C3
Data Representation

Outline

1. Bits and Bytes
2. Representation of Character Sets
   a. ASCII
   b. Unicode
3. Representation of Data Types
   - Numeric, Boolean, Character, String
4. Representation of Numbers
5. Representation of Images, Sound and Video (not included in Syllabus)

1. Bits and Bytes

- Values are stored inside a computer as a series of 0s and 1s

$$\underbrace{0 \qquad\qquad 1}_{\text{Binary digit or bit}}$$

1. Bits and Bytes

- How big is a Byte?
- Depend on the computer (hardware) architecture
  - Early computers by CDC Corp : 6-bit byte
  - Early computers by BB&N : 10-bit byte
  - Today's computers : 8-bit byte

$$\underbrace{01000001}_{\text{8 bits = 1 byte}} \rightarrow \text{'A'}$$

1. Bits and Bytes

- Byte Size determines the maximum number of possible values that can be stored.
  - 1 bit : 0 or 1 → 2 = 2^1 values
  - 2 bits : 00, 01, 10 or 11 → 4 = 2^2 values
  - 3 bits : 000, 001, 010, 011, 100, 101, 110 or 111 → 8 = 2^3 values
  - 4 bits : … → 2^4 = 16 values
  - …
  - 8 bits : … → 2^8 = 256 values
  - 16 bits : … → 2^{16} = 65,536 values
  - 32 bits : … → 2^{32} = 4,294,967,296 values

1. Bits and Bytes

- The bits themselves have no intrinsic meaning, the interpretation of the values is determined by the way the hardware and software use the bits

2. Representation of Character Sets

- Using a byte for a character, we have 256 codes, enough to represent each of the characters on a standard keyboard

01000001

2. Representation of Character Sets

Computer 1
01000001→ ‘A’

Computer 2
11001001→ ‘A’

- These two computers cannot communicate with each other because they uses different Character Representation Sets.

2. Representation of Character Sets
a. ASCII

- 1960s – agreed a standard set of codes
- American Standard Code for Information Interchange (ASCII)
- "Ask-key"

