

Data Representation

GCSE Computer Science

Topic Covered



- Unit of Data
- Binary to Denary vice versa
- Binary to Hexadecimal, vice versa
- Hexadecimal to Denary, vice versa
- Binary Addition
- Binary Shift (Multiplication and Division)
- Representations of Characters
- Representation of Images
- Representation of Sound

Assessment

Assessment Criteria	Achieved (Student)	Achieved (Teacher)
<ul style="list-style-type: none"> a) I know different types of data: text, number. b) I know programs use different data types 		
<ul style="list-style-type: none"> a) I Know that computers use binary to understand what to do b) I can explain the difference between data and information 		
<ul style="list-style-type: none"> a) I can change a denary number to binary number and vice versa b) I can convert positive denary whole numbers (0–255) into 2 digit hexadecimal numbers and vice 		
<ul style="list-style-type: none"> a) I can explain that binary is at the base of all actions on a computer and all programs have to translate to binary to work b) I understand the relationship between the number of bits per character in a character set c) I understand the number of characters which can be represented (for example ASCII, extended ASCII and Unicode). 		
<ul style="list-style-type: none"> a) I can complete binary addition b) I can explain how data is stored in memory c) I understand how numbers, images, sound and character sets use the same bit patterns. d) I recognised how Metadata is included in the file 		
<ul style="list-style-type: none"> a) I understand the relationship between data representation and data quality b) I understand the need for compression and can explain the two types of compressions techniques (Lossy and Lossless) 		
<ul style="list-style-type: none"> a) I know different types of data: text, number. b) I know programs use different data types 		
<ul style="list-style-type: none"> a) I Know that computers use binary to understand what to do b) I can explain the difference between data and information 		
<ul style="list-style-type: none"> a) I can change a denary number to binary number and vice versa b) I can convert positive denary whole numbers (0–255) into 2 digit hexadecimal numbers and vice 		



Section 1: Unit of Data

Data Representation



Today's learning Intentions

- **Know:** what binary is
- **Be able** to understand the unit of Data such as bit, byte, megabyte, gigabyte etc..
- **Understands** Understand binary and denary base system

Keywords

Binary

Denary

Byte

Nibble

Megabyte

Gigabyte

Binary Number Units: Bit

- The smallest unit of binary data is a BIT (Binary digiT)
 - It has two states, 0 and 1.
- To do anything useful, you need lots of bits.
- Most computers group bits together to make it easier to store them.
- Usually memory locations hold 8 bits.
 - The amount stored in one location is called a byte.
 - This is sometimes called an octet.

Binary Number Units: Half a Byte

- Sometimes it is useful to address half a byte.
 - Maybe you don't need a whole byte to store something.
 - Half a byte is called a nibble (sometimes spelt nybble).

0	1	0	1	0	1	0	1
bit	bit	bit	bit	bit	bit	bit	bit
nibble				nibble			
byte							

- For most serious work, you need much larger stores of data.
 - As computers get more powerful and we use them more, we need to store more data.

Binary Number Units: Kilobyte (KB)

- Kilobyte (kB)
 - this is 1024 bytes
 - it is not 1000 bytes because computers work with binary numbers, so it is 2^{10} which is the nearest power of two to 1000.
- These days, a kilobyte of data is not much, so we often use much bigger groups.

Binary Number Units: Megabyte

- Megabyte (MB)
 - this is 1024 kilobytes or 2^{20} (1048576) bytes.
- 1 MB is enough data for
 - a 1024×1024 pixel bitmap image with 256 colours;
 - 1 minute of 128 kbit/s MP3 compressed music;
 - 6 seconds of uncompressed CD audio;
 - a typical book.

Binary Number Units: Gigabyte (GB)

- Gigabyte (GB)
 - this is 1024 megabytes or 2^{30} bytes.
- 1 GB is enough data for:
 - one hour of standard video;
 - seven minutes of high-definition video;
 - 114 minutes of uncompressed CD-quality audio.
- A DVD-R can hold 4.7 GB
- A dual-layered Blu-ray[®] disk can hold 50 GB.

Binary Number Units: Terabyte(TB)

- Terabyte (TB)
 - this is 1024 gigabytes or 2^{40} bytes.
- Yahoo groups has over 40 TB of data.
- Wikipedia has over 6 TB of data.
- As time goes on, we will get used to even larger units of data.

Binary Number Units

Smallest unit	1 bit (0 or 1)
8 bits	1 byte
1024 bytes	1 kilobyte
1024 kilobytes	1 megabyte
1024 megabytes	1 gigabyte
1024 gigabytes	1 terabyte
1024 terabyte	1 petabyte

Activity

Learning Intentions Covered:

Know: what binary is

Be able to understand the unit of Data such as bit, byte, megabyte, gigabyte etc..

Understands Understand binary and denary base system

In your **Data Representation** folder in your area

Open **Year 10 Data Representation Booklet**

Complete:

➤ **All task in Part 1 - Unit of Data**

Section 2 – Binary to Denary

Computer System



Today's learning Intentions

- **Know:** binary and denary base system
- **Be able** convert positive denary whole numbers (0–255) into 8 bit binary
- **Understands** binary overflow

Keywords

Binary Denary Byte Bit Pattern Binary Overflow

Decimal or Denary

- Numbers can be expressed in many different ways.
- We usually use **decimal** or **denary**.
 - Denary numbers are based on the number 10.
 - We use **ten** digits: 0,1,2,3,4,5,6,7,8,9.
 - When we put the digits together, each column is worth **ten** times the one to its right.

Decimal or Denary

- So, the denary number 5162 is

Place value	1000	100	10	1
Digit	5	1	6	2

5 x 1000 =	5000
1 x 100 =	+100
6 x 10 =	+60
2 x 1 =	+2
Total	5162

Binary

- Binary is base 2.
- This means that each column or position can have one of 2 possible values

0 or 1

8	4	2	1
1	0	0	1

- In base 10 the columns go up in powers of 10.
- In base 2 they go up in powers of 2.
- Each column value is the double of the previous

Converting Binary to Denary

CONVERTING BASE 2 TO BASE 10

How to convert **111001** into a denary

- **Step 1:** Draw the table (starting at 1 to 128)
- **Step 2:** Add the binary number to the table
- **Step 3:** Add a numbers together where there is a 1 in the column

128	64	32	16	8	4	2	1
0	0	1	1	1	0	0	1

Each column which has a 1 in it gets counted.

So we have

$$32+16+8+1=57$$

Converting Binary to Denary

- It is simpler to make machines that only need to distinguish two states, not ten. That is why computers use **binary** numbers.
- Each column is worth 2 times the value on its right:

128	64	32	16	8	4	2	1

- So 10010111 is:

128	64	32	16	8	4	2	1
1	0	0	1	0	1	1	1

128

+16

+4

+2

+1

=151

Activity

Learning Intentions Covered:

Know: binary and denary base system

Be able to convert binary to denary and vice versa

Understands binary overflow

In your **Data Representation** folder in your area

Open **Year 10 Data Representation Booklet**

Complete:

➤ **All task in Part 2 - Binary to Denary**

Section 3 – Denary to Binary

Example: Denary Number = 12

How to convert **14** into a Binary

- **Step 1:** Draw the table (starting at 1 to 8)
- **Step 2:** Find out all the numbers we need to add together to get the total
- **Step 3:** Turn all the numbers we add together **ON** by adding **1** under them and **OFF** on the rest by adding **0**

128	64	32	16	8	4	2	1
0	0	0	0	1	1	1	0

$$8 + 4 + 2 = 14$$

So the number 14 in binary = 0000**1110**

Example: Denary Number = 141

How to convert **141** into a Binary

- **Step 1:** Draw the table (starting at 1 to 8)
- **Step 2:** Find out all the numbers we need to add together to get the total
- **Step 3:** Turn all the numbers we add together **ON** by adding **1** under them and **OFF** on the rest by adding **0**

128	64	32	16	8	4	2	1
1	0	0	0	1	1	0	1

$$128 + 8 + 4 + 1 = 141$$

So the number 141 in binary = **10001101**

Denary to Binary (Divide by 2)

To convert from **base 10 (denary)** to **base 2 (binary)** simply divide by 2 repeatedly, recording the remainders until the answer is 0.

