

Week 1 – Bits & Bytes

Student number: 589932

Assignment 1.1: Bits & Bytes intro

What are Bits & Bytes?

- A bit (binary digit) is a single place or symbol in a binary number, similar to how a digit is a single place in a decimal number. A bit can only be 0 or 1, and by itself it carries very little information because it can represent only two states—such as on/off or true/false.
- A byte consists of 8 bits and is the smallest addressable unit in computer memory. A byte can represent 256 possible values (0–255) and is commonly used to store characters, numbers, and other basic data types.

What is a nibble?

- A nibble is a group of 4 bits, which is exactly half of a byte.
- A nibble can represent 16 possible values, from binary 0000 to 1111.

What relationship does a nibble have with a hexadecimal value?

- Because a nibble contains 4 bits, and 4 bits can express 16 unique combinations, it maps perfectly to one hexadecimal digit (0–F).
- 4 bits represent values from 0 to 15, which is exactly the range of a single hex digit.
- So:
 - 1 nibble (4 bits) → 16 combinations
 - Hexadecimal base = 16
 - Therefore 1 hex digit = 1 nibble

Why is it wise to display binary data as hexadecimal values?

- Binary grows very long and hard to read (e.g., 001110000101001).
- Since each nibble corresponds to a single hex digit, hexadecimal is a much more compact and readable representation of binary data.
- Hexadecimal makes it easier to:
 - interpret large binary sequences
 - reduce errors
 - visualize memory, colors, instructions, and byte patterns
- For example, instead of writing 11111111 00000000, you can simply write FF 00

What kind of relationship does a byte have with a hexadecimal value?

- A byte is 8 bits, which equals two nibbles.
- Since each nibble corresponds to one hex digit, a byte corresponds to two hexadecimal digits.
- So the structure is:
 - 1 byte = 8 bits
 - 8 bits = 2 nibbles
 - 2 nibbles = 2 hexadecimal digits

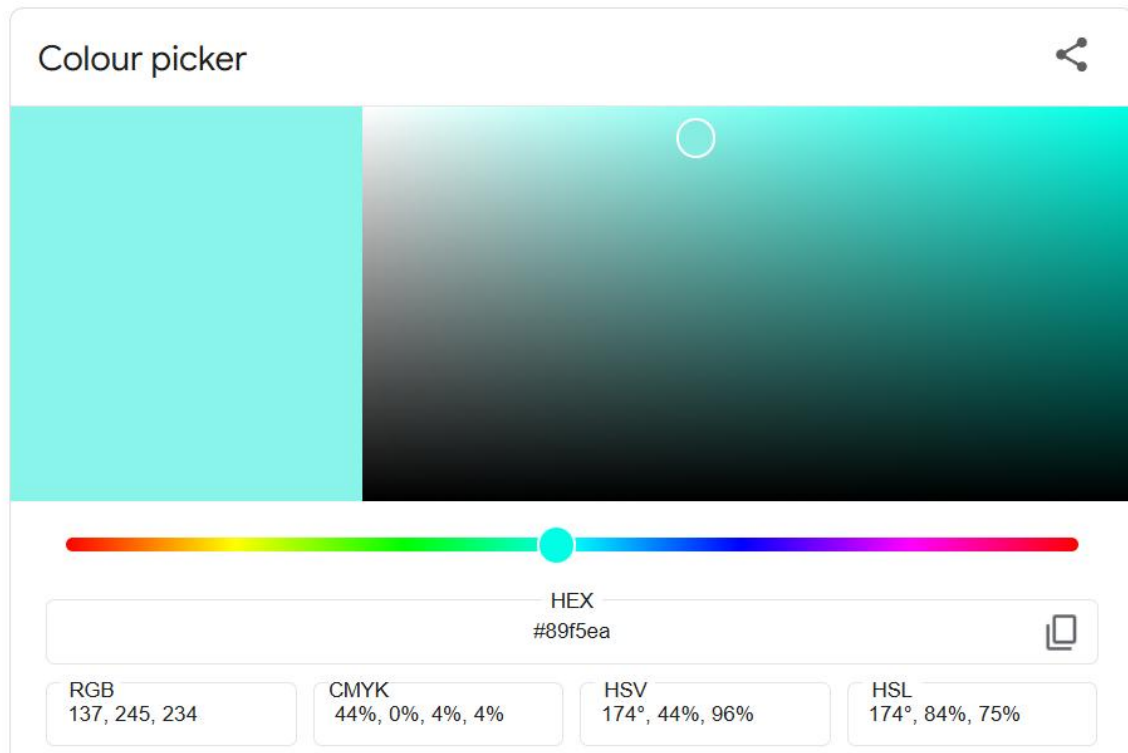
An IPv4 subnet is 32-bit, show with a calculation why this is the case.

