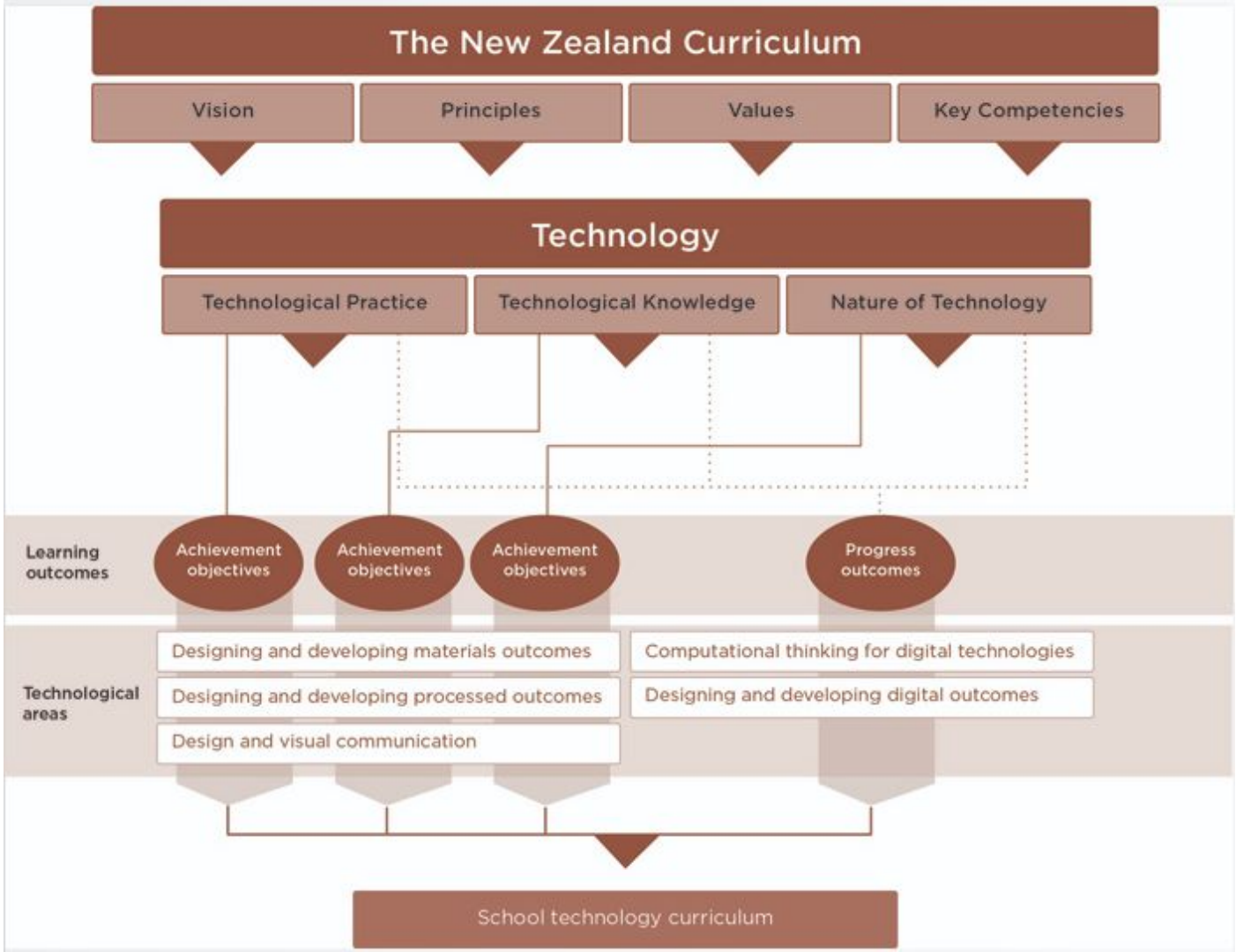
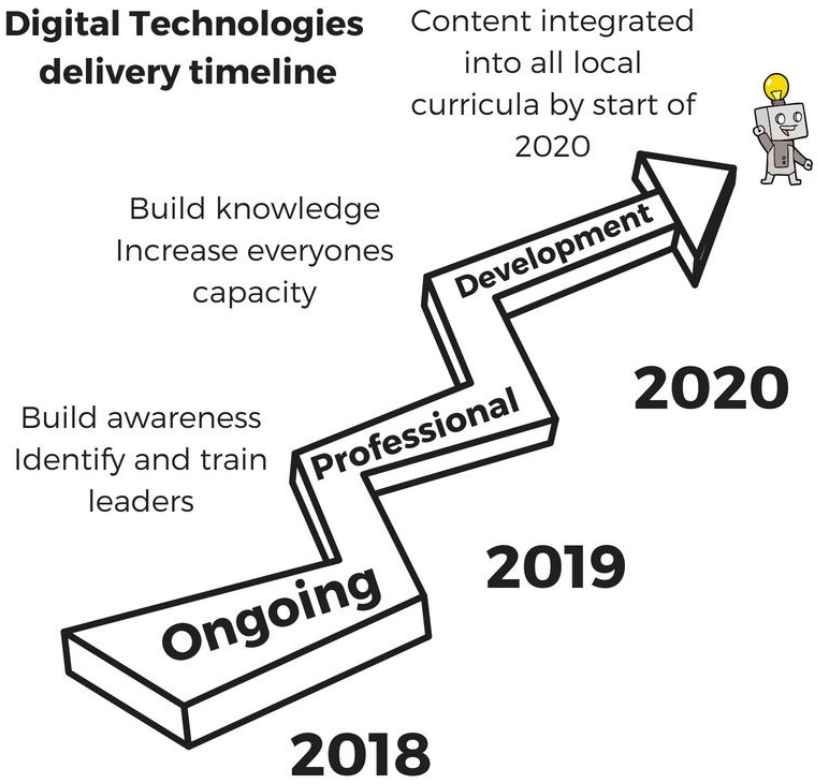
MATHEMATICS & THE RAINBOW