To convert 135 from base 10 to binary:

135	÷ 2	67	Remainder	1
67	÷ 2	33	Remainder	1
33	÷ 2	16	Remainder	1
16	÷ 2	8	Remainder	0
8	÷ 2	4	Remainder	0
4	÷ 2	2	Remainder	0
2	÷ 2	1	Remainder	0
1	÷ 2	0	Remainder	1

The answer is the remainder column starting at the last value



1 0 0 0 0 1 1 1

Denary to Binary (MSB)

Method:

Start with the Most Significant Bit (MSB) – that's the one on the left!

For each bit

If Number is **Greater** or **Equal** the current bit

Add on **1** to the answer

Number = Number – current bit

Else

Add on **0** to the answer

Next bit

Denary to Binary (MSB)

EXAMPLE: DENARY NUMBER = 121

128	64	32	16	8	4	2	1
Is 121 \geq 128?	No					Add on 0 to answer	0
Is 121 \geq 64?	Yes	Number = $121 - 64 = 57$				Add on 1 to answer	01
Is 57 \geq 32?	Yes	Number = $57 - 32 = 25$				Add on 1 to answer	011
Is 25 \geq 16?	Yes	Number = $25 - 16 = 9$				Add on 1 to answer	0111
Is 9 \geq 8?	Yes	Number = $9 - 8 = 1$				Add on 1 to answer	01111
Is 1 \geq 4?	No					Add on 0 to answer	011110
Is 1 \geq 2?	No					Add on 0 to answer	0111100
Is 1 \geq 1?	Yes	Number = $1 - 1 = 0$				Add on 1 to answer	01111001
Answer = 01111001							

Denary to Binary (MSB)

EXAMPLE: DENARY NUMBER = 127

128	64	32	16	8	4	2	1
-----	----	----	----	---	---	---	---

Is 127 \geq 128?	No					Add on 0 to answer	0
Is 127 \geq 64?	Yes	Number = 127 - 64 = 63				Add on 1 to answer	01
Is 63 \geq 32?	Yes	Number = 63 - 32 = 31				Add on 1 to answer	011
Is 31 \geq 16?	Yes	Number = 31 - 16 = 15				Add on 1 to answer	0111
Is 15 \geq 8?	Yes	Number = 15 - 8 = 7				Add on 1 to answer	01111
Is 7 \geq 4?	Yes	Number = 7 - 4 = 3				Add on 1 to answer	011111
Is 3 \geq 2?	Yes	Number = 3 - 2 = 1				Add on 1 to answer	0111111
Is 1 \geq 1?	Yes	Number = 1 - 1 = 0				Add on 1 to answer	01111111

Answer = 01111111

Activity

Learning Intentions Covered:

Know: binary and denary base system

Be able to convert binary to denary and vice versa

Understands binary overflow

In your **Data Representation** folder in your area

Open **Year 9 Data Representation Booklet**

Complete:

➤ **All task in Part 3 - Denary to Binary**

Binary Overflow

- Computers have fixed sizes to store numbers.
- These cannot get any bigger.
- That means that if you try and store a number which is too big you will get strange results.

This situation is called **OVERFLOW**.

Binary Overflow Explain

256	128	64	32	16	8	4	2	1
1	0	1	1	0	0	1	0	0

- Let us try and store 356 into 8 bits.
- The binary for this is above.
- However we need 9 bits to store it....

Binary Overflow Explain

256	128	64	32	16	8	4	2	1
1	0	1	1	0	0	1	0	0

- The bit on the right will be deleted.
- It has “overflowed”.
- This leaves us with the binary value 01100100 which is 100 in denary.
- If you tried to store 356 into a 8 bit number you would be left with 100!

Binary Trivia to impress your mates with...

- Ever entered 2 very large numbers into a calculator and multiply them. You will get an error.
- The reason for this is that the calculator stores the numbers and the result in binary!
- So your answer has overflowed (is bigger than what the calculator can handle).
- The calculator knows this which is why it displays error instead of an incorrect value.

It is not because of the size of the screen! Scientific calculators have the exact same problem as cheap ones!

So do calculators on mobile phones. So does excel and your computers calculator!!

Binary Overflow Activity

What is the actual value stored for the following denary numbers. You can only use 8 bits!!

432 = 110110000 **But** 0110000

541 = 1100011101 **But** 00011101

270 = 100001110 **But** 00001110

412 = 110011100 **But** 10011100

Make notes on binary in your Power Point

Starter

Convert the following Binary to Denary and Denary to binary
You can only use 8 bits!!

$$11100000 = \mathbf{224}$$

$$10110000 = \mathbf{176}$$

$$10011101 = \mathbf{157}$$

$$240 = \mathbf{11110000}$$

$$255 = \mathbf{11111111}$$

$$121 = \mathbf{10011101}$$

Convert the following Denary to Hexadecimal

$$142 = \mathbf{8E}$$

$$157 = \mathbf{9D}$$

Activity

Learning Intentions Covered:

Be able to convert binary to denary and vice versa

In your **Data Representation** folder in your area

Open **Year 10 Data Representation Booklet**

Complete:

➤ **All task in Part 2 and 3 - Binary to Denary**

Section 4 – Hexadecimal to Binary



Today's learning Intentions

- **Know:** what Hexadecimal base system is
- **Understand** why programmer use Hexadecimal
- **Be able to** convert Hexadecimal to Binary vice versa

Keywords

Hexadecimal

Binary

Most Significant Bit

Less Significant Bits

Hexadecimal

- Programmers often write numbers down in **hexadecimal (hex)** form.
- Hexadecimal numbers are based on the number 16.
- They have 16 different digits:
0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
- Each column is worth 16 times the one on its right.

256

16

1

Hexadecimal Base System

Remember:

- Denary is base **10** i.e. 0123456789
- Hex is base **16** i.e. 0123456789 ABCDEF

Denary	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Hex	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Converting Binary to Hexadecimal

- To convert from binary to hexadecimal is straightforward.
- Simply take each group of four binary digits, (nibble) starting from the right and translate into the equivalent hex number.
- Example **11110011**

Binary	1	1	1	1		0	0	1	1
Hex			F					3	

Converting Hexadecimal to Binary

- To convert from hexadecimal to binary simply reverse the process, though you may prefer to go via denary.
- Treat each digit separately.
- Example **DB**

Hex	D					B			
Denary	13					11			
Binary	1	1	0	1		1	0	1	1

- So DB in hexadecimal is 11011011 in binary.

