



Encoding Schemes



Objectives

- Explore how computers store data in different numeration systems including binary, decimal, hex, and ASCII.
- Examine how to convert between different numeration systems.
- Explore the properties of the XOR operation.



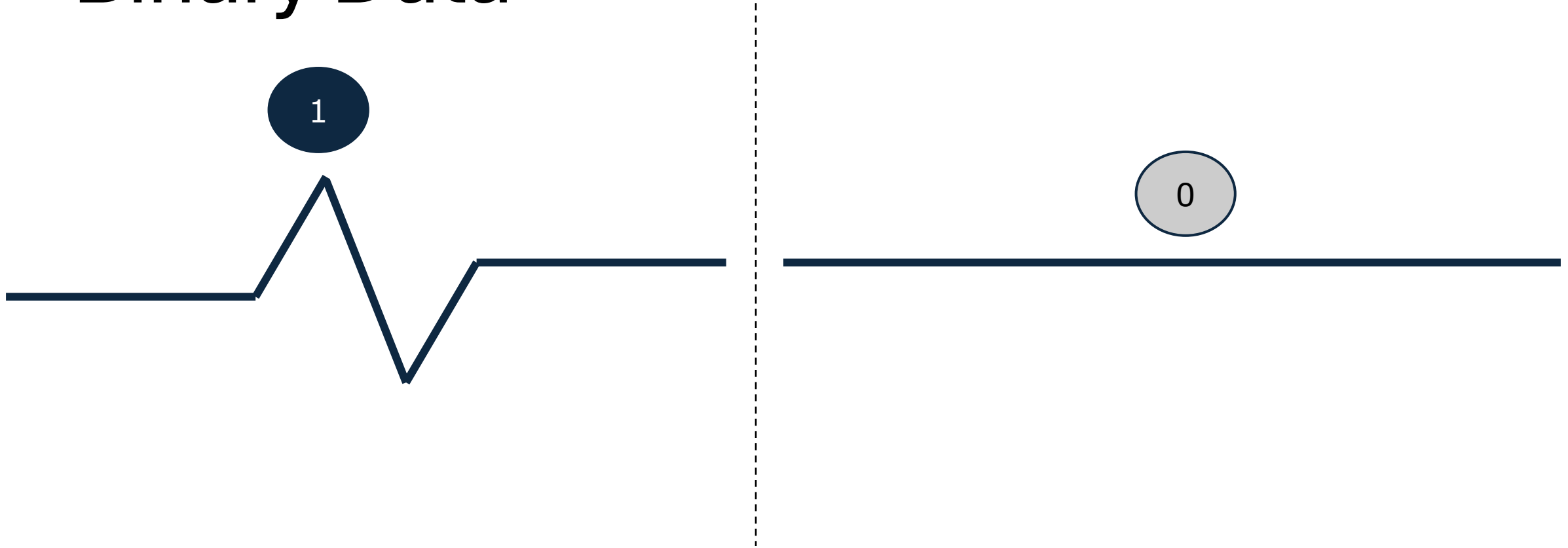


References

- [CyberChef](#)
- Binary to Decimal E-mates [[Link](#)]
- Hex Numbers E-Mates [[Link](#)]