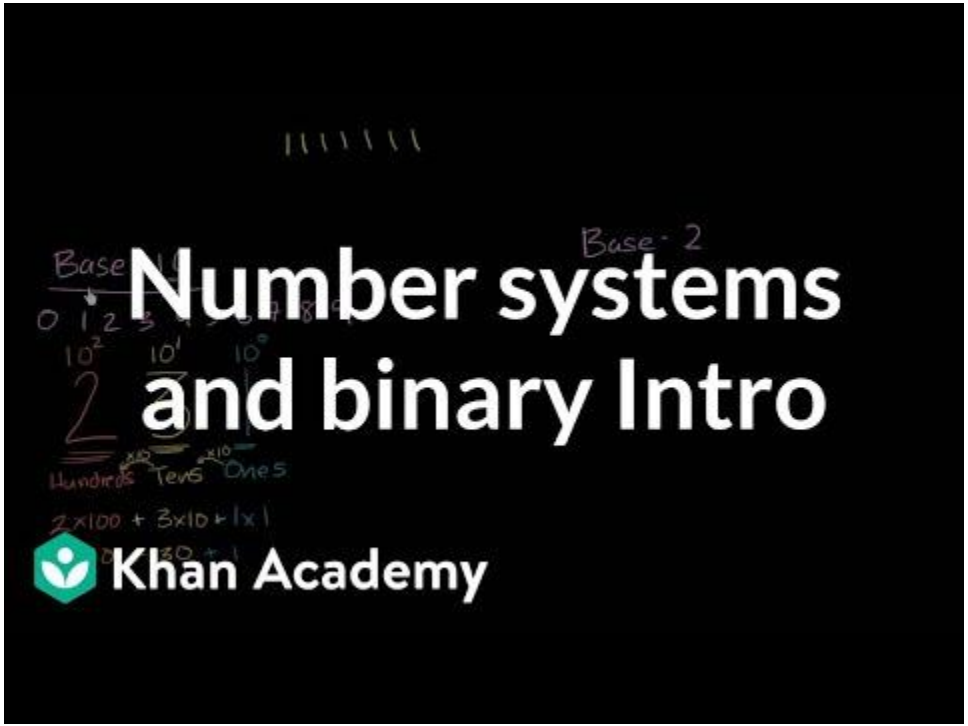

The New Zealand Curriculum
Vision
Principles
Values
Key Competencies
Technology
Technological Practice
Technological Knowledge
Nature of Technology
Learning outcomes
Achievement objectives
Achievement objectives
Achievement objectives
Progress outcomes
Technological areas
Designing and developing materials outcomes
Designing and developing processed outcomes
Design and visual communication
Computational thinking for digital technologies
Designing and developing digital outcomes
School technology curriculum