Activity

Learning Intentions Covered:

Know: what Hexadecimal base system is

Understand why programmer use Hexadecimal

Be able to convert Hexadecimal to Binary vice versa

In your **Data Representation** folder in your area

Open **Year 9 Data Representation Booklet**

Complete:

➤ **All task in Part 4 - Binary to Hexadecimal**

Section 5 – Hexadecimal to Denary

Hexadecimal Base System

Remember:

- Denary is base **10** i.e. 0123456789
- Hex is base **16** i.e. 0123456789 ABCDEF

Denary	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Hex	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Converting from 2-digit Hex to Denary

Method 1:

1. Convert left number (MSB) to denary value & multiply by 16
2. Convert right denary (LSB) and add to results from Step 1

Example:

Convert Hexadecimal value **F6** to denary

MSB Denary Value	MSB *16	LSB Denary Value	Total
15	$15 * 16 = 240$	6	$240 + 6 = 246$

Converting from 2-digit Hex to Denary

Method 2:

1. Convert hex 2 numbers into two binary nibble
2. Add the two binary numbers together
3. Convert the binary

Example:

Convert Hexadecimal value **F6** to denary

F6	
F = 15	6
1111	0110
Answer = 11110110	

128	64	32	16	8	4	2	1
1	1	1	1	0	1	1	0
128	64	32	16	0	4	2	0
$128 + 64 + 32 + 16 + 4 + 2 = 246$							
F6 = 146							

Converting from 2-digit Hex to Denary

Method 3:

- To convert from hexadecimal to denary all we do is multiply the numbers by their place values and add them together.
- For example, the hexadecimal number **8D**.

Place Value	256	16	1
Hex digits	0	8	D
Denary	0	=8*16	=13*1
	0	128	13

- $0 + 128 + 13 = 141$
- So, **8D** is **141** in denary.

Activity

Learning Intentions Covered:

Know: what Hexadecimal base system is

Understand why programmer use Hexadecimal

Be able to convert Hexadecimal to Binary vice versa

In your **Data Representation** folder in your area

Open **Year 10 Data Representation Booklet**

Complete:

➤ **All task in Part 5 - Denary to Hexadecimal**

Section 6 – Denary to Hexadecimal

Converting Denary to Hexadecimal

Method 1

- We can convert denary numbers to hexadecimal by repeated division just as we did to get binary numbers.
- Take the denary number **237**.

Denary Value	/16 Whole Number	/16 Remainder	
237	14	13	
	Hex Equivalent	Hex Equivalent	ANSWER
	E	D	ED

- We have the hexadecimal values 14 and 13 as remainders.
- 14 in hexadecimal is E.
- 13 in hexadecimal is D.
- So, reading from the bottom again **237** in hexadecimal is **ED**.

Converting Denary to Hexadecimal

Method 2

1. Convert denary number into 8-bit binary
2. Convert 1st nibble into 1st Hex digit and Vice versa
3. Convert 2nd nibble into 2nd Hex digit and Vice versa

Example:

Convert **237** to 2-digit hexadecimal

128	64	32	16	8	4	2	1
1	1	1	0	1	1	0	1
0	64	32	0	8	4	0	1
$128 + 64 + 32 + 8 + 4 + 1 = 237$							

11101101	
1110	1101
14 = E	13 = D
Answer = ED	

Why Used Hexadecimal

- Large binary numbers are hard to remember
- Programmers use hexadecimal values because:
 - each digit represents exactly 4 binary digits;
 - hexadecimal is a useful shorthand for binary numbers;
 - hexadecimal still uses a multiple of 2, making conversion easier whilst being easy to understand;
 - converting between denary and binary is relatively complex;
 - hexadecimal is much easier to remember and recognize than binary;
 - this saves effort and reduces the chances of making a mistake.

Why Use Hexadecimal

- It is shorter way of writing binary
 - each hex digit is equivalent to four binary digits (nibble).
 - an 8-bit binary number can be written using only two different **hex** digits - one **hex** digit for each nibble
- It is easy to remember Hexadecimal than a binary

Activity

Learning Intentions Covered:

Know: what Hexadecimal base system is

Understand why programmer use Hexadecimal

Be able to convert Hexadecimal to Binary vice versa

In your **Data Representation** folder in your area

Open **Year 10 Data Representation Booklet**

Complete:

- **All task in Part 5 - Hexadecimal to Denary**
- **All task in Part 6 - Denary to Hexadecimal**

Section 7 – Binary Addition

The LMC Instruction



Today's learning Intentions

- **Know:** What binary addition is
- **Understand** Binary Addition rules
- **Be able to** add to 8-bit binary together and deal with Overflow

Keywords

Binary

Binary Addition

Byte

Nibble

Binary Addition

- **The Rules for binary addition:**
 - $0 + 0 = 0$
 - $0 + 1 = 1$
 - $1 + 1 = 0$ carry 1
 - $1 + 1 + 1 = 1$ carry 1
- Add the binary equivalents of denary 4 + 5.

Denary	Binary			
4	0	1	0	0
+ 5	0	1	0	1
= 9	1	0	0	1
Carry		1		

Adding Two 8-bit Numbers

	0	1	1	0	0	1	1	1
+	0	0	0	1	0	1	1	0
								1

You always start from the right side and take each column in turn.

$$1 + 0 = 1 !$$

Adding Two 8-bit Numbers

	0	1	1	0	0	1	1	1
+	0	0	0	1	0	1	1	0
							0	1

carr
y

1

$1 + 1 = 0$ carry 1

When you carry it will go onto the next column

Adding Two 8-bit Numbers

	0	1	1	0	0	1	1	1
+	0	0	0	1	0	1	1	0
						1	0	1

carr
y

1 1

Adding Two 8-bit Numbers

	0	1	1	0	0	1	1	1
+	0	0	0	1	0	1	1	0
					1	1	0	1

carr

y

1 1

Adding Two 8-bit Numbers

	0	1	1	0	0	1	1	1
+	0	0	0	1	0	1	1	0
				1	1	1	0	1

carr

y

1 1

Adding Two 8-bit Numbers

	0	1	1	0	0	1	1	1
+	0	0	0	1	0	1	1	0
			1	1	1	1	0	1

carr

y

1 1

Adding Two 8-bit Numbers

	0	1	1	0	0	1	1	1
+	0	0	0	1	0	1	1	0
		1	1	1	1	1	0	1

carr

y

1 1

Adding Two 8-bit Numbers

	0	1	1	0	0	1	1	1
+	0	0	0	1	0	1	1	0
	0	1	1	1	1	1	0	1

carr

y

1 1

Binary Numbers Overflow

- Sometimes we run into problems.
- Suppose we have eight bits in each location.
- When we add the binary equivalent of denary 150 + 145:

Denary		Binary							
150		1	0	0	1	0	1	1	0
+145		1	0	0	1	0	0	0	1
= 39	(1)	0	0	1	0	0	1	1	1
Carry		1			1				

- There is no room for a carry so it is lost and we get the wrong answer, 39 instead of 295.
- When there isn't enough room for a result, this is called **overflow** and produces an **overflow error**.

Activity

Learning Intentions Covered:

Know: What binary addition is

Understand Binary Addition rules

Be able to add to 8-bit binary together and deal with Overflow

In your **Data Representation** folder in your area

Open **Year 10 Data Representation Booklet**

Complete:

➤ **All task in Part 7 - Binary Addition**

Section 8 – Binary Shift

Binary Shift



Today's learning Intentions

- **Know:** What Binary Shift is
- **Understand** How binary shift works
- **Be able to** apply binary left and right shift

Keywords

Left Shift

Right Shift

Binary Shift

- Below is an example of a left bit shift of 1 place.
- What do you notice?

**Example of a
'left bit shift' of
1 place.**

32	16	8	4	2	1	
0	1	1	0	1	1	=27
1	1	0	1	1	0	=54

- A binary shift is a neat way of multiplying or dividing a number in powers of two.
- Each time a LEFT BINARY SHIFT occurs, the number will double in size.
- Each time a RIGHT BINARY SHIFT occurs, the number will half in size.