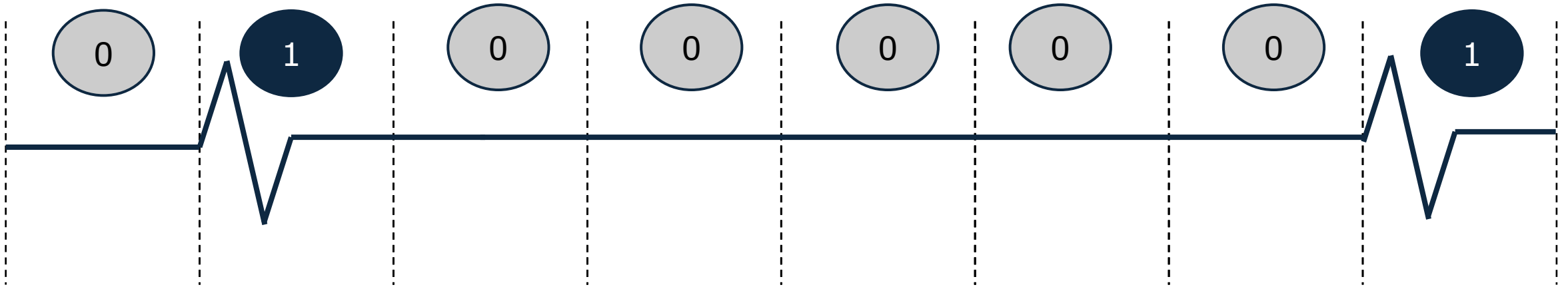


Binary Data



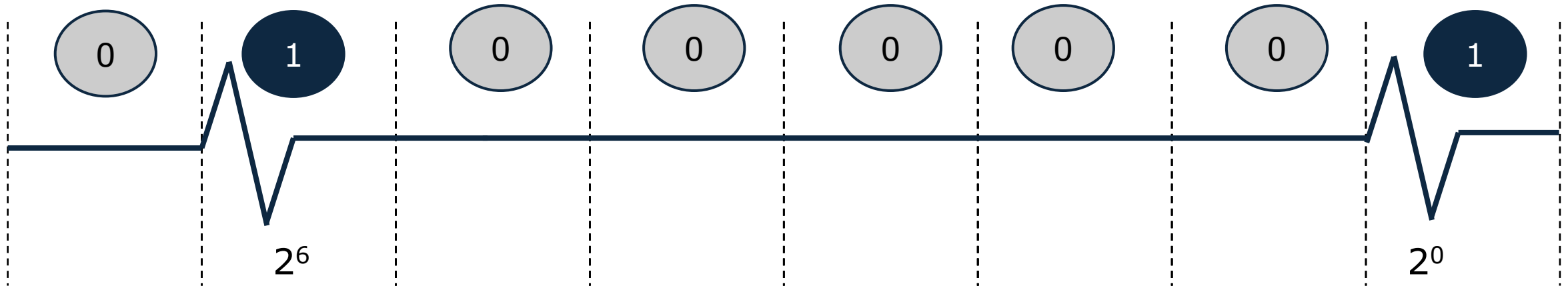
Binary refers to a number system where there are only two possible outcomes (1 or 0). Computers use transistors and capacitors to store electrical charges. These charges represent either a 1 or a 0. A bit represents a single outcome.

Bits to Bytes



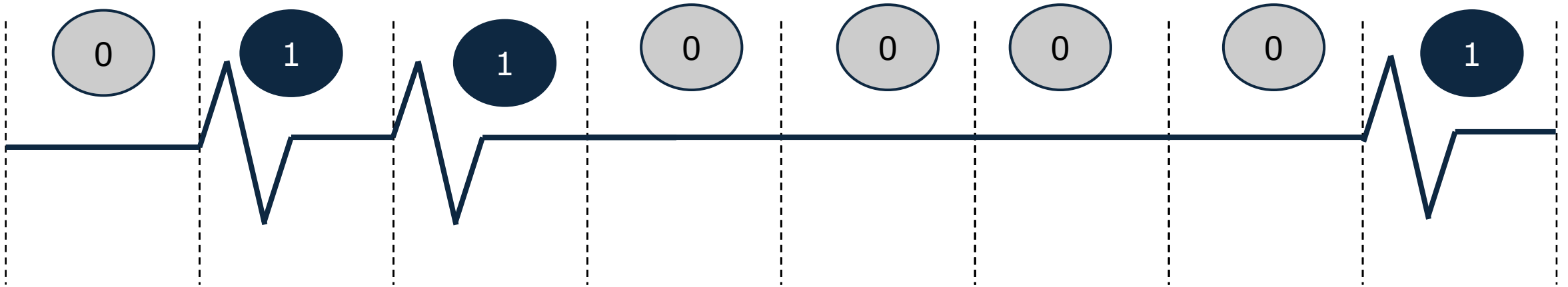
We called groupings of 8 bits a **byte**. Since each bit can have two possible outcomes, **a byte can store $2^8 = 256$ possible values.**