Delivery timelines

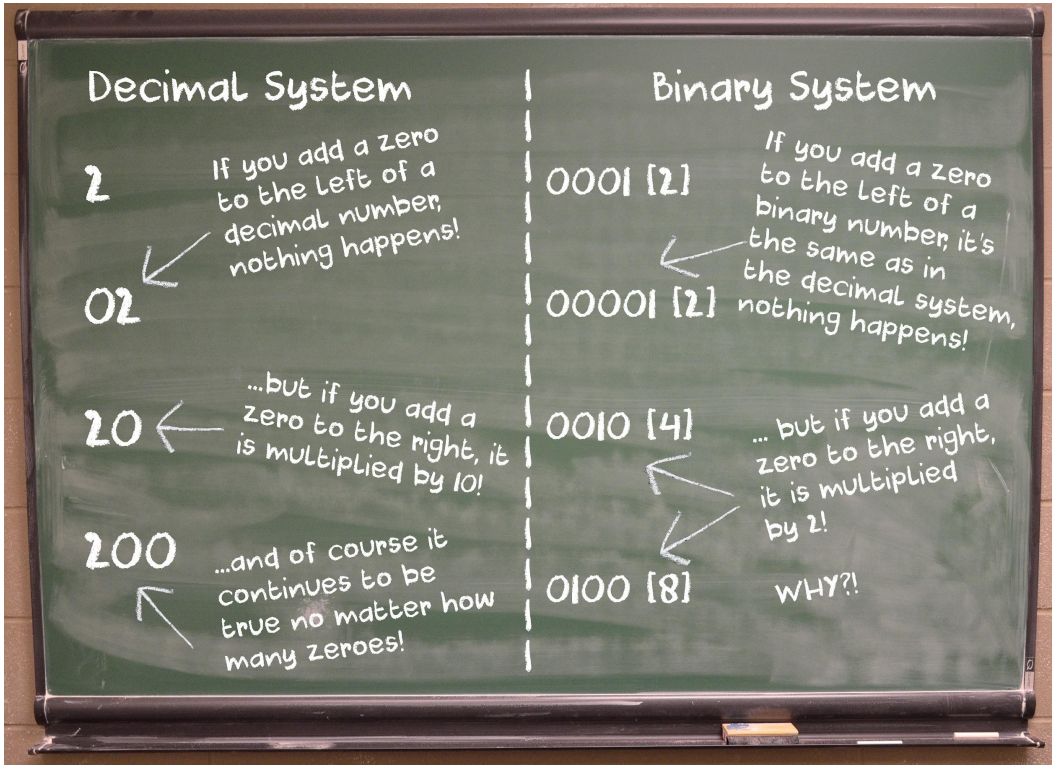


		COMPUTATIONAL THINKING						DESIGNING & DEVELOPING DIGITAL OUTCOMES				
Curriculum Level	Year	PO Level	Algorithms	Data Representation	Computational Thinking	Things to do?	Year	PO Level	Digital Applications	Digital Infrastructure	Things to do
1	0	1	Non computerised step by step instructions for a simple task		Algorithmic thinking Logical Thinking Decomposition Simple Debugging - What went wrong & how would you fix it?	CS Unplugged Hello Ruby code.org Scratch Jnr	0	1	Identifying digital applications Awareness of purpose of applications Understanding Humans make applications	Understanding basic storage & retrieval Understand basics of computer system Inputs & Outputs of a system	Hello Ruby CS Unplugged IOT exercise
	1						1				
	2						2				
2	3	2	Computerised step by step instructions (Program) for a simple task		What is an algorithm What is a program Inputs Sequences Outputs Simple Debugging	Scratch Jnr CS Unplugged Kiwi Code Hello Ruby Hour of code code.org	3				Unmaking Music Film making Animation Makey Makey
	4						4	2	Understanding how applications change over time Understanding techs impact on society Use more advanced applications & Files types	Understanding components of a computer system & how it works together Understand human's role in system Intellectual property	Robotics 3D Printing AR/VR Tech for good Unmaking & Remaking Wearables - Lily Pads Microbit Aurdino Rasbry Pi
3	5						5				
	6	3	Create a basic algorithm & program on their own to solve a slightly complex task including iteration	Understand Binary - two states	Building on outputs and inputs Evaluate code understanding there can be more then 1 solution Error Detection Loops & Iteration Logical Thinking - Prediction	Hour of code Code Avengers code.org Scratch CS Unplugged code.org	6				
4	7						7				
	8	4	Create a simple program using all CT concepts used so far	Error Detection in data transmission Understand amount of data and searching & sorting a computer does & how it does it	Evaluate code efficiency, elegance and eloquence Selection - If statements Comparative operators Variables UX- Efficiency & usability	Scratch gamefroot Codeavengers MinecraftEDU CS Unplugged	8	3	Select appropriate software type for tasks & use it to make the required outcome Design, develop, store, test, & evaluate content	Understand operating systems, and security Understand file conventions and file management procedures Understand storage Privacy & Security Social, ethical, end user concerns	
5	9						9				
	10	5	Create a complex program - multiple algorithms - uses all CT concepts used so far Understand and employ heuristics	How complex types of data is stored	Variables Logical Operators When to use control structures Testing Functions Parameters	Hex & bitmap Construct3 Scratch gamefroot Codeavengers MinecraftEDU Swift Play ground Text based programming	10				

INTRO TO NUMBER SYSTEMS
(KHAN ACADEMY)



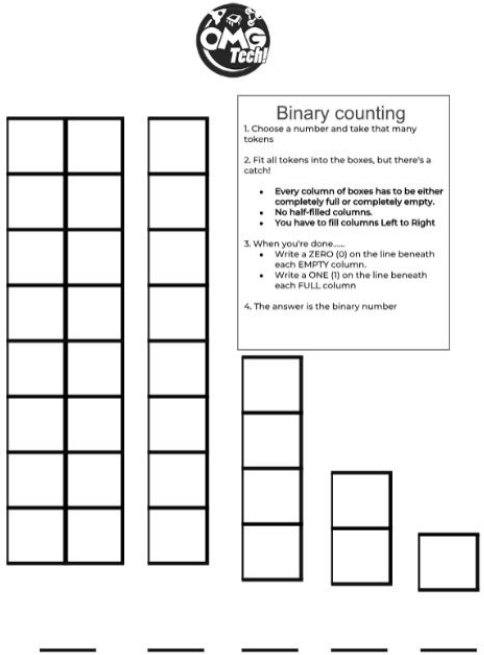
BINARY VS DECIMAL



BINARY ACTIVITY

OFFLINE!