Binary Shift

‘LEFT’ Binary Shifts (*Multiplying by powers of 2*)

- Below is another example of left bit shifting.
- Notice how as each of the bits shift to the left, any gaps created from the right are filled with a zero.

	128	64	32	16	8	4	2	1	
	0	0	0	1	1	0	1	1	=27
1LBS	0	0	1	1	0	1	1	0	=54
2LBS	0	1	1	0	1	1	0	0	=108
3LBS	1	1	0	1	1	0	0	0	=216

Binary Shift

‘RIGHT’ Binary Shifts (*Dividing by powers of 2*)

- Below is an example of right bit shifting.
- Notice how as each of the bits shift to the right, any gaps created from the left are filled with a zero...
- ...and as bits move to the right and off the edge – they are ignored.

	128	64	32	16	8	4	2	1	
	1	1	0	1	1	0	0	0	=216
1RBS	0	1	1	0	1	1	0	0	=108
2RBS	0	0	1	1	0	1	1	0	=54
3RBS	0	0	0	1	1	0	1	1	=27

Starter Activity

Activity 1: Complete the following (Left Binary Shift)

12×4

2×16

8×8

5×4

128	64	32	16	8	4	2	1
-----	----	----	----	---	---	---	---

Activity 2 Complete the following (Right Binary Shift)

$218 \div 4$

$32 \div 2$

$240 \div 16$

$180 \div 4$

128	64	32	16	8	4	2	1
-----	----	----	----	---	---	---	---

Activity

Learning Intentions Covered:

Know: What Binary Shift is

Understand How binary shift works

Be able to apply binary left and right shift

In your **Data Representation** folder in your area

Open **Year 10 Data Representation Booklet**

Complete:

➤ **All task in Part 8 - Binary Shift**

Section 9 – Character Representation

Character Representation



Today's learning Intentions

- **Know:** What character set is
- **Understand** the use of binary codes to represent characters
- **Understand** the relationship between the number of bits per character in a character set
- **Understand** the difference between ASCII, extended ASCII and Unicode.

Keywords

Character set

ASCII

EASCII

Unicode

Binary Characters

- Last few lessons you saw how to store numbers in binary.
- Storing characters is actually very similar!
- What we do is assign each letter a number and then simply convert the number into binary.
- The numbers we assign to each letter is known as a character set.

Character	Value	Binary
A	1	0000 0001
B	2	0000 0010
C	3	0000 0011
D	4	0000 0100
E	5	0000 0101
F	6	0000 0110

Representing Characters

- Computers can only work with binary data.
- Binary data can easily represent numbers.
- If we make each number represent a letter, or character, then we can store characters.
- One common code (ASCII) uses seven bits to store each character.
- This is enough for 127 (128) different characters. Enough for all the English language letters plus many symbols such as !"£\$%.

ASCII Codes

- ASCII was agreed to deal with basic textual messages:

All the main alphabetic characters, upper and lower case	52 characters
All the numeric symbols, 0–9	10 characters
32 punctuation and other symbols, and ‘space’	33 characters
32 codes reserved for non-printable control codes	32 characters

- In total 127 codes (95 printable, 32 non-printable) plus the null character 00000000 used as a control character but with no associated symbol.

ASCII Characters

- ASCII = American Standard Code for Information Interchange:

Binary	Denary	Hex	Character
100 0001	65	41	A
100 0010	66	42	B
100 0011	67	43	C
100 0100	68	44	D
100 0101	69	45	E
100 0110	70	46	F
100 0111	71	47	G

ASCII Characters

- In a full table of ASCII values you will see that the alphabetical symbols are in numerical order, but lower case characters have higher values than upper case ones.

Binary	Denary	Character		Binary	Denary	Character
1000001	65	A		1100001	97	a
1000010	66	B		1111001	121	y
1011010	90	Z		1111010	122	z

- When the system sorts using ASCII values, the word 'Zebra' comes before 'apple'.

Extended Binary Coded Decimal (EBCDIC)

- 128 characters are not enough to encode all languages.
- There are other systems that use more than 7 bits.
- EBCDIC is the Extended Binary Coded Decimal Interchange Code – this is used in IBM mainframes.
- It uses 8 bits so can encode 256 different characters.

Unicode

- In order to store character sets for languages like mandarin we need more than 8 bits.
- Unicode extends Ascii by using 16 bits for each character.
- Unicode can store 65, 535 different characters.
- Unicode is become more widely used as computers becoming more widely used across the globe.

Unicode

- Unicode is more common these days.
- Unicode comes in different variants, but can encode more than 107,000 characters, covering 90 languages.
- For example,
 - Հայերեն
 - हिन्दी
 - ಕನ್ನಡ
 - ქართული
 - कश्मीरी
 - کشمیری
 - Кыргызча
 - മലയാളം
 - मराठी

Character Set (Exam)

- A character set is the total list of characters that are available for a character coding system.
- ASCII can code for 128 characters – they are mapped to the 128 possible 7-bit binary numbers.
- Unicode defines what characters it encodes and then uses a suitable number of octets to store them as a number.

Activity

- Use a Spreadsheet to produce a character conversion table like this one.
- There are functions to do the conversions.

Denery	Hex	Char	Denery	Hex	Char	Denery	Hex	Char	Denery	Hex	Char
33	21	!	58	3A	:	83	53	S	108	6C	l
34	22	"	59	3B	;	84	54	T	109	6D	m
35	23	#	60	3C	<	85	55	U	110	6E	n
36	24	\$	61	3D	=	86	56	V	111	6F	o
37	25	%	62	3E	>	87	57	W	112	70	p
38	26	&	63	3F	?	88	58	X	113	71	q
39	27	'	64	40	@	89	59	Y	114	72	r
40	28	(65	41	A	90	5A	Z	115	73	s
41	29)	66	42	B	91	5B	[116	74	t
42	2A	*	67	43	C	92	5C	\	117	75	u
43	2B	+	68	44	D	93	5D]	118	76	v
44	2C	,	69	45	E	94	5E	^	119	77	w
45	2D	-	70	46	F	95	5F	_	120	78	x
46	2E	.	71	47	G	96	60	`	121	79	y
47	2F	/	72	48	H	97	61	a	122	7A	z
48	30	0	73	49	I	98	62	b	123	7B	{
49	31	1	74	4A	J	99	63	c	124	7C	
50	32	2	75	4B	K	100	64	d	125	7D	}
51	33	3	76	4C	L	101	65	e	126	7E	-
52	34	4	77	4D	M	102	66	f			
53	35	5	78	4E	N	103	67	g			
54	36	6	79	4F	O	104	68	h			
55	37	7	80	50	P	105	69	i			
56	38	8	81	51	Q	106	6A	j			
57	39	9	82	52	R	107	6B	k			

Activity

Learning Intentions Covered:

Know: What character set is

Understand the use of binary codes to represent characters

Understand the relationship between the number of bits per character in a character set

Understand the difference between ASCII, extended ASCII and Unicode.

In your **Data Representation** folder in your area

Open **Year 10 Data Representation Booklet**

Complete:

➤ **All task in Part 9 - Representation of Characters**

Section 10 – lamge

Image Representation



Today's learning Intentions

- **Know:** how an image is represented as a series of pixels represented in binary
- **Understand** what Metadata included in the file
- **Understand** the effect of colour depth and resolution on the size of an image file.

