |
| **5** | 32 |
| **6** | 64 |
| **7** | 128 |

**Decimal:** 40

32 + 8

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| **0** | **0** | **1** | **0** | **1** | **0** | **0** | **0** |

# Number Conversions

| n | 2^n |
|---|---|
| **0** | 1 |
| **1** | 2 |
| **2** | 4 |
| **3** | 8 |
| **4** | 16 |
| **5** | 32 |
| **6** | 64 |
| **7** | 128 |

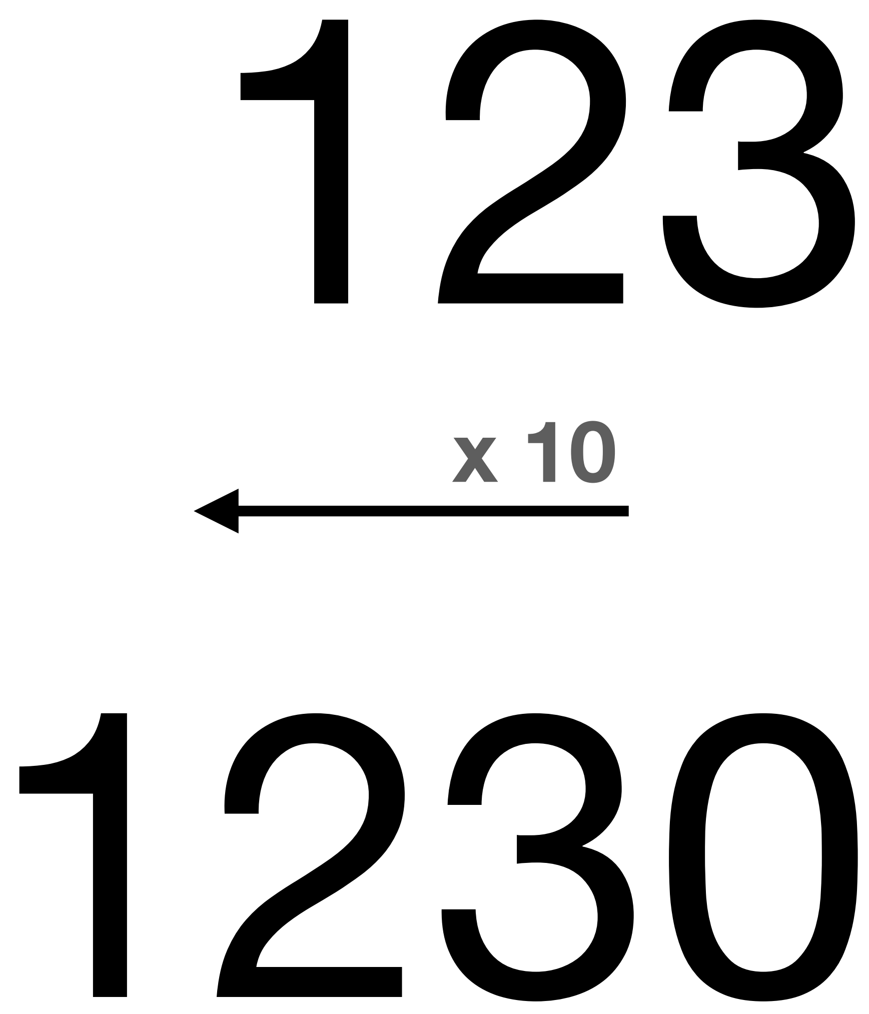
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| 20 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| 40 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 |

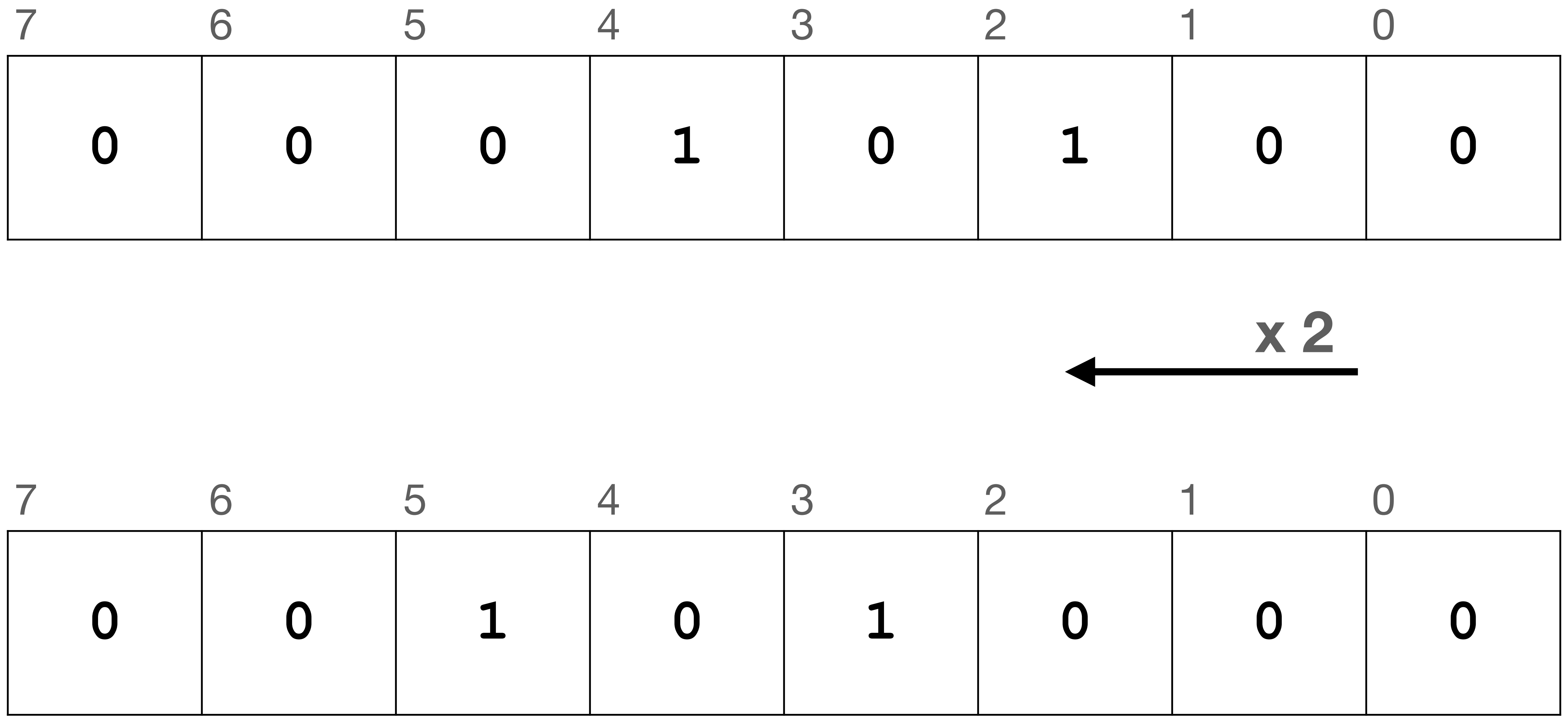
- x 2 is shifting everything to the left!