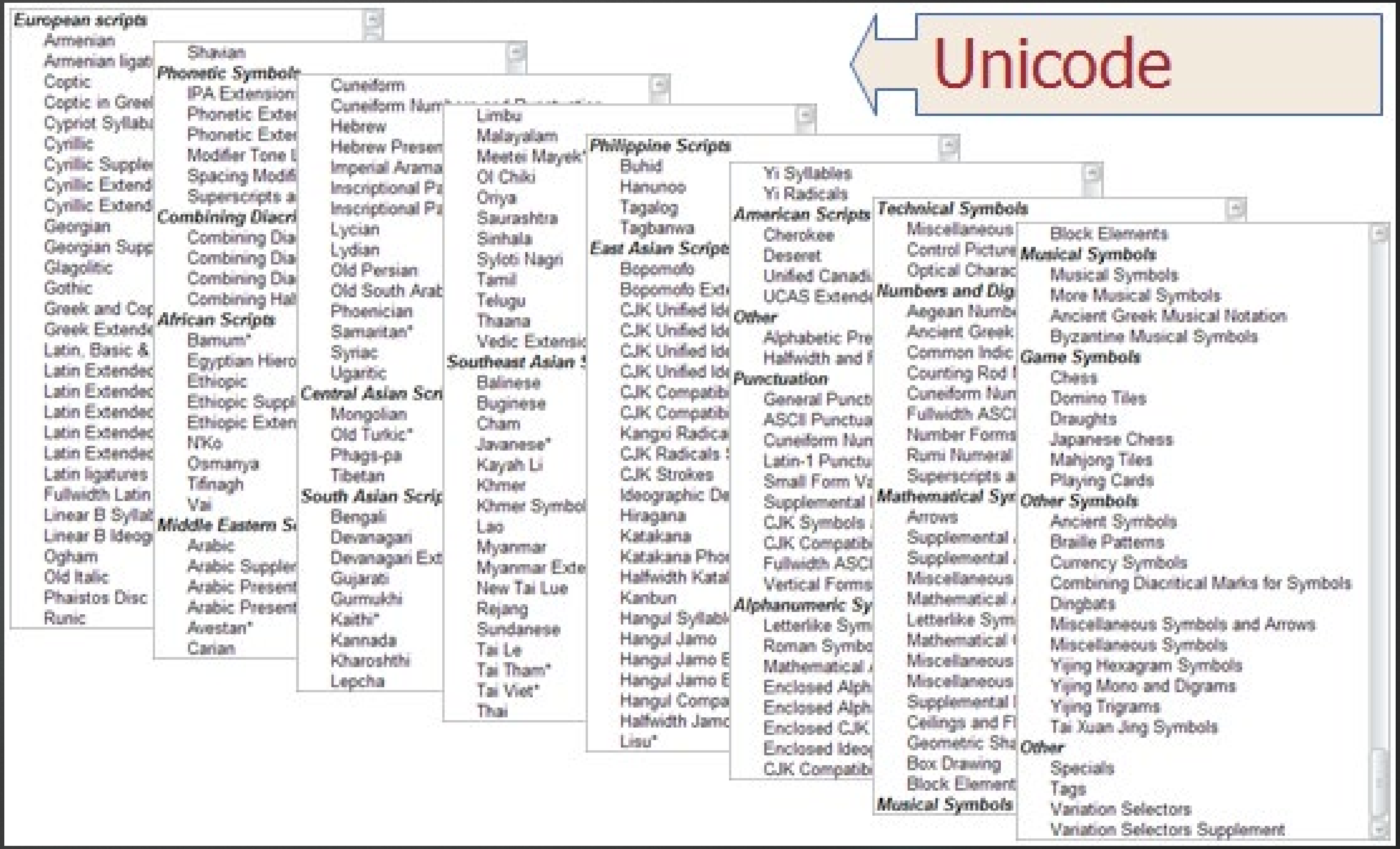
2. Representation of Character Sets
a. ASCII

- Uses 7 bits to represent each character and the 8th bit as a means of checking the rest of the 7 bits.

  (we will learn this under the topic '*Parity Check*')