Keywords

Pixels

Bit-Map

Metadata

Resolution

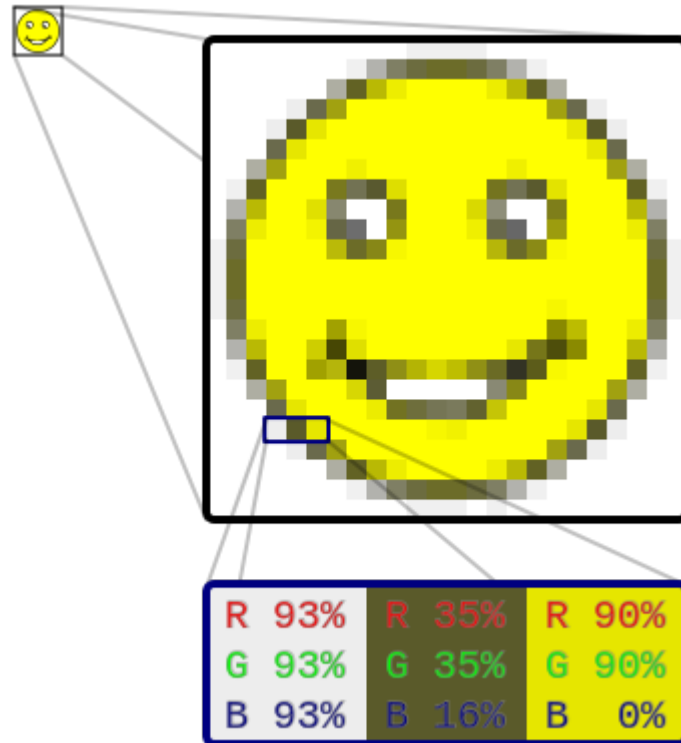
Bit Map Images

Bit maps

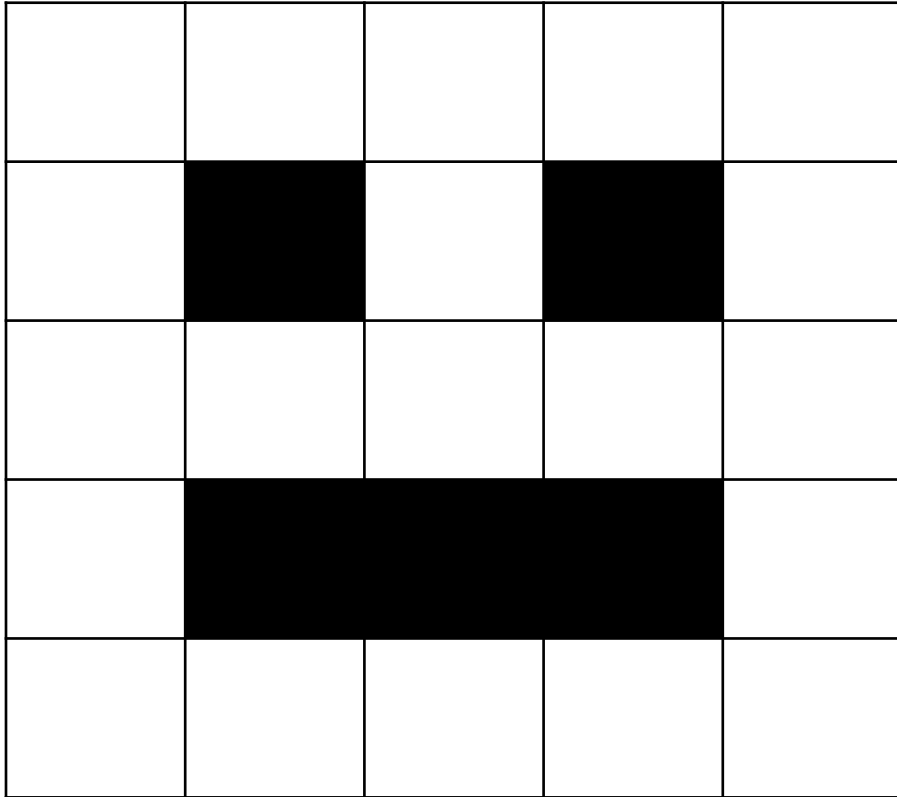
- Binary data can represent images.
- Pictures must be shown on a screen or a printer as a series of dots or **pixels**.
- They may also be stored as a series of bits. Each bit corresponds to a part of the image.
- Images stored in this way are called **bit-mapped image** files.
- The more dots in a given area, the better the quality.
- This is called the **resolution** of the picture.

Images

- So far we have considered how basic data is stored on a computer.
- Images are a little more complicated to store.
- In order to understand how images are stored in binary it is important to remember that images use pixels.



Binary Images



A image is essentially a grid of pixels. Each pixel will represent a colour.

To begin with we will only consider 2 colours. Black and white.

White can be represented as a 0 and black can be represented as 1

Binary Scan Line

0	0	0	0	0
0	1	0	1	0
0	0	0	0	0
0	1	1	1	0
0	0	0	0	0

Images are stored in scan lines. Each line is encoded from left to right, top to bottom.

The image here would get the following binary values.

00000
01010
01110
00000

Binary Activity

Draw a image by
changing the
background colour.

Then write down
what the binary will
be below –

Binary 4 colours

00	00	00	00	00
00	10	00	10	00
00	00	11	00	00
00	01	01	01	00
00	00	00	00	00

- We call this representation a bit-plane.
- It is possible to combine bit planes to produce more colours.
- Each bit-plane will double the number of available colours.

00 – white

01 – black

10 – red

11 - yellow

Binary 8 colours

111	111	111	111	111
110	010	101	010	110
110	100	011	100	110
110	001	001	001	110
111	111	111	111	111

This pattern can continue.

000 – white

001 – black

010 – red

011 - yellow

100 – blue

101 – grey

110 – purple



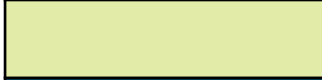

111 - green

Binary Activity

Create an image using 3 bit planes. Present it exactly the same as slide 7.

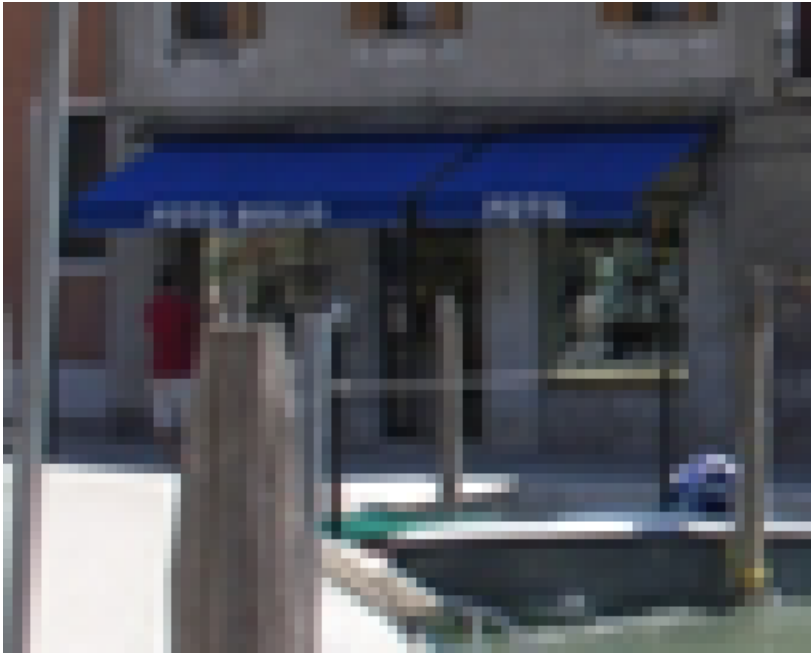
Binary *RGB*

- Colour on computers is made up of three main components.
- Red, Green and blue.
- By combining these colours in different proportions you can make any colour.
- Each of the channels can have a value from 0 to 255.
- The table below shows some example colours.