# Number Conversions

| n | 2^n |
|---|---|
| 0 | 1 |
| 1 | 2 |
| 2 | 4 |
| 3 | 8 |
| 4 | 16 |
| 5 | 32 |
| 6 | 64 |
| 7 | 128 |



**Decimal**



| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |

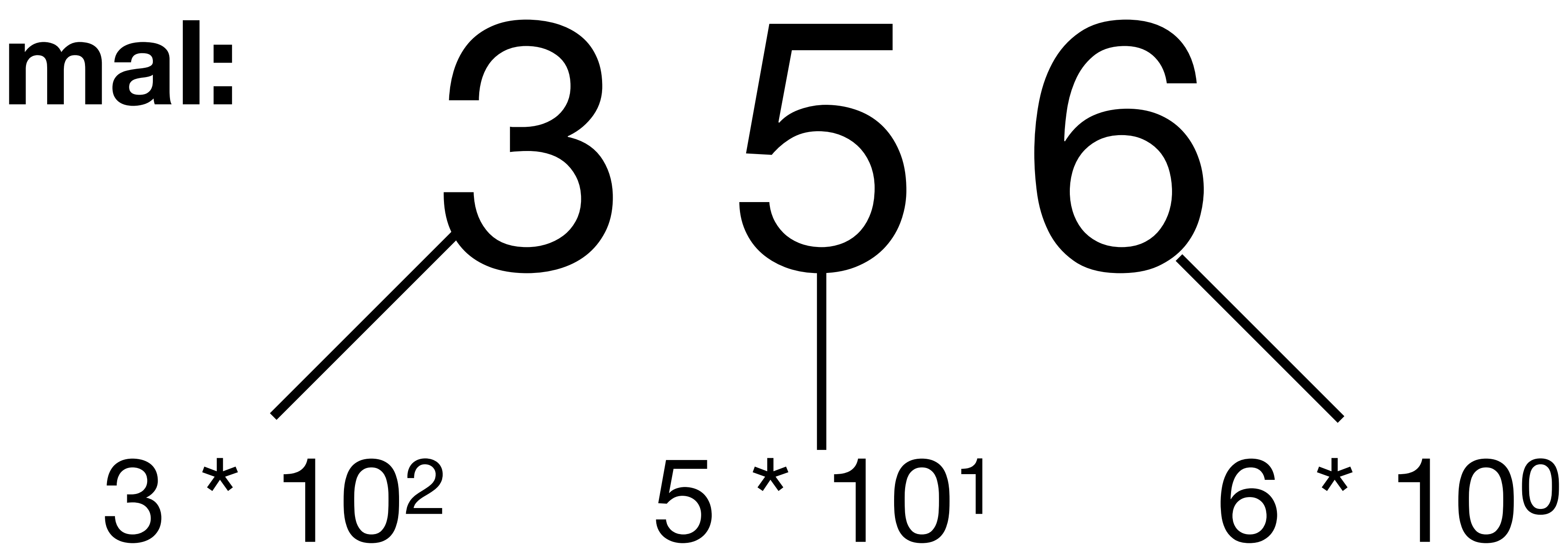
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 |

