How can an image be represented in binary?

Bitmap images

A digital image showing how each pixel of a bitmap image can be represented in binary using 1 bit

Bitmap images are made up of **individual pixels**. The colour of each pixel is represented as a binary number so the whole image is therefore stored as a series of binary numbers.

In the example on the right the animation zooms into the small image to show the individual pixels. In this example only 1 bit is used to represent each pixel so the binary code for each pixel is 1 or 0, giving 2 possible colours. This image therefore has a colour depth of 1 bit.

If more memory bits are used to represent each pixel then more combinations of binary numbers are possible so more colours are possible in the image.

Vector images

Vector and bitmap images compared, showing the effect of magnification on each

Vector images store information as mathematical instructions rather than as individual pixels. For example, to save a vector image of a circle the software only needs to store:

- the coordinates of the circle centre
- the radius
- the line thickness
- the line style and colour
- the fill type and colour
- the image scale.

This information would be stored using binary codes for the instructions. The result is that the file size of a vector image would be **considerably smaller** than the equivalent bitmap image which would need to store the colour information for every pixel.

Although vector images can have graduated fills these are mathematically defined and cannot represent the level of detail needed for photographic images.

As well as the smaller file sizes, vector images have the advantage that they can be scaled to any size without the loss of detail that would occur with a bitmap image. Also, if a vector image was scaled-up and then saved it would have the same file size as the original.

Why does metadata need to be included in an image file?

A typical example of the metadata stored in a digital image
Metadata is needed in a bitmap image file because the software that displays an image needs to know:

- The **height** and **width** of the image – so each line of the image starts in the correct place.
- The **resolution** – so the image displays at the correct size.
- The **colour depth** – so the correct number of bits are used to represent the colour of each pixel.

An image with the CORRECT colour depth metadata – 1 bit colour depth	**The same data but the WRONG colour depth metadata – 2 bit colour depth**

An image with the CORRECT size metadata – 16 x 24 pixels	**The same data but with the WRONG size metadata – 24 x 16 pixels**
	The original image, before zooming in to reveal the individual pixels.

NOTE: An image file may also contain metadata describing the type and amount of compression in the image, the software used to create the file and, if it is a digital photograph, the date and time the image was taken and the camera/lens settings.

What is colour depth and how does it effect the size of an image file?

Colour depth in bits (n)	Number of possible colours (2^n)	Example binary code(s) used to store the colour information about each pixel
1	2	0, 1
2	4	00, 01, 10, 11
3	8	000, 001, 111 etc.
4	14	0000, 0001, 1110 etc.
8	256	10010101, 11101101 etc.
24	16,777,216	010110101110001101001101

Colour depth describes the number of bits of memory that are used to store the colour information about each pixel in a bitmap image.

With 1 bit colour depth the number of bits used to store the information about each pixel is 1. This allows 2 colours, represented by 0 or 1.

With 3 bit colour depth the number of bits used to store the information about each pixel is 3. This allows 8 colours, represented by the binary codes 000, 001, 010, 011, 100, 101, 110 or 111.
SUMMARY: The greater the colour depth of a bitmap image, the greater the file size because more memory is used to store the colour data about each pixel.

Colour mapping, palette tables and Direct colour

Colour mapping – With low colour depths (up to 8 bit) it is practical to map every colour to a binary code because there are not too many colours involved. This is called **colour mapping** and an example might be the

binary code 110 representing yellow.

Palette tables – The **GIF** file format uses 8 bits for each pixel so each pixel can display 256 different colours. However, every GIF image can have a **different set** of 256 colours because they are stored as part of the file in a **palette table**. This table stores each of the 256 colours using 24 bit direct colour giving 16,777,216 possible colours. Each of the 256 colours in the palette is **indexed** using 8 bits so it is the 8 bit index value that is actually stored for each pixel in a GIF file.

Direct colour – As the colour depth increases, colour mapping and palettes become impractical due to the number of possible colours that the image can display. In higher colour depths (8 bit and above) a system called **direct colour** is used instead. In this the bits allocated to each pixel **separately encode the relative proportions** of **R**ed, **G**reen and **B**lue to specify what is called an **RGB** colour. For example, if 24 bit direct colour was used then the code for a yellow pixel (maximum red and green and zero blue) would be;

- 255, 255, 0 in denary
- 111111111111111100000000 in binary (note the use of 24 bits)
- FFFF00 in hexadecimal

EXPANSION TOPIC – Colour mapping and Direct colour