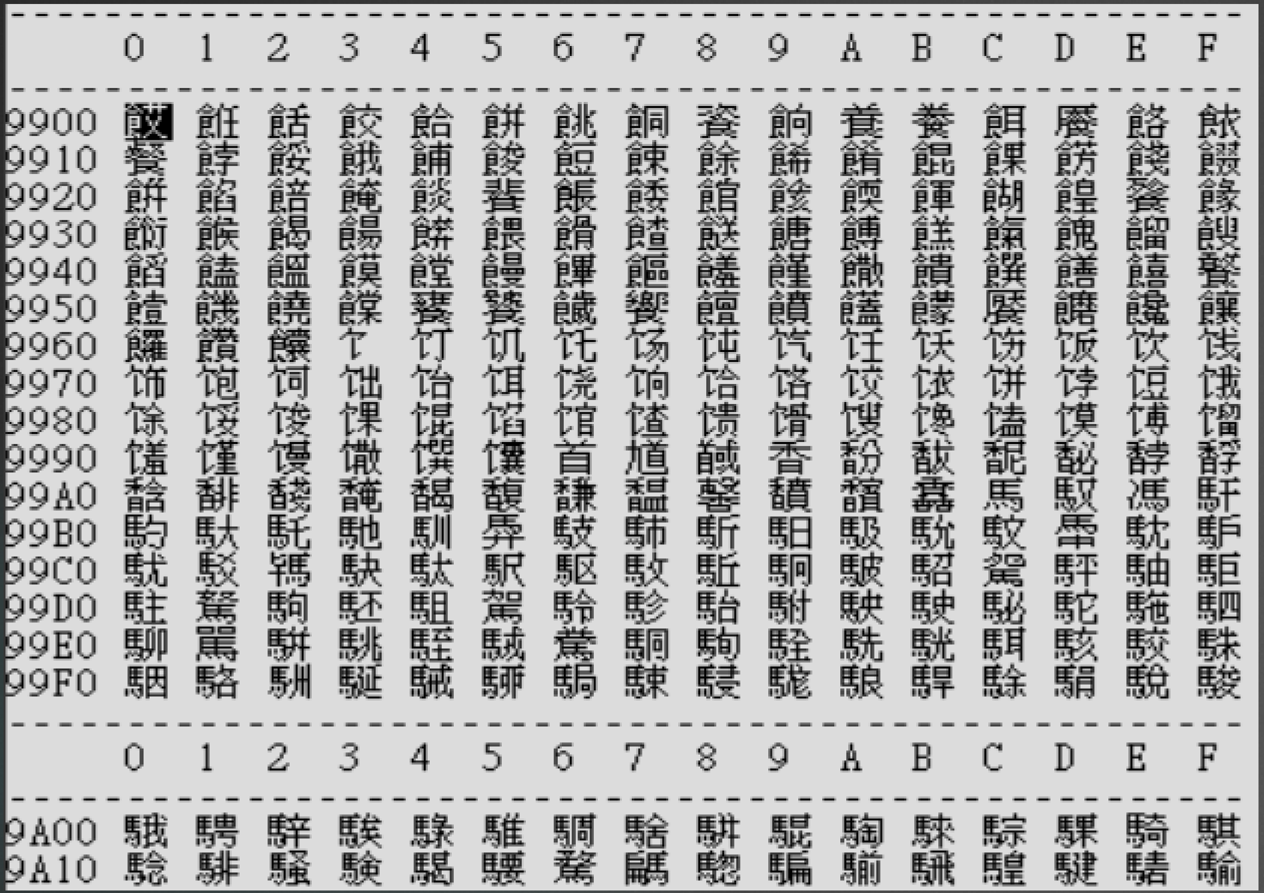
- 128 characters can be represented in the standard ASCII character set.