**Binary**

# Hexadecimal Numbers

**Numbers with even wackier math**

**Decimal:**



3 5 6

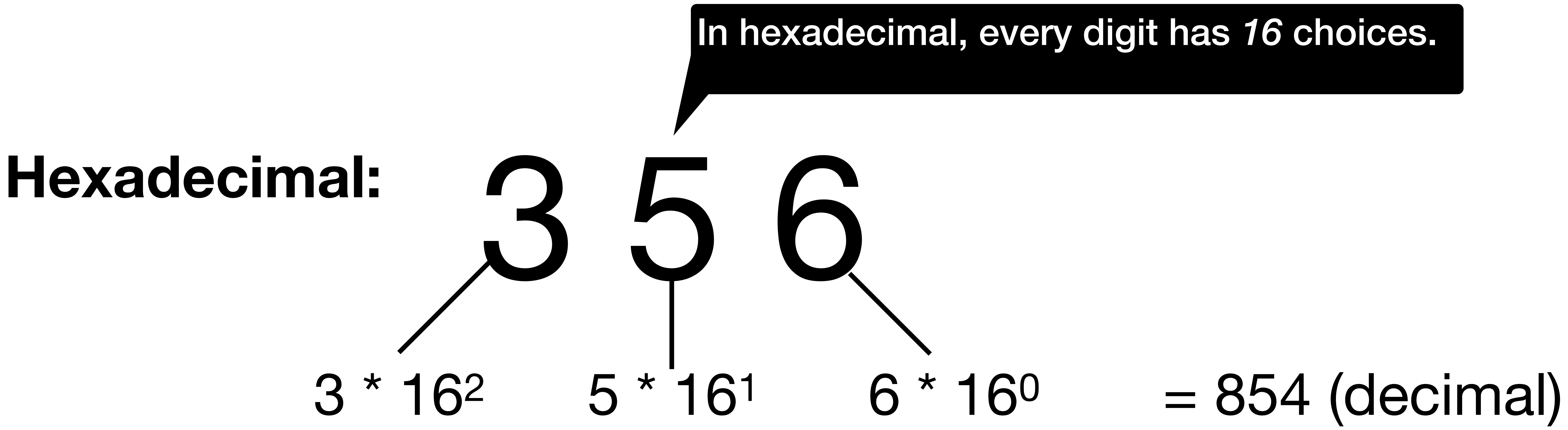
$3 * 10^2$

$5 * 10^1$

$6 * 10^0$

# Hexadecimal Numbers

## Numbers with even wackier math



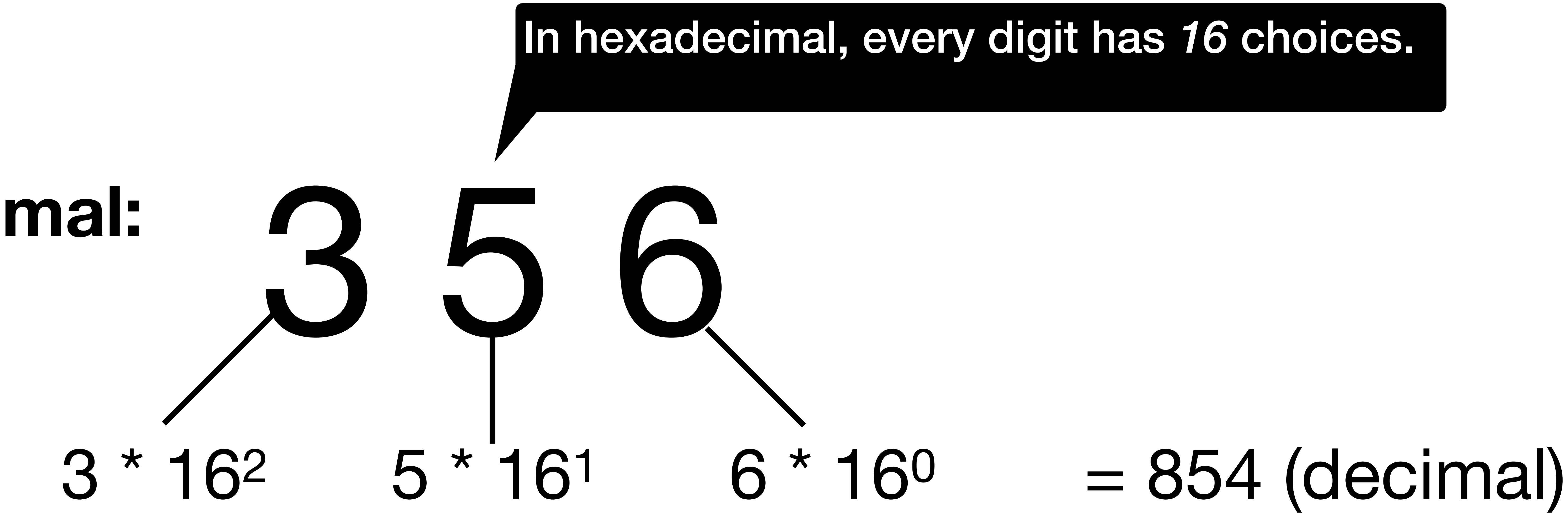
In hexadecimal, every digit has *16* choices.

**Hexadecimal:** 3 5 6

$3 * 16^2$ $5 * 16^1$ $6 * 16^0$ = 854 (decimal)

# Hexadecimal Numbers

## Numbers with even wackier math



In hexadecimal, every digit has *16* choices.

**Hexadecimal:** 3 5 6

$3 * 16^2$ $5 * 16^1$ $6 * 16^0$ = 854 (decimal)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
|  |  |  |  |  |  |  |  |  |  | 10 | 11 | 12 | 13 | 14 | 15 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |

# Hexadecimal Numbers

## Numbers with even wackier math

- Why 16?
- Because 16 is $2^4$! So every digit needs exactly 4 bits -- this will turn out to be very convenient
- Usually, we prefix hexadecimal numbers with `0x`

| | | | | | | | | | | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |

# Hexadecimal Numbers

## Numbers with even wackier math

| n | 16^n |
|---|---|
| 0 | 1 |
| 1 | 16 |
| 2 | 256 |
| 3 | 4096 |

0x A B

|  |  |  |  |  |  |  |  |  |  | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |

# Hexadecimal Numbers

## Numbers with even wackier math

| n | 16^n |
|---|---|
| 0 | 1 |
| 1 | 16 |
| 2 | 256 |
| 3 | 4096 |

0x A B

10 * 16 + 11 * 1 = 171

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |

| | | | | | | | | | | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |

# Hexadecimal Numbers

## Numbers with even wackier math

| n | 2^n |
|---|---|
| 0 | 1 |
| 1 | 2 |
| 2 | 4 |
| 3 | 8 |
| 4 | 16 |
| 5 | 32 |
| 6 | 64 |
| 7 | 128 |

