Week 1 – Bits & Bytes

Student number: 576255

Assignment 1.1: Bits & Bytes intro

What are Bits & Bytes?

A bit (binary digit) is the smallest possible data value a computer uses, and can only be a 0 or a 1, matching the on/off state of electronic circuits. A byte is a group of 8 bits. With 8 bits you can represent the numbers 0 to 255 (2^8 possible combinations). A byte is used as the base unit for things like characters, colours, file sizes and memory.

What is a nibble?

A nibble is a group of 4 bits. Because each bit can be 0 or 1, a nibble can represent decimal values from 0 to 15. It's basically half a byte.

What relationship does a nibble have with a hexadecimal value?

Hexadecimal (base 16) makes use of the symbols 0-9 and A-F, so one hex digit can represent the values from 0 to 15. A nibble is 4 bits, which can also represent 0 to 15.

Because of that, a nibble (4 bits) maps exactly to a hexadecimal digit. For example:
1010 (a nibble) = 10 = A

It's because of that 1-to-1 mapping that hex and nibbles fit together nicely.

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

Raw binary gets long very quickly and is hard to read, example:

1011101010110010…

Hex divides it up in nice chunks of 4 so that same value can be much shorter and cleaner, BA B2. It's still linked directly to the bits (1 hex digit = 4 bits), but it's easier to read, copy, debug and discuss. This is why we often show binary data as hex.

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

A byte is 8 bits. 1 hex digit represents 4 bits so:

1 byte = 8 bits = 2 hex digits

The byte with these bits 11111111 is FF

So, every byte will be written as exactly two hexadecimal characters (like 00, 4A, FF, …)

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

An IPv4 address is four decimal numbers separated by dots, like:

192.168.0.1 or a subnet like 255.255.0.0.

Each of those four numbers is an octet, and as one octet = 1 byte = 8 bits, the total number of bits in an IPv4 address = 4 octets × 1 byte per octet × (8 bits per byte) = 4 × 8 = 32 bits.

If you write an address or subnet in binary, you'll always see 32 bits, for example:

255 = 11111111

255.255.0.0 → 11111111.11111111.00000000.00000000 → 4 × 8 bits = 32 bits in total.

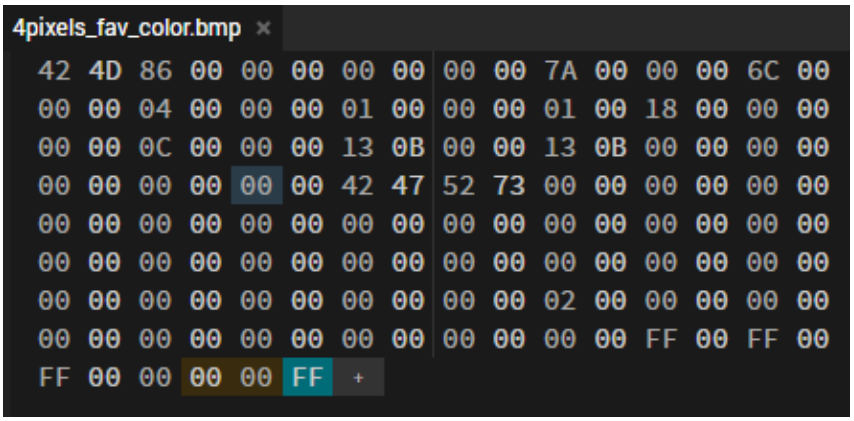
Assignment 1.2: Your favourite color

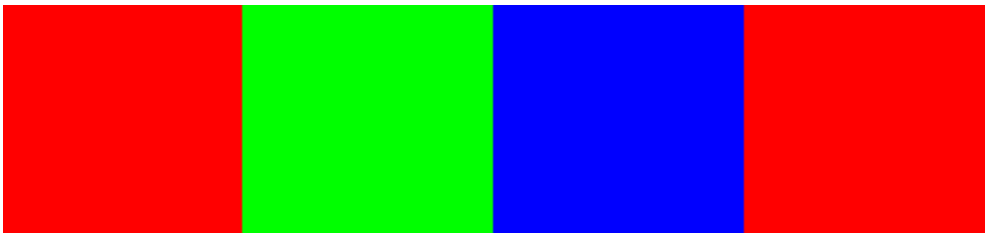
Hexadecimal color code: **#FF0000**

Assignment 1.3: Manipulating binary data

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

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.

Decimal → Hexadecimal

Step	Current value	÷ 16 = quotient	Remainder	Hex digit
1	576266	36016	10	A
2	36016	2251	0	0
3	2251	140	11	B
4	140	8	12	C
5	8	0	8	8

- We stop when the quotient becomes **0**.
- Now we **read the remainders from bottom to top**

8, C, B, 0, A → **8CB0A**

So, **576266 = 8CB0A**

Decimal → Binary

Step	Current value	÷ 2 = quotient	Remainder (bit)
1	576266	288133	0
2	288133	144066	1
3	144066	72033	0
4	72033	36016	1
5	36016	18008	0

6	18008	9004	0
7	9004	4502	0
8	4502	2251	0
9	2251	1125	1
10	1125	562	1
11	562	281	0
12	281	140	1
13	140	70	0
14	70	35	0
15	35	17	1
16	17	8	1
17	8	4	0
18	4	2	0
19	2	1	0
20	1	0	1

Again, we stop when the quotient becomes **0**, and then:

- **Read the remainders from bottom to top**:
  1 0 0 0 1 1 0 0 1 0 1 1 0 0 0 0 1 0 1 0

So, **576266 = 10001100101100001010**