Bytes to Decimal



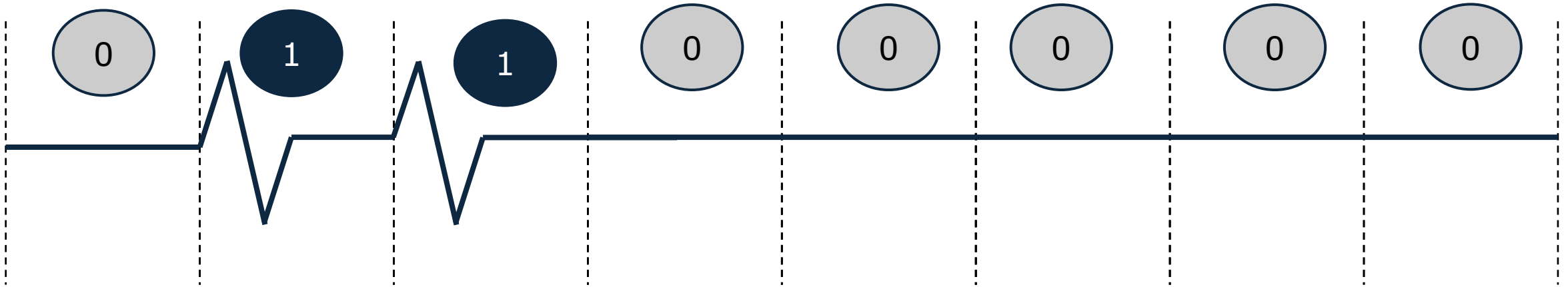
To convert bytes to decimal (base 10 numbering), we sum each enabled bit raised to the power of its position. So, $0\mathbf{1}00000\mathbf{1} = 2^6 + 2^0 = 64 + 1 = 65$

Practice #1



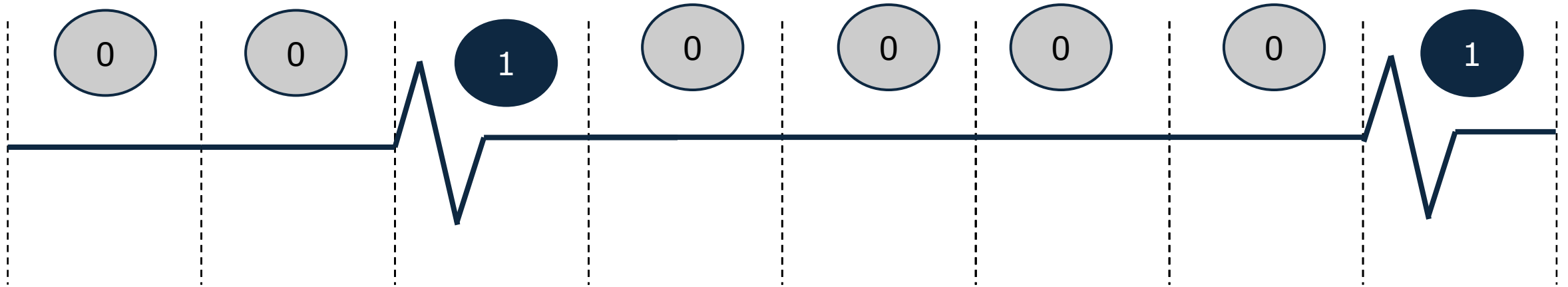
What is the decimal value? = $2^6 + 2^5 + 2^0 = 97$

Practice #2



What is the decimal value? _____

Practice #3



What is the decimal value? _____

Decimal to Hex

- Normally, we count in base-10 (decimal) numbering system. It uses the symbols 0-9
- However, with computers we often represent numbers in base-16 (hexadecimal.) It uses the symbols, 0-9, a-f.
- We also use the notation 0x??, to indicate hexadecimal.

Decimal	Hex
1	0x01
2	0x02
3	0x03
4	0x04
5	0x05
6	0x06
7	0x07
8	0x08
9	0x09
10	0x0a
11	0x0b
12	0x0c
13	0x0d
14	0x0e
15	0x0f
16	0x10



Hexadecimal

- We use hexadecimal often to compactly represent a byte.
- Since the largest byte (255 = 0xff), a single byte can be represented by two symbols.

Decimal	Hex
10	0x0a
20	0x14
30	0x1e
40	0x28
50	0x32
60	0x3c
70	0x46
80	0x50
90	0x5a
100	0x64
110	0x6e
120	0x78
130	0x82
140	0x8c
150	0x96
255	0xff