0x A B

10 * 16 + 11 * 1 = 171

| 3 | 2 | 1 | 0 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |

| | | | | | | | | | | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |

# Hexadecimal Numbers

## Numbers with even wackier math

| n | 2^n |
|---|---|
| 0 | 1 |
| 1 | 2 |
| 2 | 4 |
| 3 | 8 |
| 4 | 16 |
| 5 | 32 |
| 6 | 64 |
| 7 | 128 |

0x A B

10 * 16 + 11 * 1 = 171

| 3 | 2 | 1 | 0 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |

$\underbrace{1\ 0\ 1\ 0}_{10}$

|  |  |  |  |  |  |  |  |  |  | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |

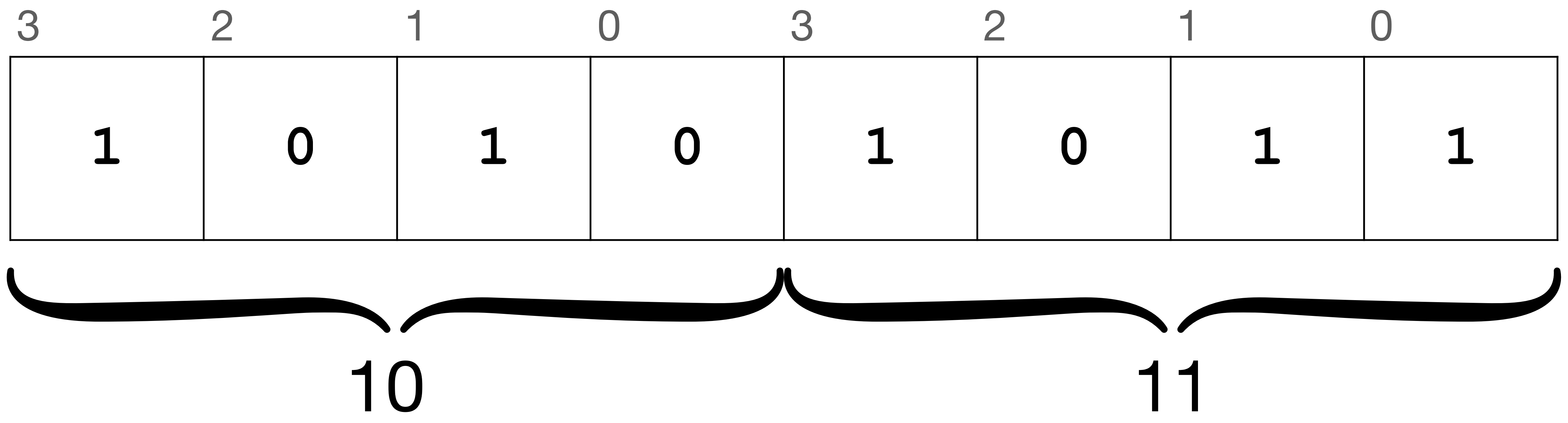
# Hexadecimal Numbers

## Numbers with even wackier math

| n | 2^n |
|---|---|
| **0** | 1 |
| **1** | 2 |
| **2** | 4 |
| **3** | 8 |
| **4** | 16 |
| **5** | 32 |
| **6** | 64 |
| **7** | 128 |

0x A B

10 * 16 + 11 * 1 = 171



| 3 | 2 | 1 | 0 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 |

|   |   |   |   |   |   |   |   |   |   | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |

# Hexadecimal Numbers

## Numbers with even wackier math

| n | 2^n |
|---|---|
| 0 | 1 |
| 1 | 2 |
| 2 | 4 |
| 3 | 8 |
| 4 | 16 |
| 5 | 32 |
| 6 | 64 |
| 7 | 128 |

0x A B

10 * 16 + 11 * 1 = 171

$$\underbrace{\begin{array}{cccc}3&2&1&0\\ 1&0&1&0\end{array}}_{\text{0xA}}\underbrace{\begin{array}{cccc}3&2&1&0\\ 1&0&1&1\end{array}}_{\text{0xB}}$$

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | 10 | 11 | 12 | 13 | 14 | 15 |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |

# Hexadecimal Numbers

## Numbers with even wackier math

- Every hex digit corresponds to 4 binary digits
- We can convert 1 hex digit/4 binary digits at a time!
    - Can't do this with decimal digits

|  |  |  |  |  |  |  |  |  |  | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |

# Hexadecimal Numbers

## Numbers with even wackier math

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |

|  |  |  |  |  |  |  |  |  |  | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |

# Hexadecimal Numbers

## Numbers with even wackier math

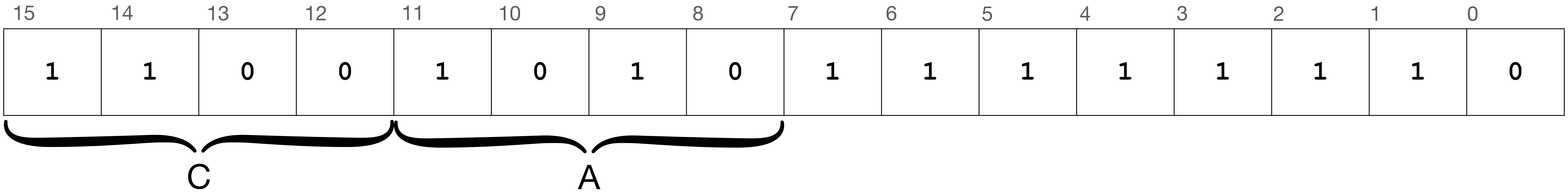
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |

