OCR

GCSE Computer Science

COMP2 REVISION WORKBOOK

NAME:

CLASS:

Table of Contents

My CS: HUB

Unit 1: Data & Logic

Paper 2: Computational Thinking, Algorithms & Programming

This section covers:

- 2.5 Computational Logic
- 2.6 Data Representation

U1L1: Units

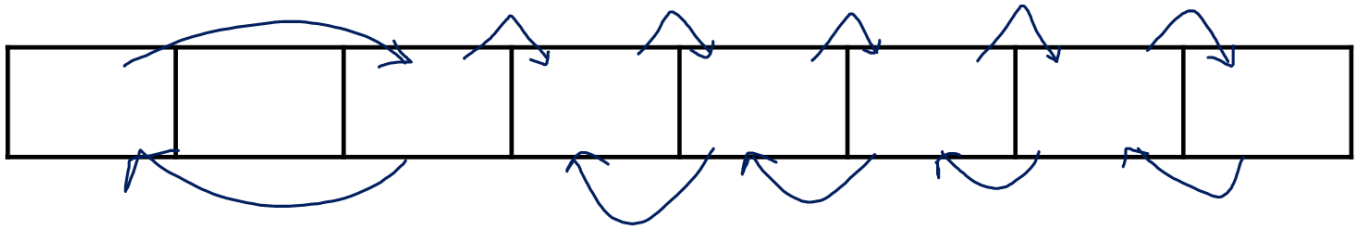
What is the smallest unit of data?

Why must data be converted into binary for a computer to process it?

Complete the following table:

Name	Size	Examples
Bit (b)		
Nibble		
Byte (B)		
Kilobyte (KB)		
Megabyte (MB)		
Gigabyte (GB)		
Terabyte (TB)		
Petabyte (PB)		

To convert between units, we can use the following diagram:



Exam Questions

1. Put these units in order of size: Gigabyte, Kilobyte, Nibble, Megabyte, Byte
2. Where does the unit 'bit' originate from?

U1L2: Binary Numbers

Converting Binary-Denary

00110101

Converting Denary-Binary

218

Adding Binary Integers

00110011 + 01010101

Binary Shifts

Shift 01001010 left two places

Shift 10110101 right three places

What is meant by 'even parity'?

What is meant by 'odd parity'?

U1L3: Hexadecimal

Binary-Hexadecimal

0101 1011

Denary-Hexadecimal

199

Hexadecimal-Binary

3E

Hexadecimal-Denary

9B

Mid-Unit Review

Unit 2.6 Review
Data – Binary

Key Terms

Parity bit –

Check digit –

Left shift –

Right shift –

Overflow –

8-bit –

Binary Addition Rules

0 + 0 =

0 + 1 =

1 + 0 =

1 + 1 =

Parity Bits

Using odd parity, fill in the check digit.

0011 001 ___

1010 101 ___

1011 110 ___

0101 000 ___

1010 100 ___

0011 001 ___

Using even parity, fill in the check digit.

0011 011 ___

1011 101 ___

1001 110 ___

1101 000 ___

1110 100 ___

0001 001 ___

Binary Addition

Add the following binary numbers:

00110011 + 01010101 =

11001111 + 00001111 =

00110011 + 00110011 =

10101100 + 00110000 =

00001111 + 11111111 =

00110000 + 10101010 =

Binary Shifts

Shift the binary numbers:

00110011 (2LS)

01010101 (1LS)

01011111 (2RS)

01010011 (1RS)

11001100 (3LS)

00110000 (3RS)

10101011 (4LS)

10101011 (3RS)

Binary Shifts (continued...)

Each binary shift left...

Each binary shift right...