2. Representation of Character Sets
b. Unicode

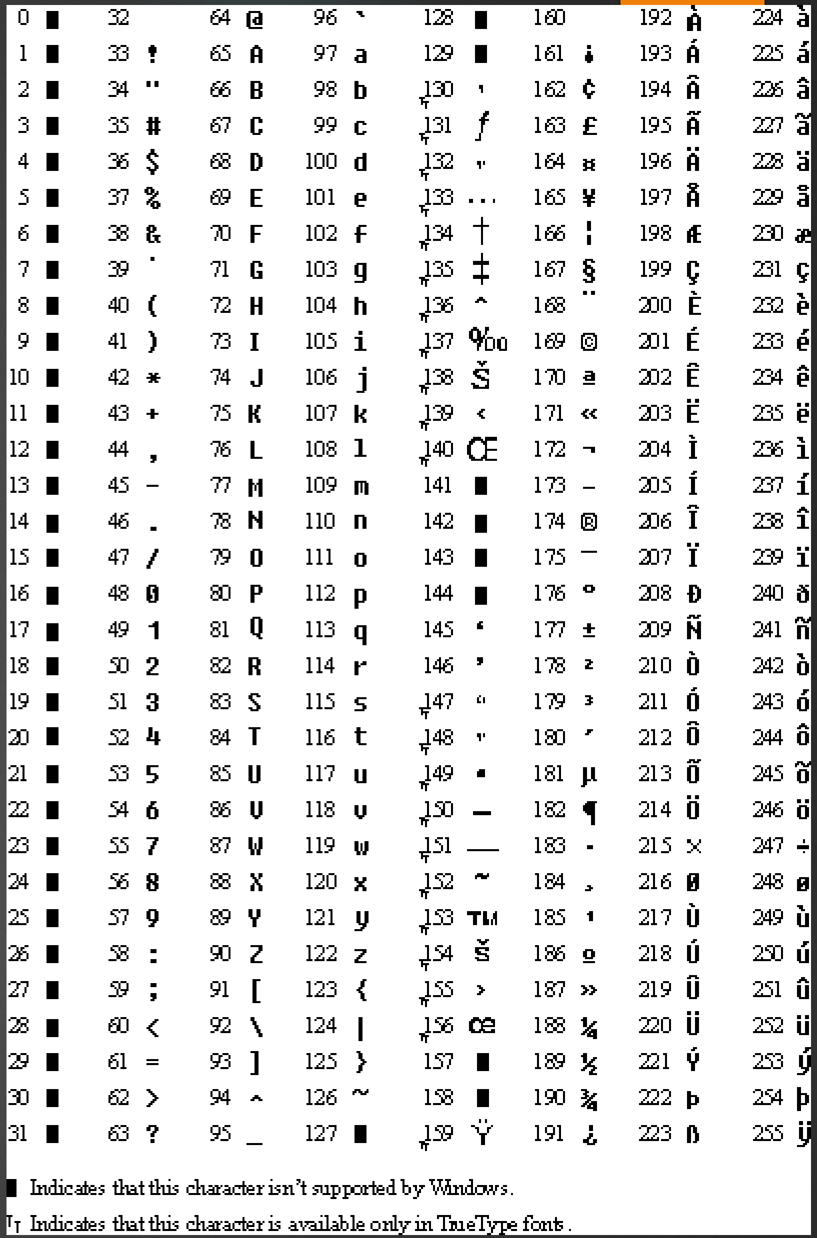


b. Unicode

- More recent
- 16-bit code (2 bytes)
- Can represent over 65,000 characters
  - All the characters used by the world's languages
- Widely used to handle documents that is required to be written in different languages
- Allows localisation of software – standard software can be adapted for use in different cultures by modifying the layout and features



- Chinese Character Set – thousands of different characters



- ANSI Set – Graphical symbols, lines and shapes

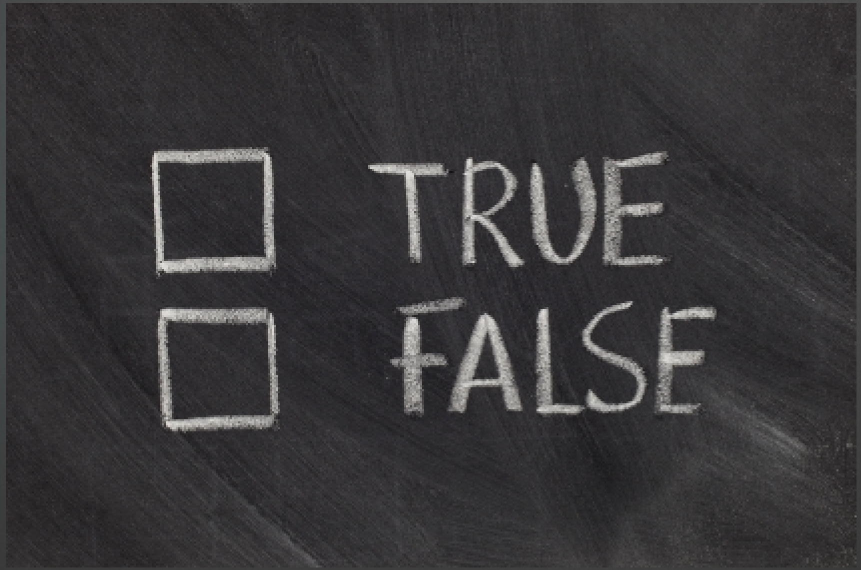
3. Representing Data Types – Numerical Data