- An IPv4 subnet mask is written as four decimal numbers, for example:
 - 255.0.0.0
 - 255.255.0.0
 - 255.255.255.0
- Each of these four numbers is one **octet**, and each octet contains 8 bits.
We can prove this by converting the octets to binary.
- Example 1: 255.0.0.0
 - 255 = 11111111 (8 bits)
 - 0 = 00000000 (8 bits)
 - 0 = 00000000 (8 bits)

- 0 = 00000000 (8 bits)
- Total bits: $8+8+8+8=32$ bits
- Example 2: 255.255.0.0
 - 255 = 11111111 (8 bits)
 - 255 = 11111111 (8 bits)
 - 0 = 00000000 (8 bits)
 - 0 = 00000000 (8 bits)
 - Total bits: $8+8+8+8=32$ bits
- Example 3: 255.255.255.0
 - 255 = 11111111 (8 bits)
 - 255 = 11111111 (8 bits)
 - 255 = 11111111 (8 bits)
 - 0 = 00000000 (8 bits)
 - Total bits: $8+8+8+8=32$ bits
- Every IPv4 subnet mask is made of 4 octets, and each octet contains 8 bits, therefore:
 - $4 \times 8 = 32$ bits
- This is why IPv4 is always 32-bit, no matter which subnet mask you use

Assignment 1.2: Your favourite color

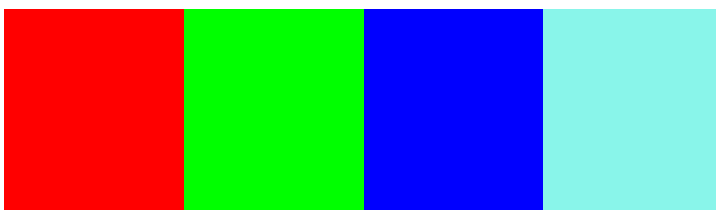
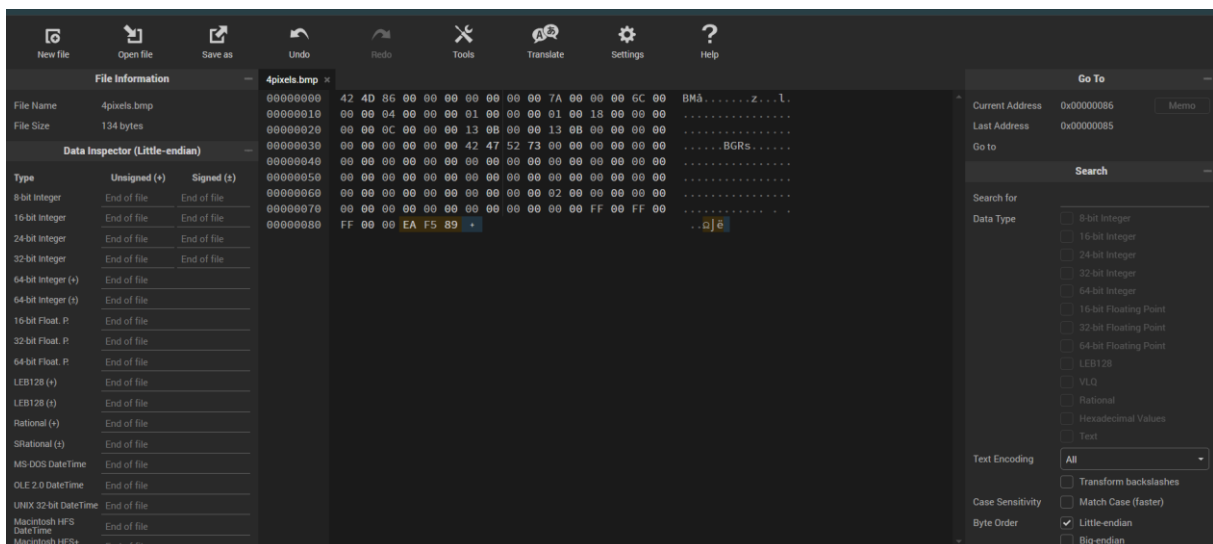
Hexadecimal color code: #89f5ea



Assignment 1.3: Manipulating binary data

Color	Color code hexadecimaal (RGB)	Big Endian	Little Endian
RED	#FF0000	FF 00 00	00 00 FF
GREEN	#00FF00	00 FF 00	00 FF 00
BLUE	#0000FF	00 00 FF	FF 00 00
WHITE	#FFFFFF	FF FF FF	FF FF FF
Favourite (previous assignment)	#89F5EA	89 F5 EA	EA F5 89

Screenshot modified BMP file in hex editor:



Assignment 1.4: Student number to HEX and Binary

Convert your student number to a hexadecimal number and a binary number.

Explain in detail that the calculation is correct. Use the PowerPoint slides of week 1.

My student number is 589932

Division	Quotient	Remainder	Hex digit
$589932 \div 16$	36870	12	C
$36870 \div 16$	2304	6	6
$2304 \div 16$	144	0	0
$144 \div 16$	9	0	0
$9 \div 16$	0	9	9

⇒ Hexadecimal = 0x9600C

Hex	Binary (4 bits)
9	1001
6	0110
0	0000

Hex	Binary (4 bits)
0	0000
C (12)	1100

⇒ Binary = 100101100000000001100

To convert my student number 589932 to hexadecimal, I repeatedly divided the number by 16, because hexadecimal is base-16. Each remainder represents one hexadecimal digit. By reading the remainders from bottom to top, the final hexadecimal value becomes 0x9600C, which matches the calculation shown in the Week 1 slides.

To convert the same number to binary, I used the relationship between hexadecimal and binary, where each hexadecimal digit corresponds to a 4-bit binary group. Converting each digit of 9600C results in the binary sequence:

- 9 → 1001
- 6 → 0110
- 0 → 0000
- 0 → 0000
- C → 1100

Combining these groups produces the final binary value: 100101100000000001100

Ready? Save this file and export it as a pdf file with the name: **week1.pdf**