BINARY WORKSHEET



BINARY WORKSHEET

Fill out all the numbers in Binary...

A	B	C
1: _ _ _ _ _	2: _ _ _ _ _	3: _ _ _ _ _
4: _ _ _ _ _	5: _ _ _ _ _	6: _ _ _ _ _
7: _ _ _ _ _	8: _ _ _ _ _	9: _ _ _ _ _
10: _ _ _ _ _	11: _ _ _ _ _	12: _ _ _ _ _
13: _ _ _ _ _	14: _ _ _ _ _	15: _ _ _ _ _
16: _ _ _ _ _	17: _ _ _ _ _	18: _ _ _ _ _
19: _ _ _ _ _	20: _ _ _ _ _	21: _ _ _ _ _
22: _ _ _ _ _	23: _ _ _ _ _	24: _ _ _ _ _
25: _ _ _ _ _	26: _ _ _ _ _	27: _ _ _ _ _
28: _ _ _ _ _	29: _ _ _ _ _	30: _ _ _ _ _
31: _ _ _ _ _	32: _ _ _ _ _	33: _ _ _ _ _

EXTRAS FOR EXPERTS:
Can you guess what 32 & 33 would be? What about 156?

How many squares would be in the 6th column?

What is the largest column that we use to count in binary?

Can you write your birthday in binary?
Can you write your name?
INTERESTING FACT: Although we have been writing "0's and 1's", our computer does not store 0's and 1's; it has no way of doing this! It uses physical mechanisms such as high and low voltage ie 1= High voltage 0 = low voltage

INTRODUCING HEXADECIMAL

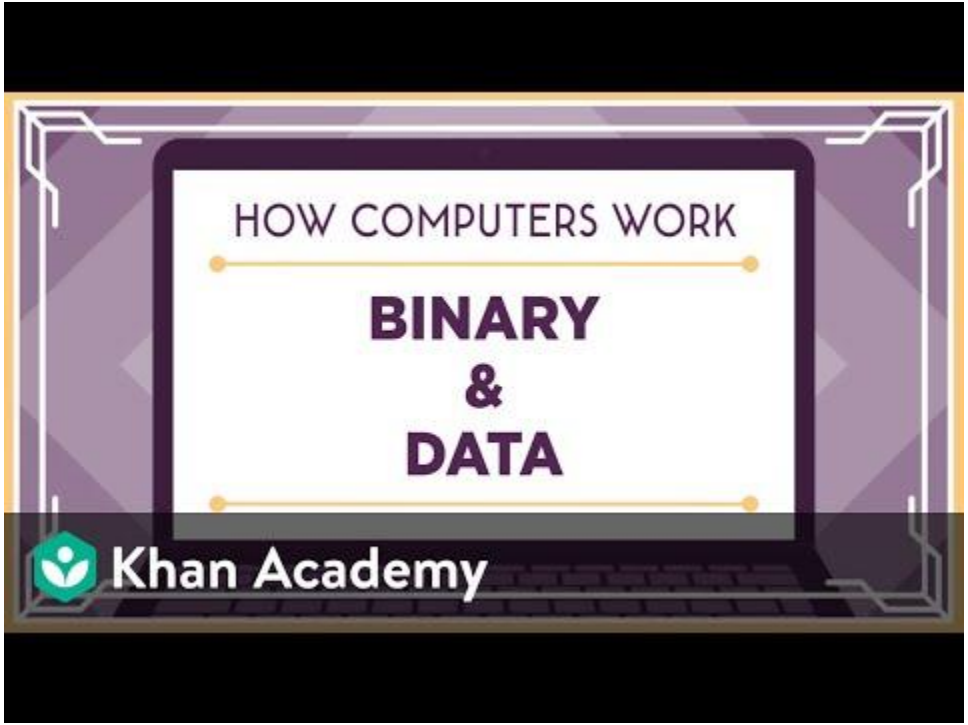
EXAMPLE:

	NUMBER 74								
		128	64	32	16	8	4	2	1
STEP 1		0	1	0	0	1	0	1	0
		8	4	2	1	8	4	2	1
STEP 2	Binary	0	1	0	0	1	0	1	0
STEP 3	Decimal	4				10			
STEP 4	Hex	4				A			

Binary	Hexadecimal
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	A
1011	B
1100	C
1101	D
1110	E
1111	F[5]