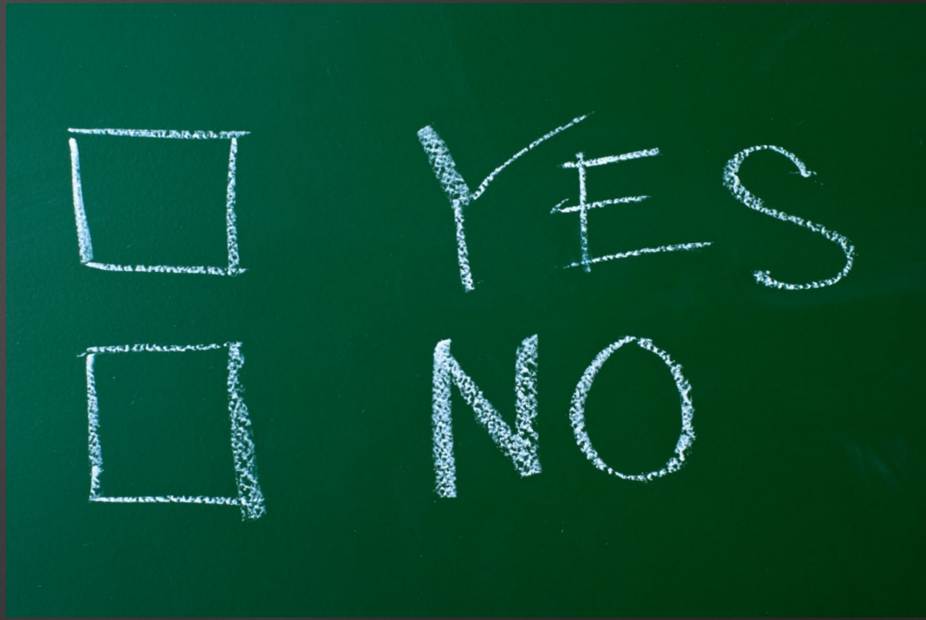
- Integers
- Negative numbers
- Real numbers

- Size: either 1, 2 or 4 bytes
  - Larger number of bytes can store larger numbers
    - 1 byte → 0 to 255
    - 2 bytes → 0 to 65,535

3. Representing Data Types – Boolean Data

- Data with two states
- 0 or 1
- Size: 1 bit





3. Representing Data Types – Character & String Data

- Character → single letter, digit or symbol
- Size: usually 1 byte

- String → sequence of characters stored together
- Size: depends on the number of characters

4. Representation of Numbers

1. Unsigned (Positive) Integers
2. Signed Integers (not in Syllabus)
    1. Sign-Magnitude
    2. One's Complement
    3. Two's Complement
3. Floating Point Numbers (not in Syllabus)

3. Representation of Positive Integers

3. Representation of Positive Integers

Hindu-Arabic	Roman
1	I
5	V
10	X
50	L
100	C
500	D
1 000	M

Positional Weighted System

- The position of a digit in a number carries different weight
- Example (Base 10):
  - Digits in base 10 = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}

9715

9 x 1000	7 x 100	1 x 10	5 x 1

Formal definition {base(radix) **10**}

$$\left(a_n a_{n-1} \ldots a_0\right)_{10} = a_n \times 10^n + a_{n-1} \times 10^{n-1} + \ldots + a_0 \times 10^0$$

9715

9 x 1000 | 7 x 100 | 1 x 10 | 5 x 1

$$9715_{10} = 9 \times 10^3 + 7 \times 10^2 + 1 \times 10^1 + 5 \times 10^0$$

How about other bases?

Base	"Digits" in the system	Example number	Value in base 10
10	0, 1, 2, 3, 4, 5, 6, 7, 8, 9	1011_{10}	$1 \times 10^3 + 1 \times 10^1 + 1 \times 10^0 = 1011_{10}$
2	0, 1	1011_2	$1 \times 2^3 + 1 \times 2^1 + 1 \times 2^0 = 11_{10}$
4	0, 1, 2, 3	1011_4	$1 \times 4^3 + 1 \times 4^1 + 1 \times 4^0 = 69_{10}$
8	0, 1, 2, 3, 4, 5, 6, 7	1011_8	$1 \times 8^3 + 1 \times 8^1 + 1 \times 8^0 = 521_{10}$
16	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F	1011_{16}	$1 \times 16^3 + 1 \times 16^1 + 1 \times 16^0 = 4113_{10}$