C

| | | | | | | | | | | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |

# Hexadecimal Numbers

## Numbers with even wackier math



| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |

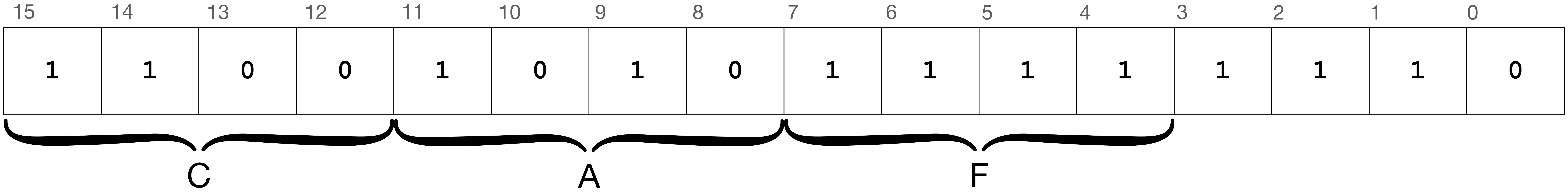
C

A

|  |  |  |  |  |  |  |  |  |  | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |

# Hexadecimal Numbers

## Numbers with even wackier math

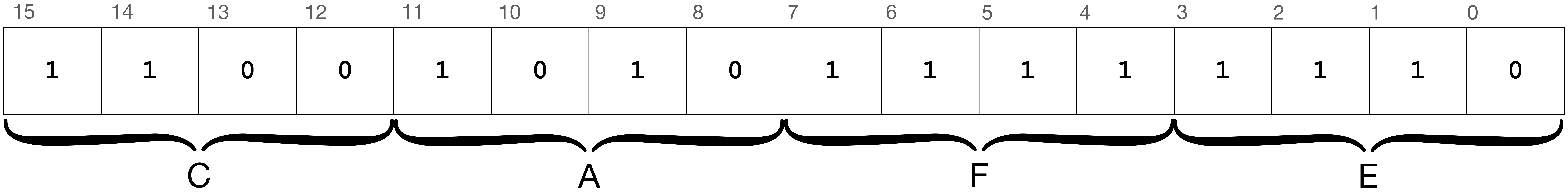


| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| C |  |  |  | A |  |  |  | F |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |

# Hexadecimal Numbers

## Numbers with even wackier math

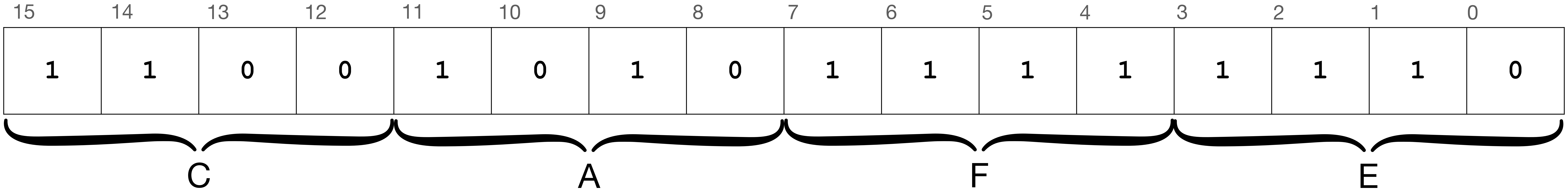


| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| C | | | | A | | | | F | | | | E | | | |

|  |  |  |  |  |  |  |  |  |  | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |

# Hexadecimal Numbers

## Numbers with even wackier math



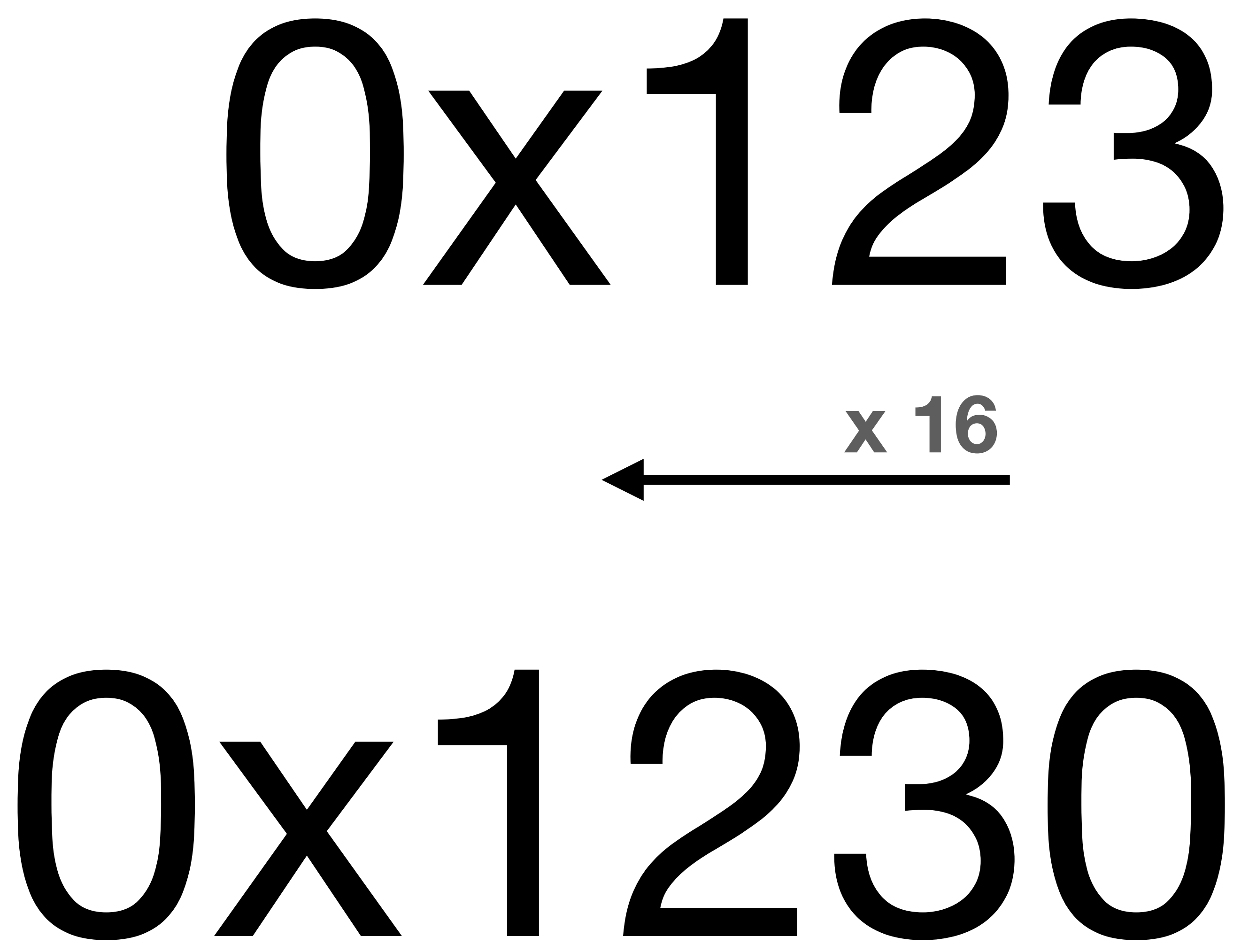
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |

C A F E

```
>>> hex(0b1100101011111110)
'0xcafe'
```

|  |  |  |  |  |  |  |  |  |  | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |

# Hexadecimal Numbers

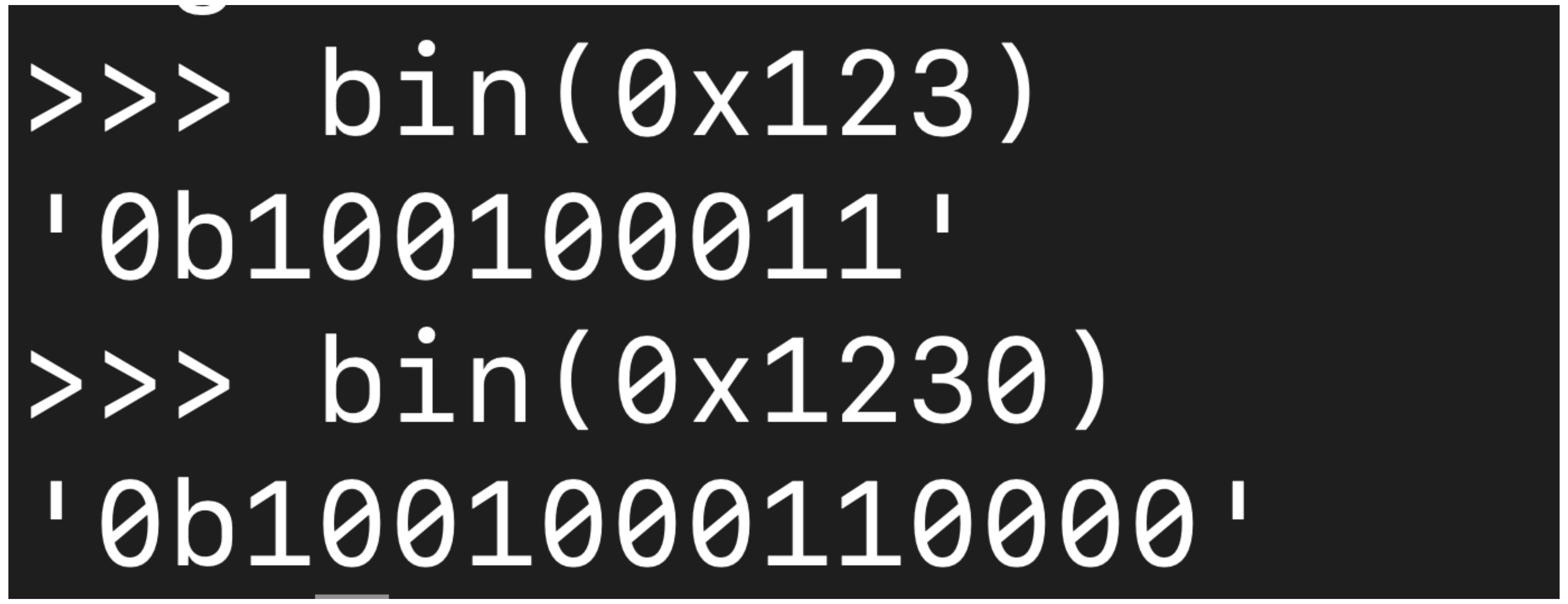


0x123

x 16

0x1230

Hexadecimal

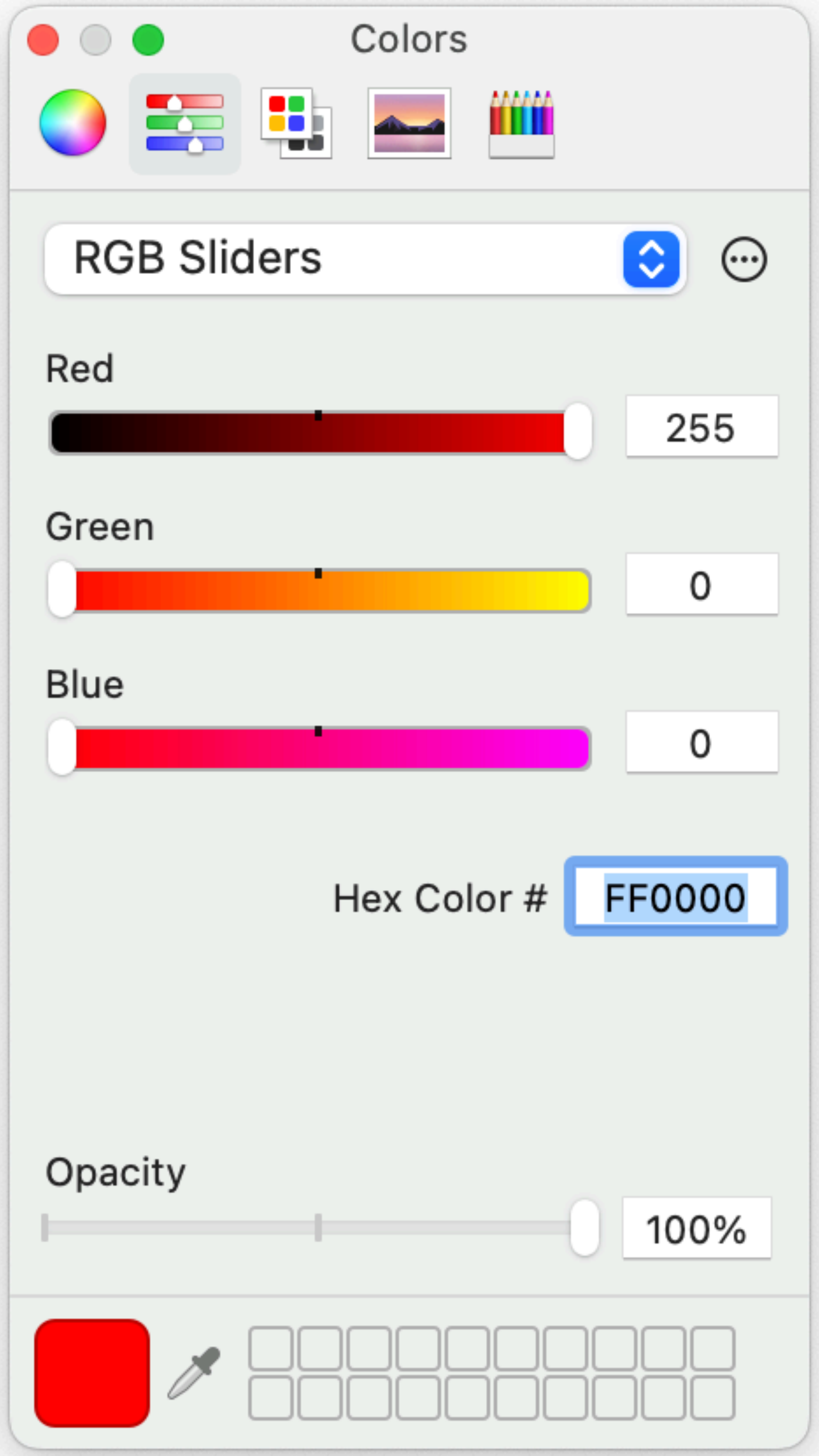


```
>>> bin(0x123)
'0b100100011'
>>> bin(0x1230)
'0b1001000110000'
```