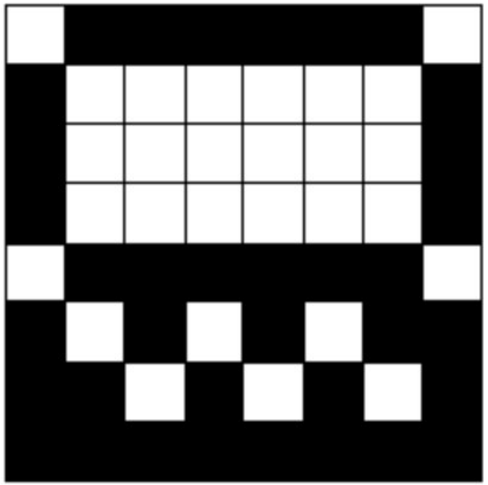
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

BINARY & DATA
(KHAN ACADEMY)



DATA REPRESENTATION

IMAGES



BINARY	HEX
01111110	7E
10000001	81
10000001	81
10000001	81
01111110	7E
10101011	AB
11010101	D5
11111111	FF

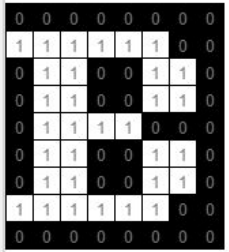
DATA REPRESENTATION

IMAGES (1-BIT)

Khan Academy

Binary Graphics 1-bit

```
// TASK: Change the letter 'B' to an 'A'

// In 1-bit graphics 0=black, 1=white
// (or arbitrary colour 1 & arbitrary colour 2)
var binaryImage = [
    '00000000',
    '11111100',
    '01100110',
    '01100110',
    '01111000',
    '01100110',
    '01100110',
    '11111100',
    '00000000'
    ];
var pixel = 20;

for (var row = 0; row < binaryImage.length; row += 1) {
    var thisRow = binaryImage[row];
    for (var col = 0; col < thisRow.length; col += 1) {
        var thisDigit = binaryImage[row][col];
```



Spin-off

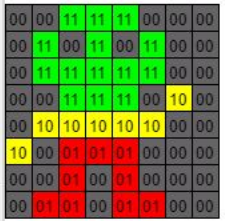
DATA REPRESENTATION

IMAGES (2-BIT)

Khan Academy

Binary Graphics 2-bit

```
// TASK: Make repton's eyes RED

// In 2-bit graphics 00=black, 01=red, 10=yellow, 11=green
var raw = [
    '0000111111000000',
    '0011001100110000',
    '0011111111110000',
    '0000111111001000',
    '0010101010100000',
    '1000010101000000',
    '0000010001000000',
    '0001010001010000'
    ];
var pixel = 20;
fill (0, 0, 0);

for (var row = 0; row < raw.length; row += 1) {
    var thisRow = raw[row];
    for (var col = 0; col < thisRow.length; col += 2) { // step 2
        var thisDigit = raw[row][col] + raw[row][col+1];
```



Spin-off

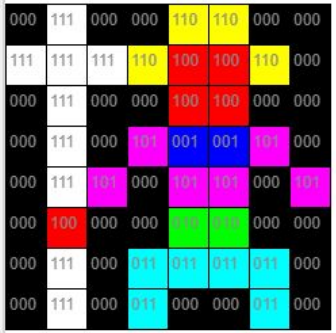
DATA REPRESENTATION

IMAGES (3-BIT)