Decimal to ASCII

```
>>> import string

>>> print(string.printable)
0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ!"#$%&'()*+,-./:;<=>?@[\\]^_`{|}~

>>> print(len(string.printable))
100

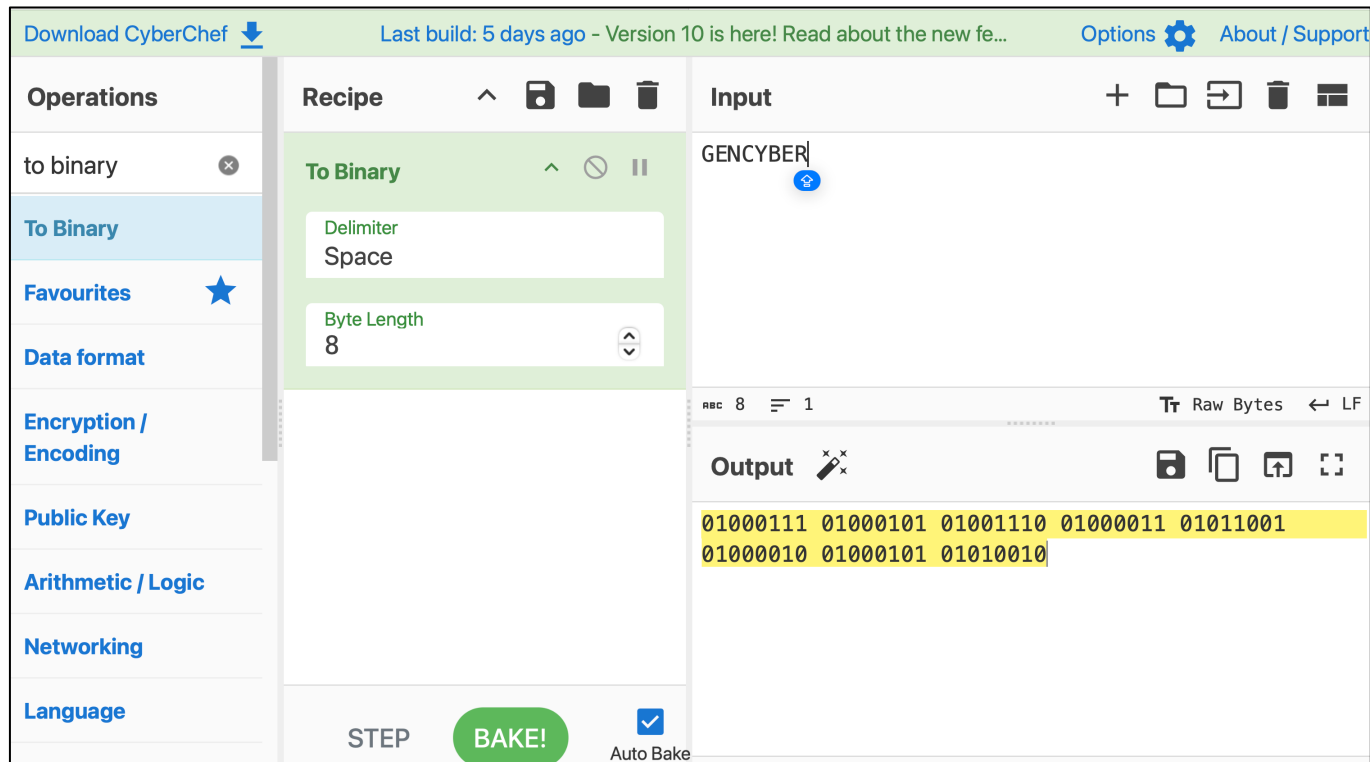
>>> chr(65)
'A'
```

Since computers can't store letters, they just create tables to represent letters, where each letter has a specific decimal value. For example, the number 65 represents the letter A and the number 66 represents B.

A	B	C	...	a
65	66	67	...	97

CyberChef

- [CyberChef](#) is a tool that can assist us in converting between different numeration systems.