Binary

# Hexadecimal Numbers

## Hex numbers are everywhere


Colors

RGB Sliders

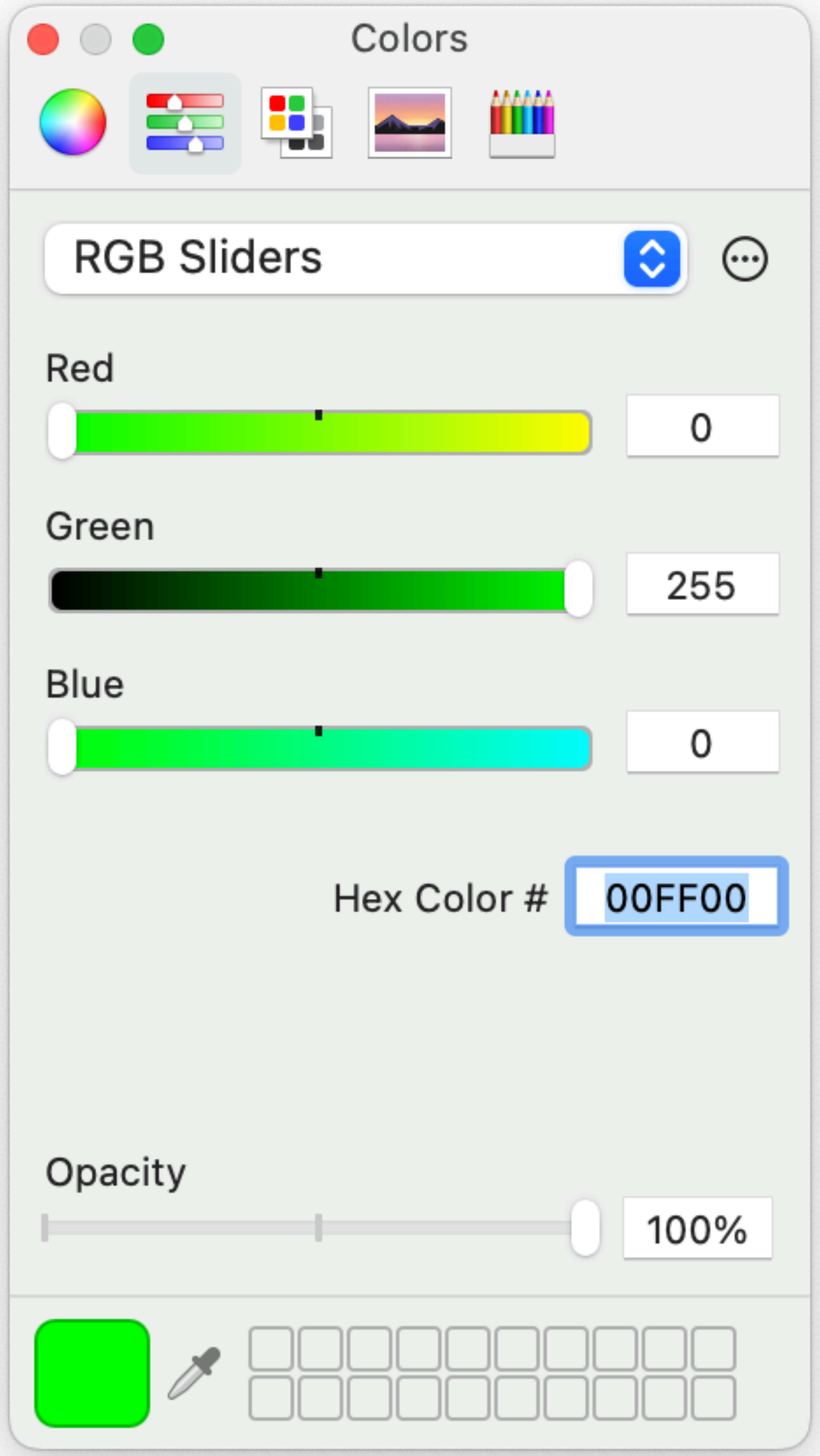
Red 255

Green 0

Blue 0

Hex Color # FF0000

Opacity 100%


Colors