Khan Academy

Binary Graphics 3-bit

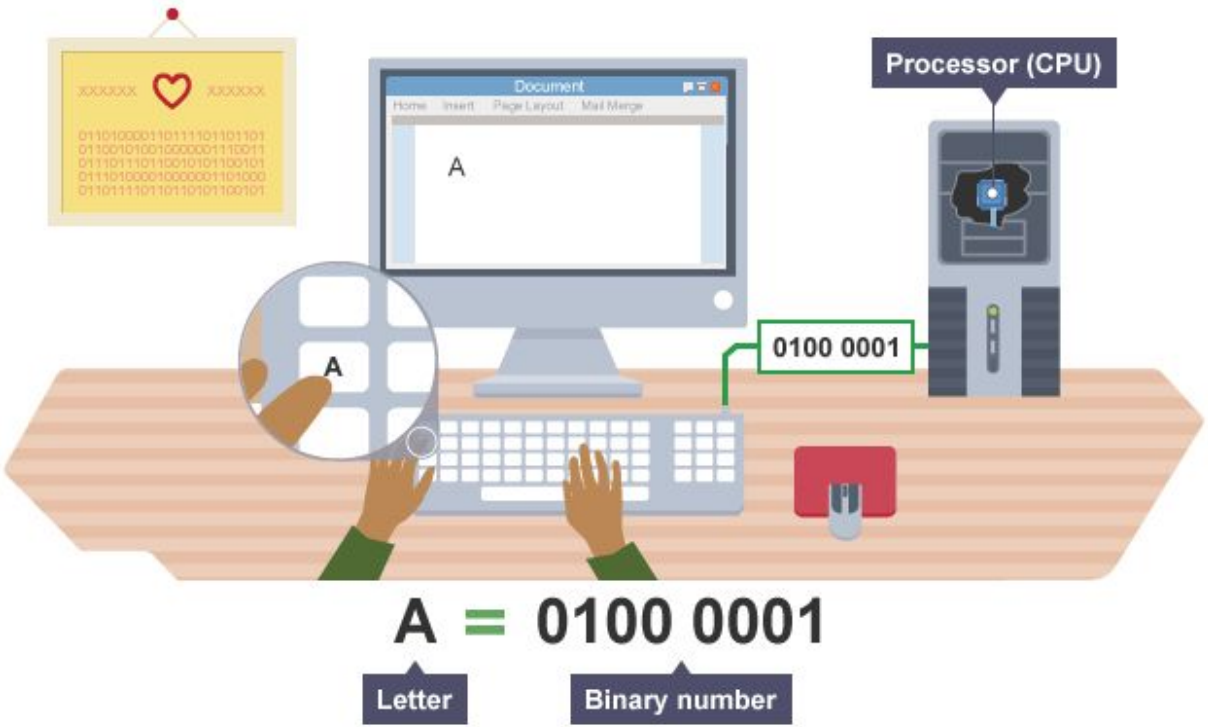
```
// TASK: Change the man's trousers to red.

// In 3-bit graphics you can map each bit to RGB
// 000 --- (black: no red, no green, no blue)
// 001 --B (blue),  010 -G- (green),   011 -GB (cyan)
// 100 R-- (red),   101 R-B (magenta), 110 RG- (yellow),
// 111 RGB (white)
var rawBinary = [ // spaces are just for human readability
    '000 111 000 000 110 110 000 000',
    '111 111 111 110 100 100 110 000',
    '000 111 000 000 100 100 000 000',
    '000 111 000 101 001 001 101 000',
    '000 111 101 000 101 101 000 101',
    '000 100 000 000 010 010 000 000',
    '000 111 000 011 011 011 011 000',
    '000 111 000 011 000 000 011 000'
    ];
var pixel = 30;
fill (0, 0, 0);

for (var row = 0; row < rawBinary.length; row += 1) {
```