Common Bases

- Base 10 = Denary
  - Decimal number formed by digits (0, 1, …, 9)
- Base 2 = Binary
  - Binary number formed by bits (0, 1)
- Base 8 = Octal
  - Octal number formed by octals (0, 1, …, 7)
- Base 16 = Hexadecimal
  - Hexadecimal number formed by hexadigits (0, 1, …, 9, A, B, … , F)

Exercise:
Convert Base-*R* to Decimal

- 1101_2
- 572_8
- $2A_{16}$
- 341_5

Decimal to Binary Conversion

- Method 1
  - Repeated Division by 2
- Method 2
  - Sum of Weight

Method 1: Repeated Division by 2

- Use successive division by 2 until the quotient is 0:
- The remainders form the answer:
  - The first remainder is the least significant bit (LSB)
  - The last remainder is the most significant bit (MSB)

Exercise: Convert Decimal to Binary (Method 1)

89	
24	
100	
73	
127	

Method 2: Sum of Weights

- Determine the set of binary weights whose sum is equal to the decimal number

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

- $9_{10} = 8 + 1 = 2^3 + 2^0 = 1001_2$
- $18_{10} = 16 + 2 = 2^4 + 2^1 = 10010_2$
- $58_{10} = 32 + 16 + 8 + 2 = 2^5 + 2^4 + 2^3 + 2^1 = 111010_2$

Exercise: Convert Decimal to Binary (Method 2)

89		128	64	32	16	8	4	2	1	
24		128	64	32	16	8	4	2	1	
100		128	64	32	16	8	4	2	1	
73		128	64	32	16	8	4	2	1	
127		128	64	32	16	8	4	2	1	

Decimal to Base-R Conversion

- Method 1 is a "sure-fire" method
  - Just repeatedly divide by R
- Method 2 still works, BUT…
  - It is usually harder as you now need to test for multiples of a weight

	3^2	3^1	3^0
	9	3	1
19	0, 1, 2	0, 1, 2	0, 1, 2

Exercise: Convert the Decimal 89 (89_{10}) to Base- R

Base	Answer
5	
8	
16	
10	89

Base-K to Base-J Conversion

- In general,
  - Use the decimal system as the bridge
- Base-K → Base-10 → Base-J

Exercise: Convert Binary to Octal

Binary	Octal
1010111010	
1111101100	
1011000001	
1000111110	
1001000101	
1101101010	

Exercise: Convert Binary to Hexadecimal

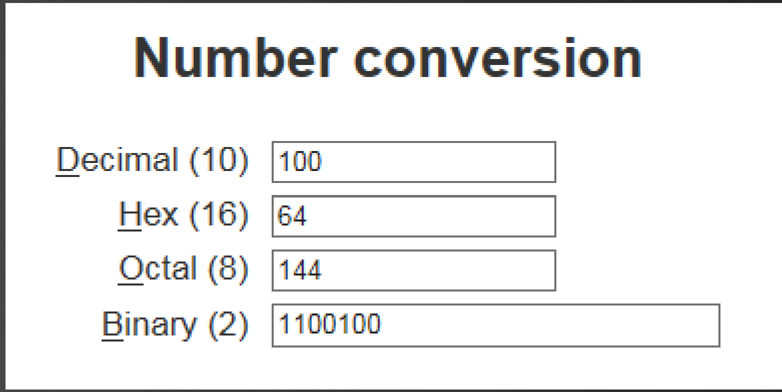
Binary	Hexadecimal
1010111010	
1111101100	
1011000001	
1000111110	
1001000101	
1101101010	

Exercise: Octal ←→ Hexadecimal

Octal	Hexadecimal
537	
123	
65	
	AAA
	BOB
	FACE

Online Tools :

- Number Conversion
  https://coderstoolbox.net/number/



- IEEE-754 Floating Point Converter
  https://www.h-schmidt.net/FloatConverter/IEEE754.html