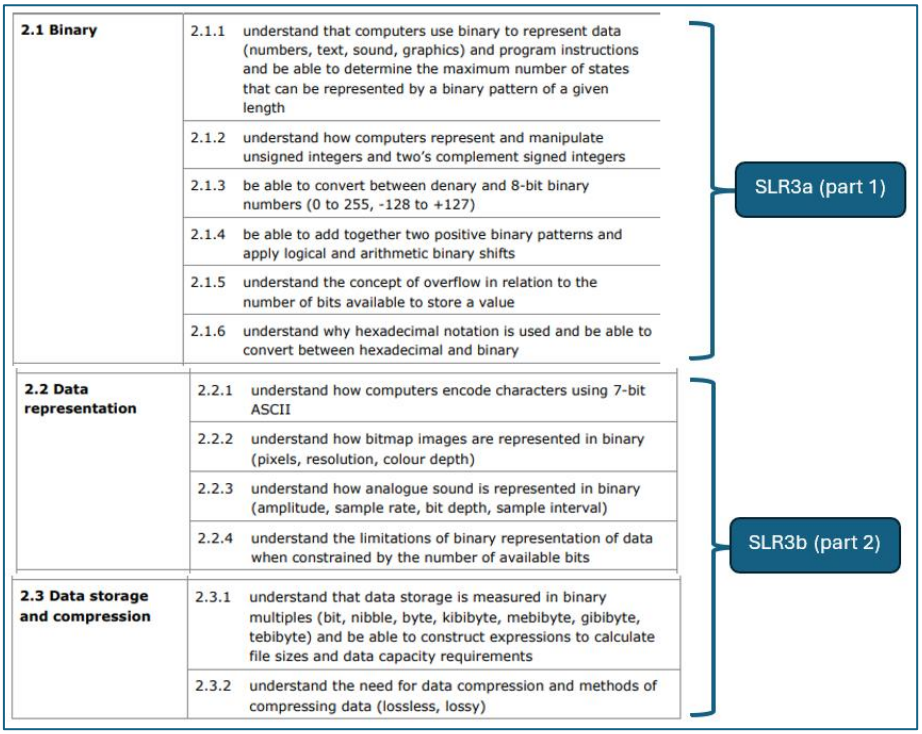
SLR3: Data Representation

edexcel GCSE
www.Python Ninja.co.uk

Edexcel CS Specification:

The SLR3 Topic covers Edexcel specification areas 2.1.1 to 2.3.2



2.1 Binary	2.1.1	understand that computers use binary to represent data (numbers, text, sound, graphics) and program instructions and be able to determine the maximum number of states that can be represented by a binary pattern of a given length
	2.1.2	understand how computers represent and manipulate unsigned integers and two's complement signed integers
	2.1.3	be able to convert between denary and 8-bit binary numbers (0 to 255, -128 to +127)
	2.1.4	be able to add together two positive binary patterns and apply logical and arithmetic binary shifts
	2.1.5	understand the concept of overflow in relation to the number of bits available to store a value
	2.1.6	understand why hexadecimal notation is used and be able to convert between hexadecimal and binary
2.2 Data representation	2.2.1	understand how computers encode characters using 7-bit ASCII
	2.2.2	understand how bitmap images are represented in binary (pixels, resolution, colour depth)
	2.2.3	understand how analogue sound is represented in binary (amplitude, sample rate, bit depth, sample interval)
	2.2.4	understand the limitations of binary representation of data when constrained by the number of available bits
2.3 Data storage and compression	2.3.1	understand that data storage is measured in binary multiples (bit, nibble, byte, kibibyte, mebibyte, gibibyte, tebibyte) and be able to construct expressions to calculate file sizes and data capacity requirements
	2.3.2	understand the need for data compression and methods of compressing data (lossless, lossy)

Edexcel Specification pdf

2.1 Binary

Content Clarification (211):

Understand that a single binary digit (**bit**) can represent either 1 or 0 (corresponding to the electrical states on and off).

Be aware that computers represent, process, store and transmit all data as binary patterns and that the meaning of a group of bits depends on its **context**.

Know how to determine the **maximum number** of **states** that can be represented by a binary pattern of a given length (2n).

Paper 01 SAM, Q1(b) addresses this requirement. Students should be familiar with decimal (base 10), binary (base 2) and hexadecimal (base 16) numbers. Paper 01 Specimen 2, Q3(a) requires students to complete a table giving the number of values per digit of base 2 and base 16 numbers

Content Clarification (212):

Know the difference between an **unsigned and a signed integer.**

Know that a **positive integer** is a whole number with a value greater or equal to zero and a negative integer is a whole number with a value less than zero.

Know that negating a **signed integer** involves changing its sign without changing its value.

Recognise circumstances where it is better to use an **unsigned rather than a signed integer** and vice versa.