Spin-off

DATA REPRESENTATION

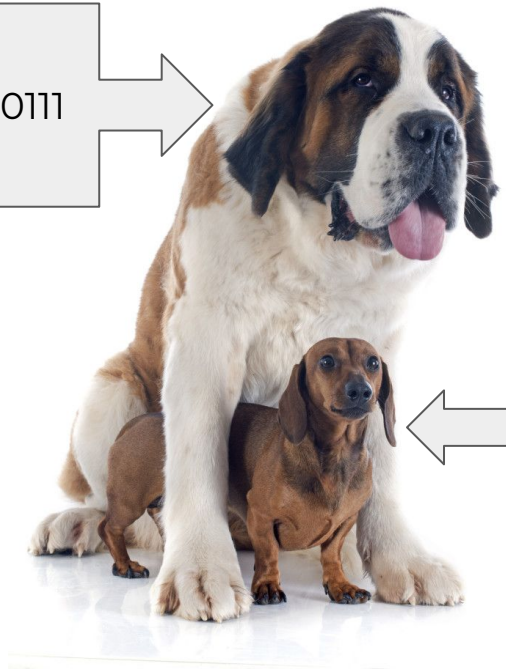
TEXT



DATA REPRESENTATION

TEXT

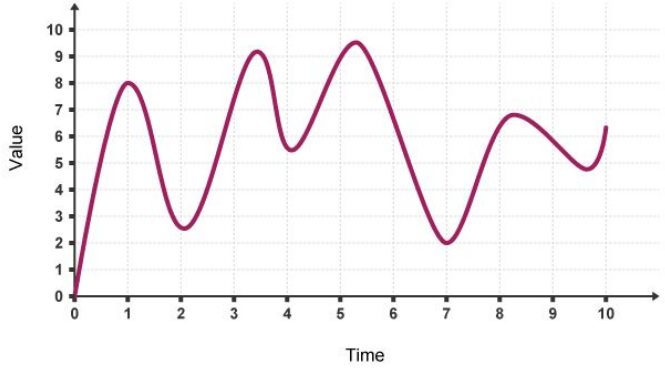
DOG = 0100 0100 0100 1111 0100 0111

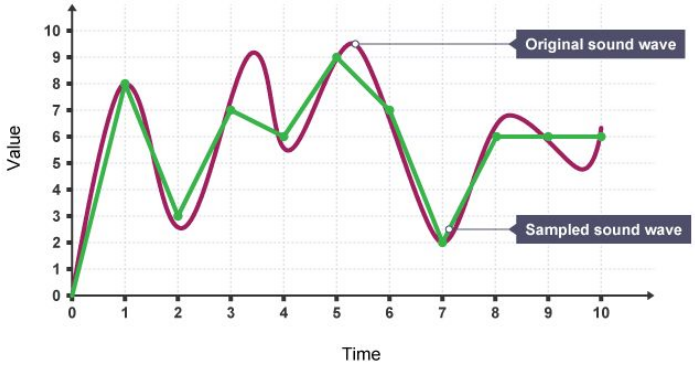


dog = 0110 0100 0110 1111 0110 0111

DATA REPRESENTATION

SOUND



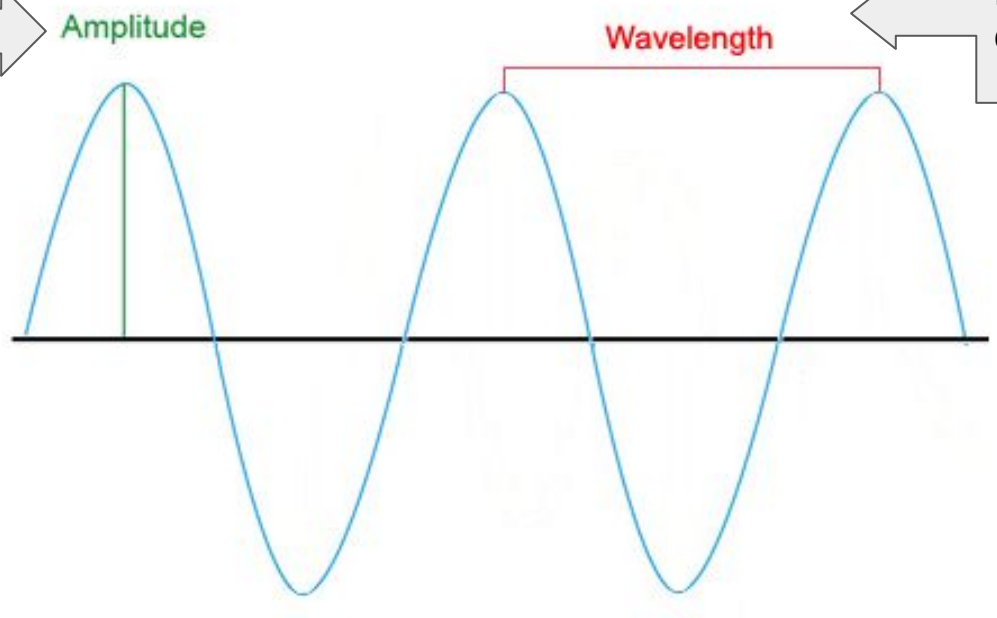


Time sample	1	2	3	4	5	6	7	8	9	10
Denary	8	3	7	6	9	7	2	6	6	6
Binary	1000	0011	0111	0110	1001	0111	0010	0100	0110	0110

DATA REPRESENTATION

SOUND

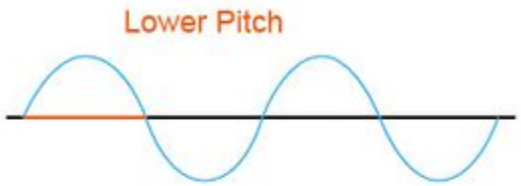
Controls how ‘loud’