XOR Boolean Operation

A	B	Result
0	0	0
0	1	1
1	0	1
1	1	0

$A = B \rightarrow 0$
 $A \neq B \rightarrow 1$

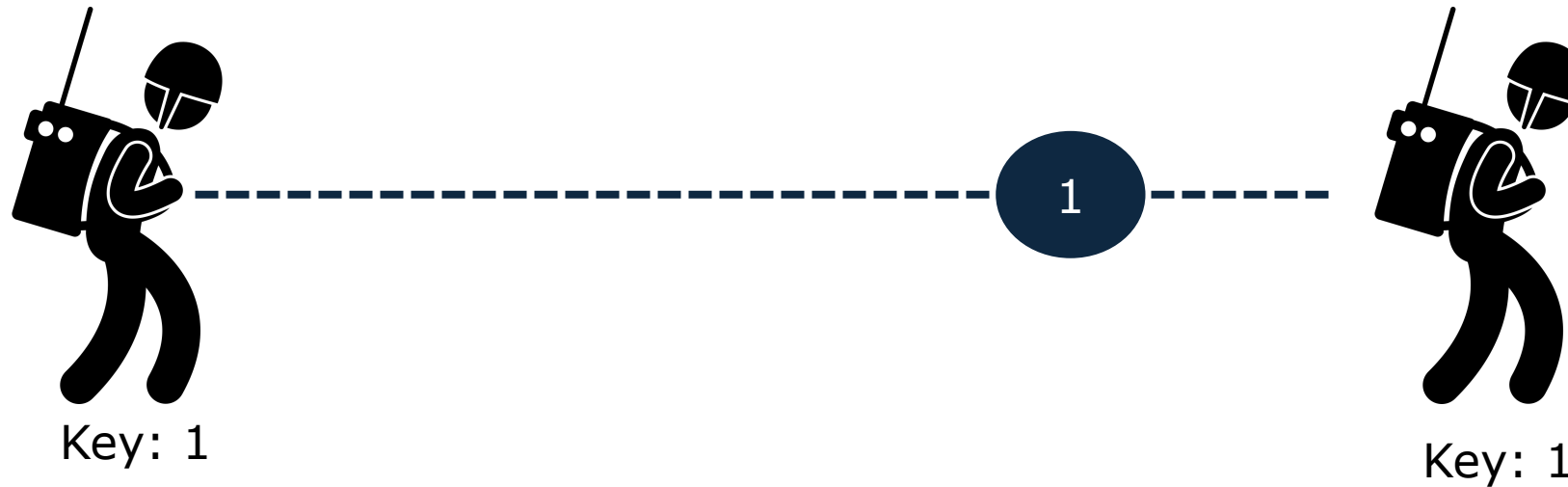
XOR is a Boolean logic operation that we use in cryptography. Its pretty simple, we compare two bits, if they are the same the result is a 0. If they are different, the result is a 1.

XOR Symmetric Key

$$\begin{aligned} A = B &\rightarrow 0 \\ A \neq B &\rightarrow 1 \end{aligned}$$

Plaintext	Key	Ciphertext
0	0	0
0	1	1
1	0	1
1	1	0

Imagine a soldier transmitted a ciphertext message of "1" that was XOR encrypted with the key "1". What was the original plaintext message?



XOR Symmetric Key Encryption

$$\begin{aligned} A = B &\rightarrow 0 \\ A \neq B &\rightarrow 1 \end{aligned}$$

Plaintext	Key	Ciphertext
0	0	0
0	1	1
1	0	1
1	1	0

Imagine a soldier transmitted a ciphertext message of "1" that was XOR encrypted with the key "1". What was the original plaintext message?