Know how signed integers are represented in **two's complemen**t, using the MSB as a negative value.

Know the range of values that can be represented with 8 bits.

You do not need to know about sign and magnitude representation.

Content Clarification (213):

You should be able to **convert** both **unsigned and signed** denary integers into binary and vice versa.

Content Clarification (214):

You should be able to **add** together two 8-bit **binary** numbers.

Understand that **binary subtraction** can be achieved by adding the two's complement of the second number to the first number. i.e. x – y = x + - y.

Know the difference between and be able to perform logical and arithmetic **binary shifts.**

Understand why performing a right shift may result in a lack of **precision**.

Understand that one use for logical binary shifts is to multiply and divide unsigned binary integers by powers of two.

Recognise that – whilst arithmetic binary shifts can be used to divide negative numbers – using left arithmetic shifts to implement multiplication of negative numbers **does not work** because the MSB is not preserved

Content Clarification (215):

Know that **overflow** occurs when the result of a calculation is too large to fit into the location assigned to hold it.

Understand the effect of overflow errors on program outcomes.

Content Clarification (216):

Know that **hexadecimal** is a base 16 number system used as shorthand for binary, and that one hexadecimal digit corresponds to four bits (one **nibble**) and can represent sixteen unique values (0 – F).

Understand reasons why humans choose to use hex in preference to binary.

2.2 Data Representation

Content Clarification (221):

Know that **ASCII** is a standard for encoding characters (letters, numbers, punctuation marks and control codes), and that ASCII assigns each character its own unique numeric value.

You do not need to learn the details of **extended ASCII** or Unicode, but be aware that alternative coding systems exist that permit a wider range of character sets and non-English characters to be represented.

You should be aware that character codes are grouped and **run in sequence** and – given the code for one character – be able to derive the code for another.

Content Clarification (222):

Know that a **pixel** (picture element) is the smallest element of a bit-mapped image and that the size of an image is expressed as width x height in pixels.

Know that the **resolution** of an image refers to its physical size when displayed on screen or in print, and is measured in pixels per inch (ppi).

The higher the resolution, the more pixels per inch and the better the image quality, e.g. a 200 x 100 pixel bitmap with a resolution of 100 ppi would measure 2" x 1", whereas, with a resolution of 200ppi, it would measure 1" x 0.5".

Recognise that an image with a **low resolution** has fewer pixels per inch and may become **pixelated** if stretched to fit into a larger space.

You should know that colour depth is the number of bits used to represent the colour of a pixel. The greater the number of bits used, the more tones/colours can be represented.

You should understand that the size (in pixels) and **colour depth** of an image determine its file size. The greater the number of pixels and the greater the colour depth, the larger the file size will be.

You should know that **metadata** is information about the properties of an image and adds to the file size of an image.

You should be able to convert binary data arranged into a bitmap image and be able to generate the binary code for a bitmap.

Content Clarification (223):

Know the difference between a **continuous analogue** and a discrete **digital** signal, and understand why an analogue sound is never fully reproducible in a digital format.

- the **amplitude** of a sound wave determines the sound's loudness
- a **sample** is a measure of amplitude at a point in time
- **sample rate** is the number of samples taken per second, measured in **hertz**,
- **sample interval** is the time between samples.
- **bit depth** is the number of bits used to represent each sound sample

Know that a **CD**-quality soundtrack uses a sample rate of **44.1KHz**, a bit depth of 16 and is recorded in stereo.

Understand how the choice of sampling rate, bit depth and sampling interval affect the accuracy of a digital representation.

You should understand how an audio sound is converted from analogue to digital.

Content Clarification (224):

You should understand that the **number of available bits** determines how a character set, an image, a sound, etc., is represented – the more bits there are, the greater the range of unique values.

2.3 Data Storage and Compression

Content Clarification (231):

Be familiar with and use **base 2 binary** multiples (IEC units) for constructing expressions to calculate file size and data capacity.

Use of denary multiples in this context is not acceptable.

You should be able to **rank the units** of measurement in size order.

You should be able to **convert larger units** to smaller one by multiplying by 1024, and smaller to larger units by dividing by 1024.

You should be able to construct expressions to calculate:

- the **file size of an image** (width x height x colour depth), or – given the file size and the values of any two of the variables – calculate the value of the missing one.
- the **file size of an audio recording** (sample rate * bit depth * duration), or – given the file size and the values of any two of the variables – calculate the value of the missing one.

Content Clarification (232):

Know that **compression** is a technique for reducing file size and understand why reducing the size of a file is sometimes necessary or desirable.

Understand that using a **lossy** algorithm to compress a file results in data being permanently lost, whereas using a **lossless** algorithm allows the original file to be exactly reconstructed from the compressed data.

Understand that different types of data lend themselves to different compression methods.

Content Clarification guide - issue 3

==========