RGB Sliders

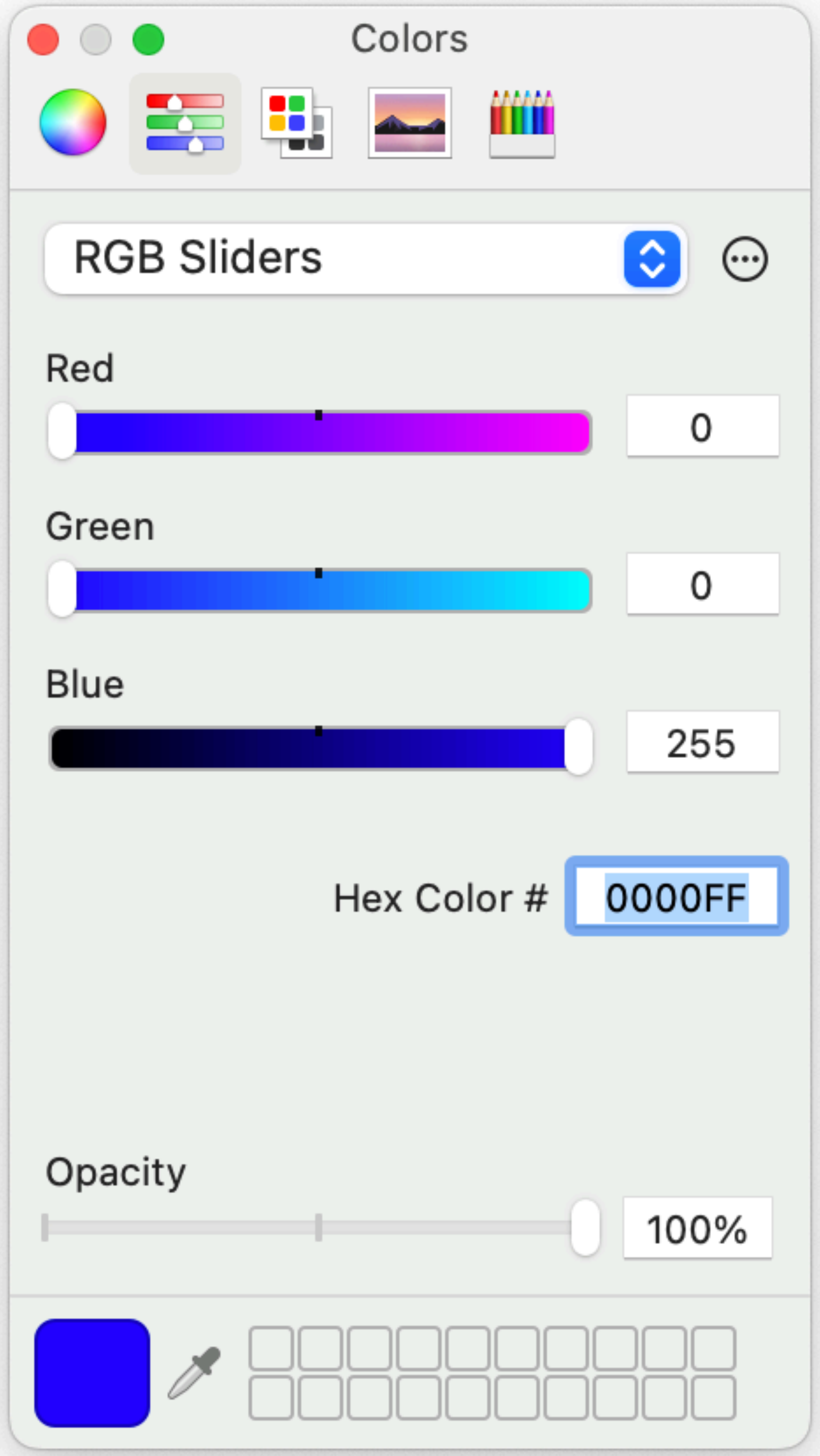
Red 0

Green 255

Blue 0

Hex Color # 00FF00

Opacity 100%


Colors

RGB Sliders

Red 0

Green 0

Blue 255

Hex Color # 0000FF

Opacity 100%