Key: 1

1



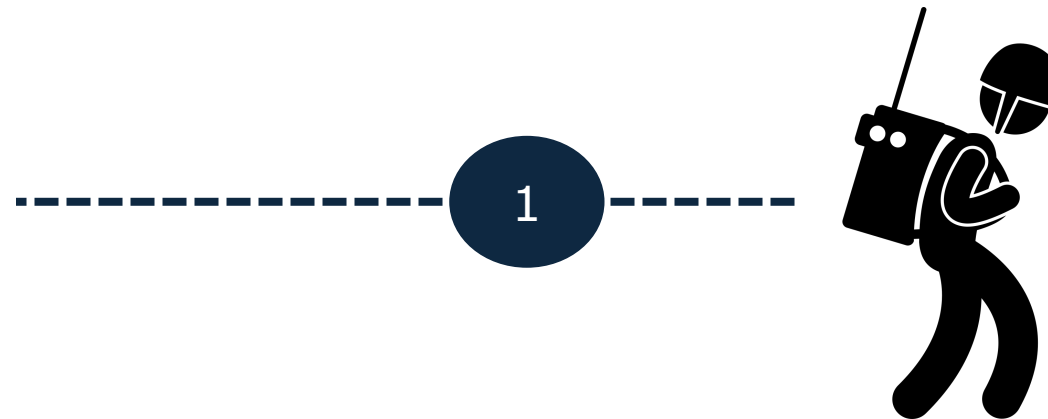
Key: 1

XOR Symmetric Key

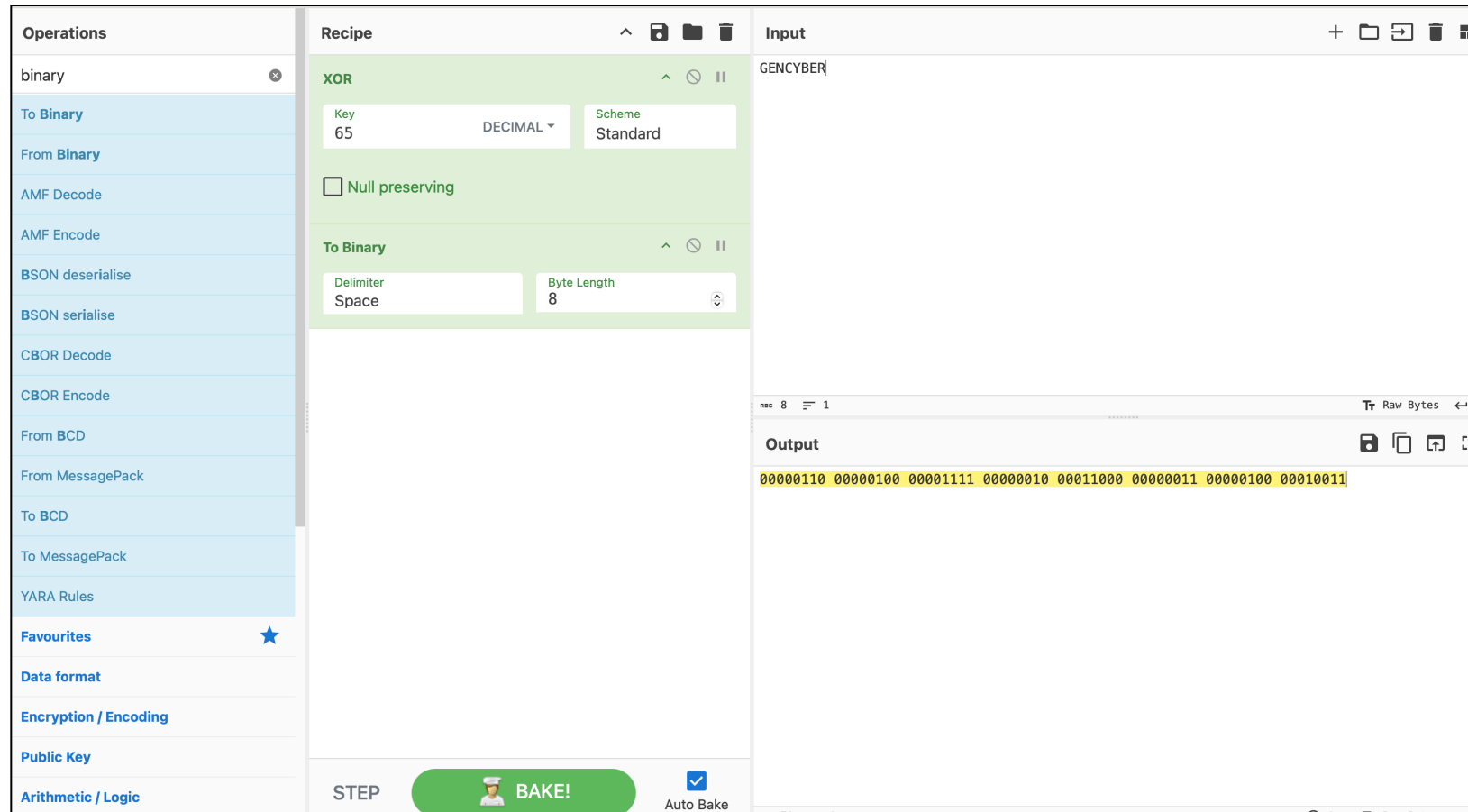
$$\begin{aligned} A = B &\rightarrow 0 \\ A \neq B &\rightarrow 1 \end{aligned}$$

Plaintext	Key	Ciphertext
0	0	0
0	1	1
1	0	1
1	1	0

Notice how the soldier **CAN ONLY DECRYPT** the message if she has the key? Without knowing the key, the plaintext could be a 1 or 0.



Practice in CyberChef



The screenshot displays the CyberChef web application interface, which is divided into three main sections: Operations, Recipe, and Input/Output.

- Operations:** A sidebar on the left lists various operations. The 'To Binary' operation is currently selected and highlighted in blue.
- Recipe:** The central area shows the active recipe. It includes:
 - XOR:** A section with a 'Key' of 65 (DECIMAL) and a 'Scheme' of Standard.
 - Null preserving:** A checkbox that is currently unchecked.
 - To Binary:** A section with a 'Delimiter' of Space and a 'Byte Length' of 8.
- Input:** The top right section contains the input text 'GENCYBER'.
- Output:** The bottom right section displays the resulting binary output: 00000110 00000100 00001111 00000010 00011000 00000011 00000100 00010011.

At the bottom of the interface, there is a 'STEP' button, a 'BAKE!' button with a chef icon, and an 'Auto Bake' checkbox which is checked.