	Red	Green	Blue
	211	57	134
	113	113	113
	221	232	152
	10	60	80

Bit Map Images

- When a **bit-mapped image** is enlarged, the **pixels** enlarge too.
- This produces a jagged, blurred image.



Resolution vs File Size

- The higher the **resolution**, the bigger the file size.



High resolution: **1288 kB**



Low resolution: **67 kB**

Bits vs Quality

- **Colour depth** or bit depth is the number of bits used to represent the colour of a single pixel, in other words, **bits per pixel**.
- The more bits used, the better the quality of the colour because more colours can be **encoded**.



24 bits per pixel



3 bits per pixel

Metadata

- **Metadata** is data about data.
- Metadata is useful in many types of file to provide extra information.
- Image files may contain metadata about:
 - the height;
 - the width;
 - the colour depth;
 - the resolution;
 - camera used;
 - exposure details;
 - when the image was created;
 - who owns the copyright;
 - contact information.

Metadata

The **metadata** for an image:

Main JPG image displayed here at 16% width ($\frac{1}{25}$ the area of the original)



Basic Image Information

Camera:	Olympus SP700
Lens:	6.3 mm (Max aperture f/3.3)
Exposure:	Auto exposure, Program AE, $\frac{1}{500}$ sec, f/3.3, ISO 64
Flash:	Auto, Did not fire
Date:	July 28, 2010 12:00:22PM (timezone not specified) (2 months, 10 days, 15 hours, 57 minutes, 34 seconds ago, assuming an image timezone of US Pacific)
File:	2,112 × 2,816 JPEG (5.9 megapixels) 1,318,237 bytes (1.3 megabytes) Image compression: 93%
Color Encoding:	WARNING: Color space tagged as sRGB, without an embedded color profile. Windows and Mac web browsers will treat the colors randomly. Images for the web are most widely viewable when in the sRGB color space and with an embedded color profile. See my Introduction to Digital-Image Color Spaces for more information.

Binary *Meta-Data*

- Images hold extra information to allow their contents to be read quickly and to allow previews.
- This is called meta-data. Information stored is –
 - File type
 - Image width
 - Image height
 - Number of bit planes (colour depth)
 - Colour table (if using less than 24 bit planes)

This information is used to read the data correctly.

Binary *Meta-Data*

0001000100010001111111111111111101010101010101010001000100010001

Unless you know what the width, height and colour depth is it will be hard to determine what the image is

0001	0001	0001	0001
1111	1111	1111	1111
0101	0101	0101	0101
0001	0001	0001	0001

Or?

000	100	010	001	000
111	111	111	111	111
110	101	010	101	010
000	100	010	001	000

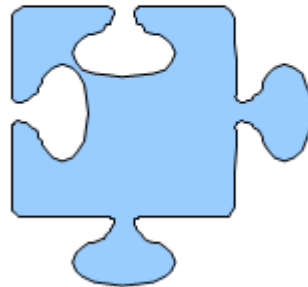
Binary *Meta-Data*

- In order to properly read the file software needs to know
 - Colour depth – how many bits represent each pixel
 - Width – how many pixels to read in for each line
 - Height – how many lines to read in
- If this information is incorrect then the image will be corrupt and not show correctly.

Vector Graphics

Vector graphics

- Pictures can also be stored as formulae.
- They define characteristics of points, lines and curves.
- They do not distort when enlarged, because the image is recalculated.



Lossy and Lossless

- It is often necessary to compress a file to make it small enough to be used - for example making a music file small enough so that enough can be stored on an iPod or website.
- There are two main possibilities:
- **Lossless**
 - These are used to make a file a smaller size but without losing any of the information.
 - Using this method you can always get back to the original file
- **Lossy**
 - Sometimes some loss of quality is acceptable.
 - For example the human ear cannot hear all frequencies, so a file format that throws away parts that people can't hear may end up with a smaller file, but it is not possible to get back to how exactly the original music sounded.

Exam Question

1. A Video Clip is compressed using a lossy compression
2. Explain why lossy compression is suitable for a video clip but not suitable for a text document

Activity

- Open the Images Representation Folder
- Complete all activities

Activity

1. Look at resolutions of scanners in online advertisements or a PC store. See if there is any relationship between dpi and the price.
2. Use image software (Photoshop or Fireworks) to reduce successively the size of a photographic image by saving it at different compression ratios.
 - At what stage does the pixilation become noticeable?
3. Draw a simple shape using the drawing tools of a word processor.
 - Enlarge it and notice how no pixilation effects take place.
4. Use an online metadata reader, such as <http://regex.info/exif.cgi>,
 - Find out information about some photographs.
5. Find out what EXIF means.

Activity

1. Look at resolutions of scanners in online advertisements or a PC store. See if there is any relationship between dpi and the price.
2. Use image software (Photoshop or Fireworks) to reduce successively the size of a photographic image by saving it at different compression ratios.
 - At what stage does the pixilation become noticeable?
3. Draw a simple shape using the drawing tools of a word processor.
 - Enlarge it and notice how no pixilation effects take place.
4. Use an online metadata reader, such as <http://regex.info/exif.cgi>,
 - Find out information about some photographs.
5. Find out what EXIF means.

Activity

Learning Intentions Covered:

Know: how an image is represented as a series of pixels represented in binary

Understand what Metadata included in the file

Understand the effect of colour depth and resolution on the size of an image file.

In your **Data Representation** folder in your area

Open **Year 10 Data Representation Booklet**

Complete:

➤ **All task in Part 10 - Representation of Images**

Section 11 – Sound

Image Representation



Today's learning Intentions

- **Know:** how sound can be sampled and stored in digital form
- **Understand** how sampling intervals and other factors affect the size of a sound file and the quality
- **Understand** sample size, bit rate and sampling frequency.