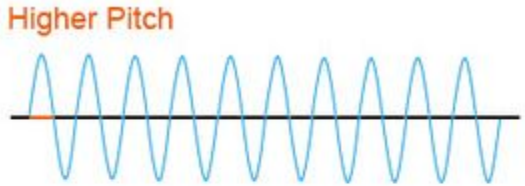
Frequency controls the pitch

DATA REPRESENTATION

SOUND

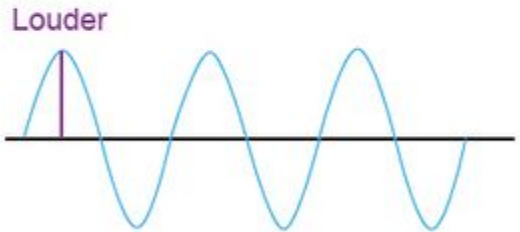


Low Frequency



High Frequency

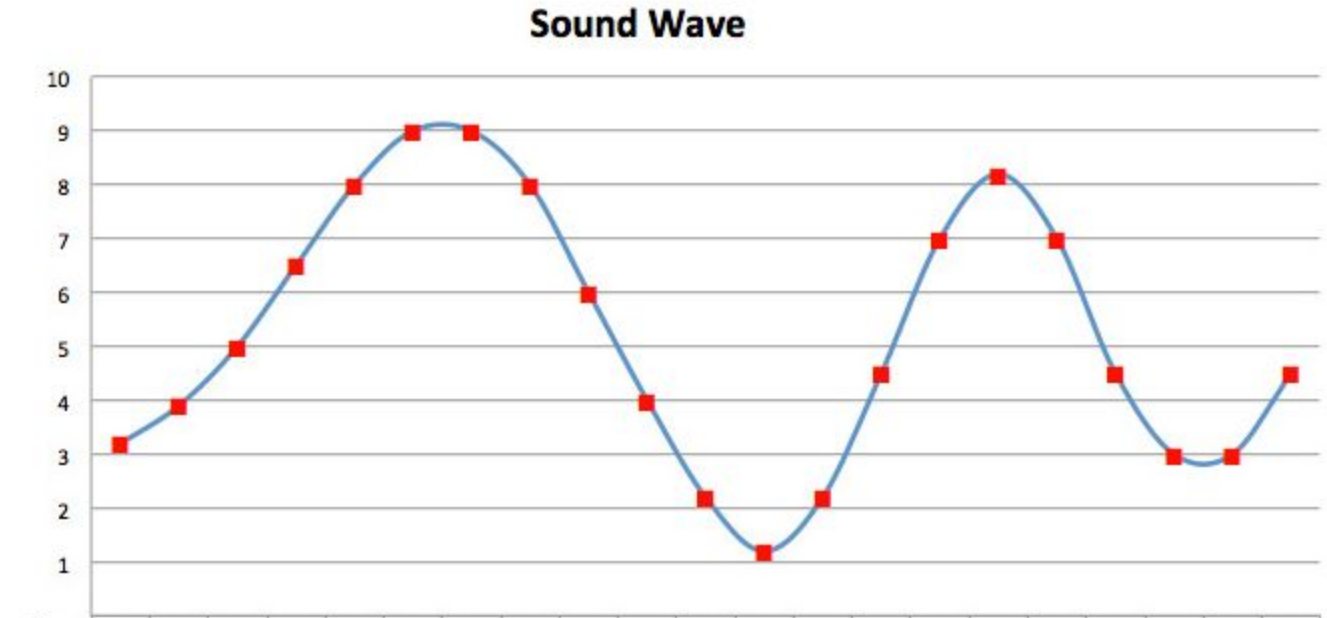
Low Amplitude



High Amplitude

DATA REPRESENTATION

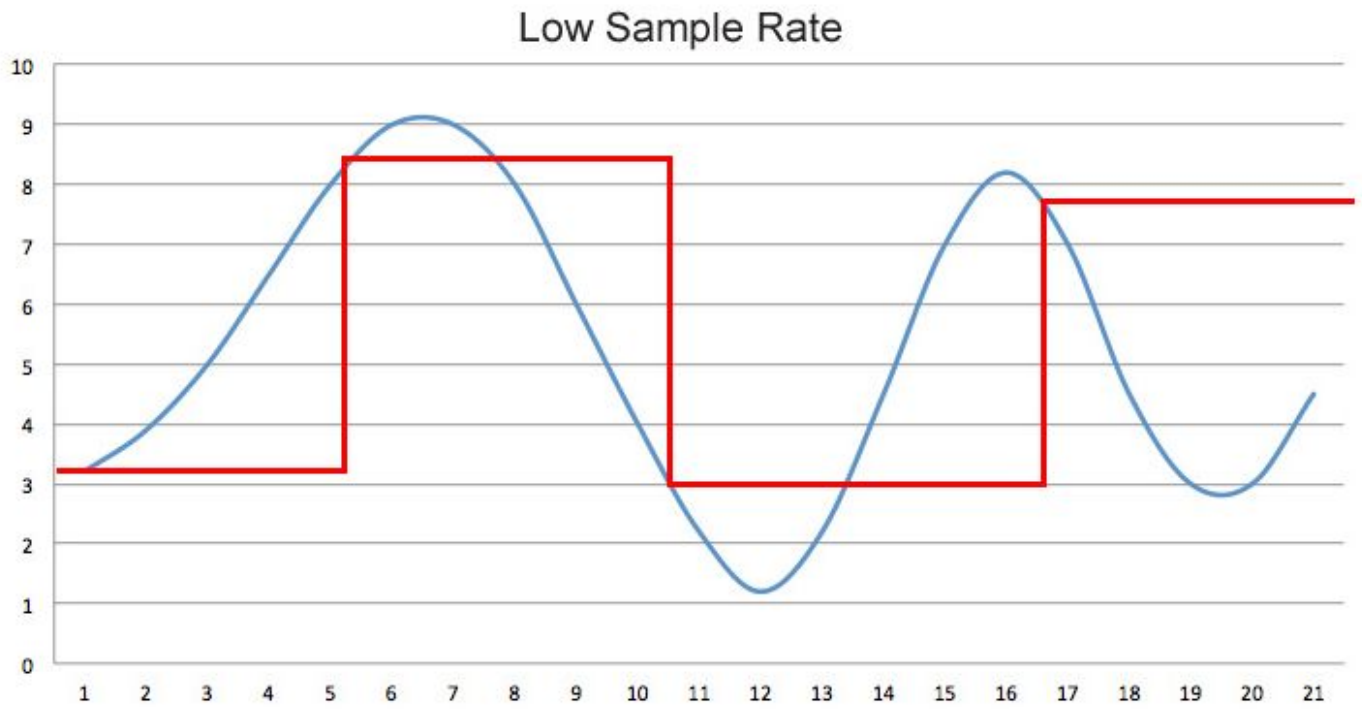
SOUND



Time	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
Sample	3.2	3.9	5	6.5	8	9	9	8	6	4	2.2	1.2	2.2	4.5	7	8.2	7	4.5	3	3	4.5

DATA REPRESENTATION

SOUND



DATA REPRESENTATION

SOUND