Keywords

Bit rate

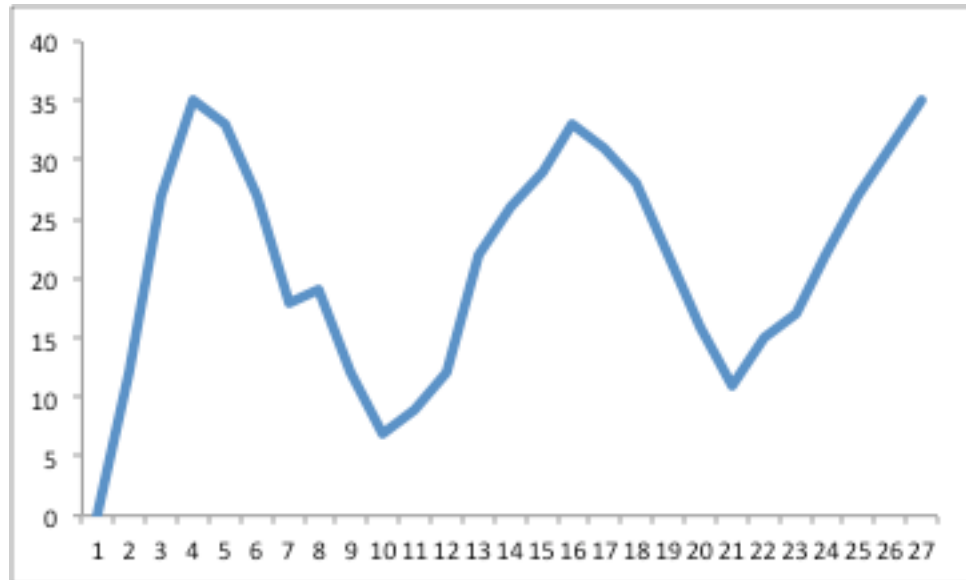
sampling

digital

frequency

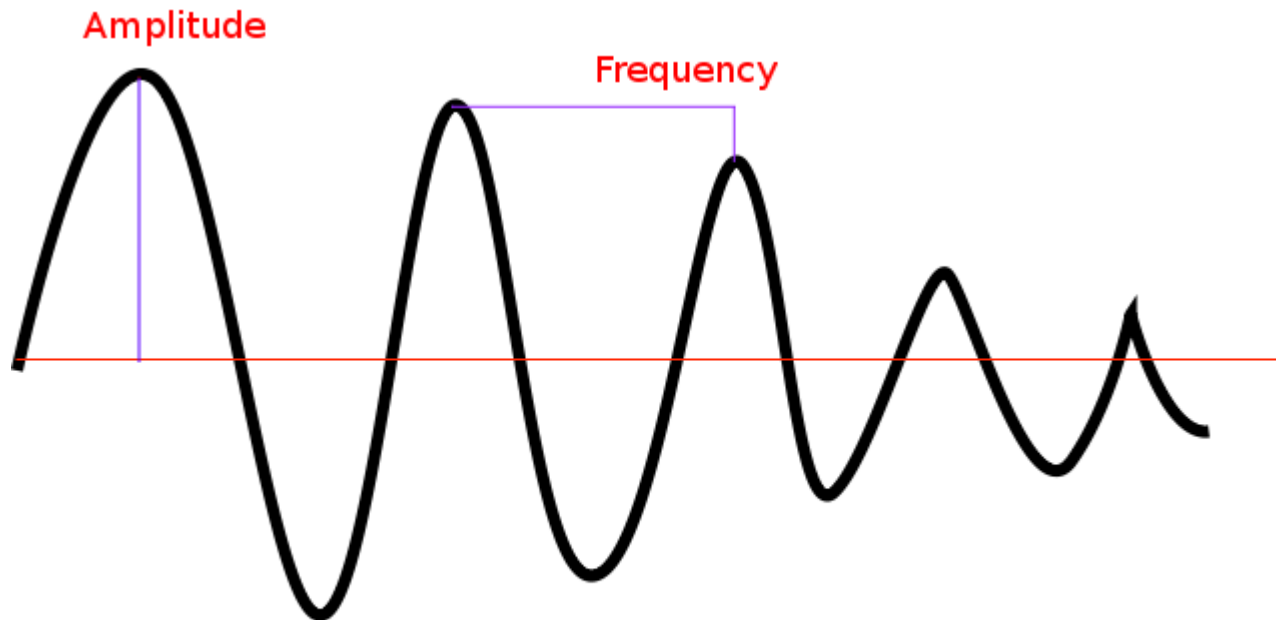
Sound Waves

- Sound is an **analogue** energy form.
- It can take a range of values:
 - for **loudness**;
 - for **frequency**.



Sound Waves

- Sound travels as a wave.
- The **amplitude** of the wave controls how loud the sound is.
- The **frequency** controls the pitch.



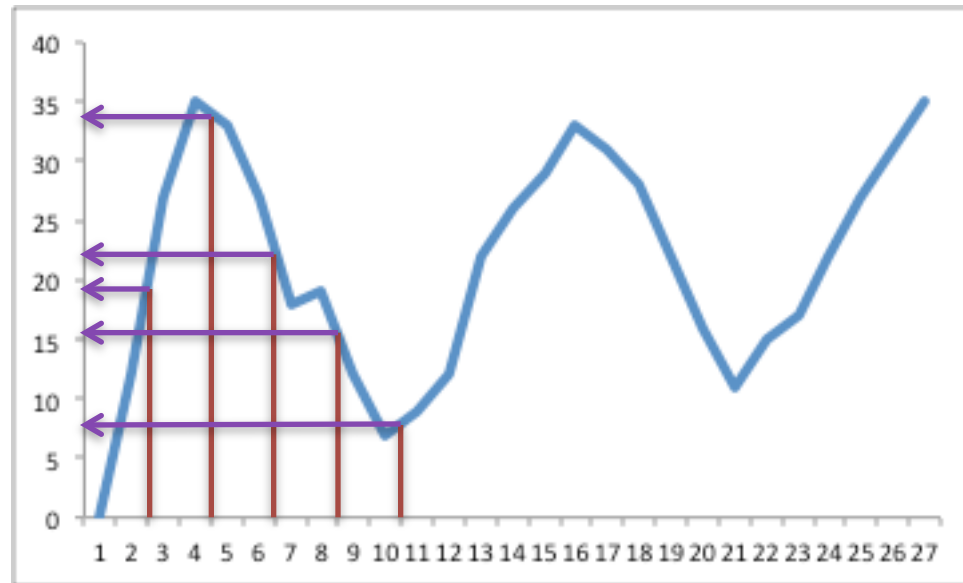
Video on Sample Rate



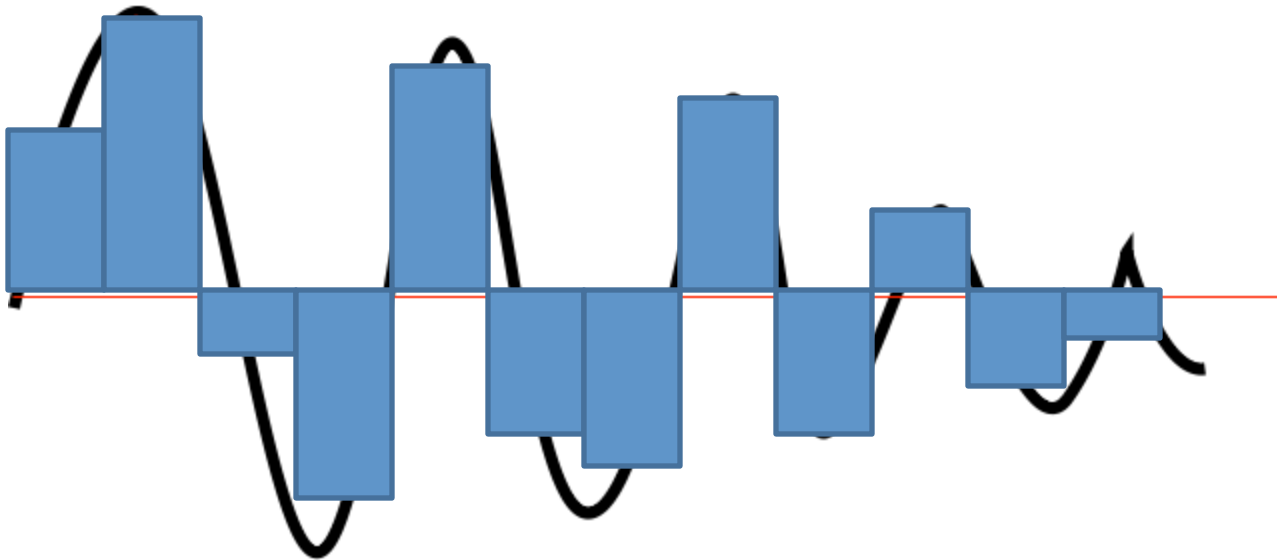
www.digitaljuice.com

Sound Sample

- Computers only work with digital signals, so the sound is **sampled**.
 - Measurements are taken at intervals.
 - Various **parameters** are measured.

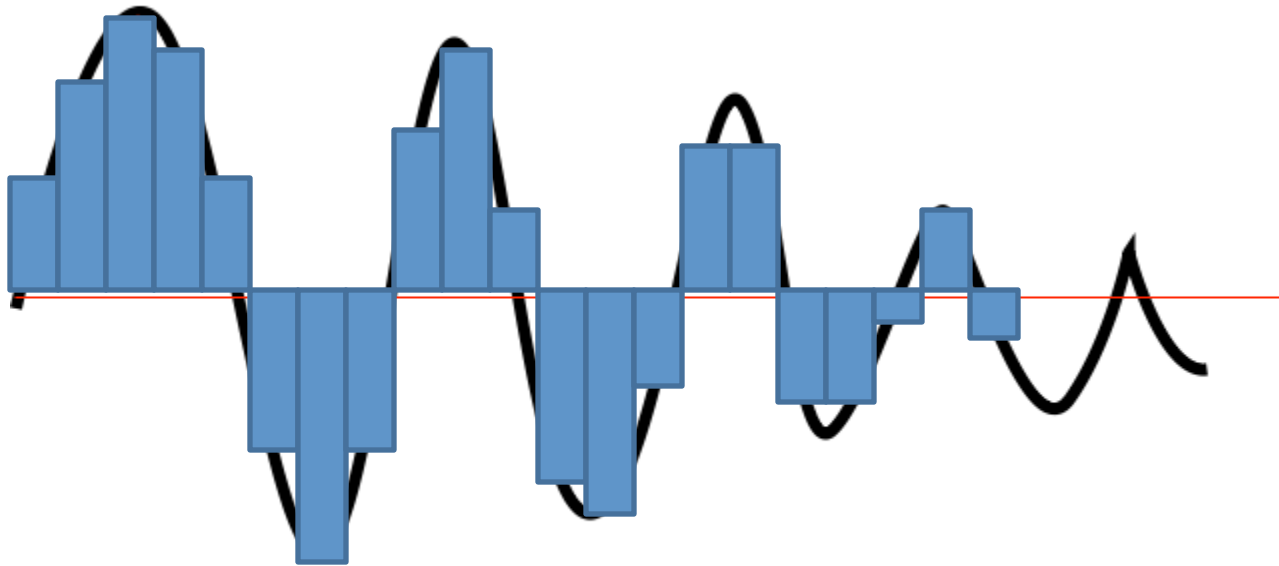


Binary *Sound Waves*



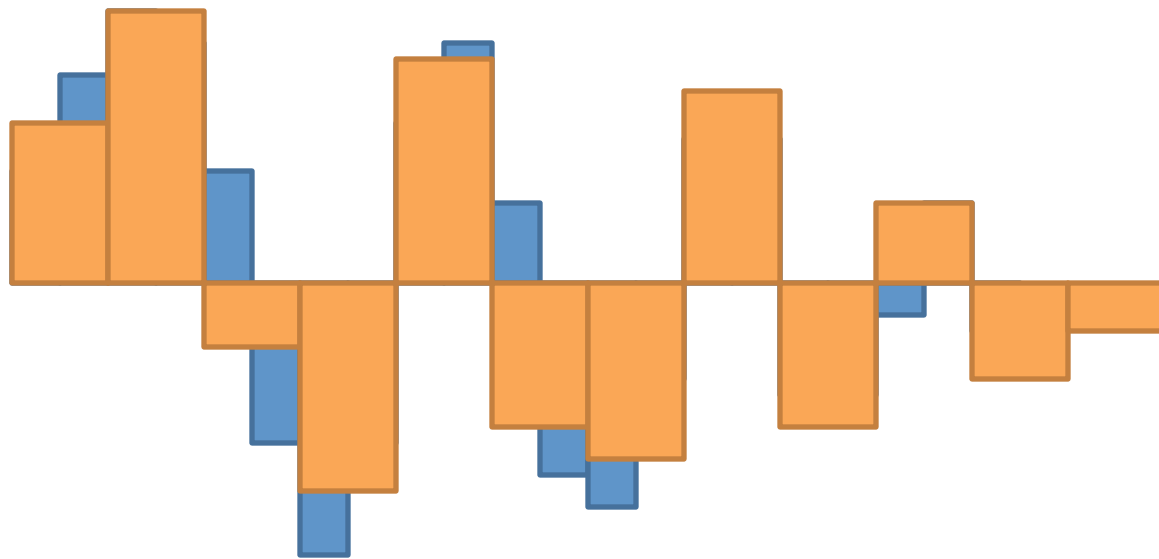
- Digital sound will sample the sound wave at certain points.
- These are represented by the bars.
- After sampling we are left with a digital version of the sound.

Higher Sampling Rate



- By doubling the sample rate we can represent the original sound wave much more closely.
- Increasing the sample rate will improve the quality of the sound file.

Higher Sampling Rate



- The lower sample rate will create a crackly sound.
- Click on the speakers to hear the difference.

Sound Quality

- These readings give us an approximation to the sound.
 - The more measurements, the better that approximation and quality of the recording.
 - More measurements, however, means a larger file size.

Time	Sampled value
2	18
4	34
6	22
...	...



Audio CD Sample Rate

- **Audio CD**
 - music is sampled 44,100 times per second
 - the samples are 2 bytes (16 bits) long
 - $44,100 \text{ samples/second} * 16 \text{ bits/sample} * 2 \text{ channels} = 1,411,200 \text{ bits per second}$
- Audio CDs store a LOT of data.
- This is much too much for normal computer storage and downloads.

Sound Sample Rate

Sampling

- The rate of sampling can be varied.
- This is the **bit rate**:
 - The number of bits sampled in a given time.
- The higher the bit rate, the better the recording:
 - but the bigger the file size.

Sound Compression

Compression

- Sound files are usually compressed for computer storage and transmission.
- **MP3** is a common compressed sound file format.
- **Algorithms** are devised to remove the parts of the data that humans are least likely to hear.

Quality Vs File Size

- Each sample for a digital file takes up space. The more samples you have the higher the file size.
 - 8000 Mhz file size was 2.1 MB
 - 44100 Mhz file size was 11.5 MB
- The difference between file size and quality is very closely related.
- Ever noticed that sound on internet sites is poor?
 - This is to help reduce the amount of time it will take to download.
- **Note** – sample rate is sometimes referred to as bit rate.

Activity

Answer the following question in your book or revision power point

1. Find out how the bit rate of a sound recorder affects the file size.
2. What is the difference between 16-bit and 24-bit audio?
3. Why is sound sampled when digitally recorded?
4. Why are sound files usually compressed?
5. Explain the relationship between file size and sample rate.

Activity 1 Answer

1. The more bits, the better the quality but the bigger the file size.
2. 24-bit audio is better quality.
3. Need to convert analogue continuous data to digital.
4. They are large files that would take a long time to download and use a lot of storage space.
5. The higher the bit rate, the better the recording but the bigger the file size.

Activity 2

1. Record sound samples using the sound recorder.
2. Look at the file sizes that are produced and relate them to the quality of recording and the length of the sample.
3. Work out how long it would take to download the file at different bit rates

Activity

Learning Intentions Covered:

Know: how sound can be sampled and stored in digital form

Understand how sampling intervals and other factors affect the size of a sound file and the quality

Understand sample size, bit rate and sampling frequency.

In your **Data Representation** folder in your area

Open **Year 10 Data Representation Booklet**

Complete:

➤ **All task in Part 11 - Representation of Sounds**

Plenary - Peer/Self Assessment

Complete all questions on the board

- 1. Use the mark scheme to mark your work**
 - Copy the correct answers where you got it wrong**