01010101 (1RS)

01010101 (1LS)

10101011 (2LS)

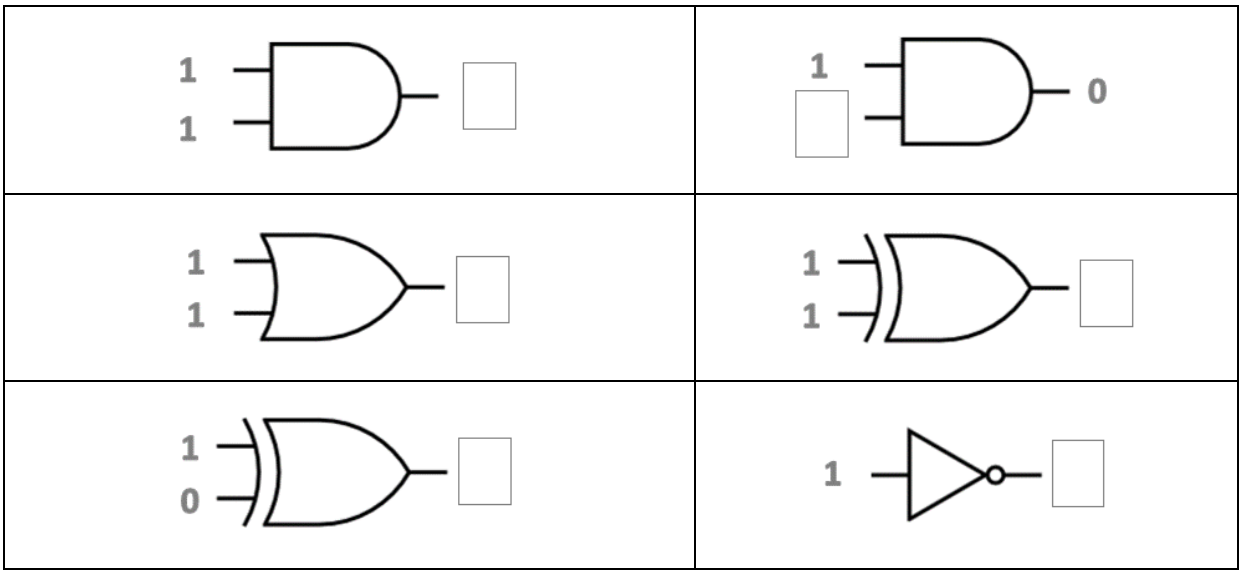
10101011 (2RS)

U1L4: Logic

AND	OR	NOT	XOR

Q = ¬A v (B ^ C)

Q = ¬(A v B) ^ C



U1L5: Characters

ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[ENG OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	|
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

What is a character?

What is a character set?

How are characters represented by a computer?

Describe the three character sets below.

ASCII

Extended-ASCII

Unicode

U1L6: Images

How are images represented by a computer?

What is metadata?

What is colour depth?

What is resolution?

Mid-Unit Review

Unit 2.6 Review
Data – Images

Key Terms

Bitmap –

Pixel –

Colour depth –

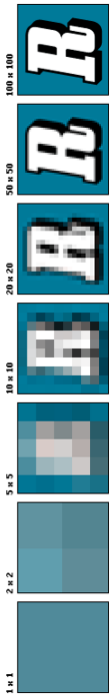
Resolution –

Metadata –

dpi –

Examples of Metadata

-
-
-
-
-



Images – Exam Questions

Duncan prints a 10 x 10 inch photograph with a resolution of 60 DPI.

a) Calculate the total number of pixels in Duncan's photograph [2].

b) Explain how decreasing the DPI would affect the image quality [2].

c) Explain the purpose of metadata in an image file [2].

1-Bit Images

Create a 1-bit image using the 8 x 8 grid.
What binary representation is black?
What binary representation is white?

2-Bit Images

Create a 2-bit image using the 8 x 8 grid.

How many bits are needed per pixel?

U1L7: Sound

How is sound represented by a computer?

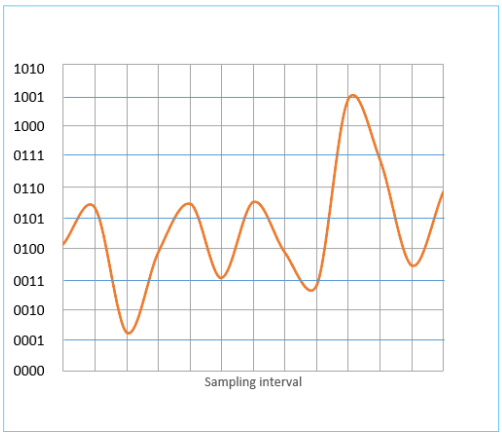
Describe the process used to store sound in digital form.

What is meant by 'sampling interval'?

What is meant by 'sample size'?

What is meant by 'bit rate'?

What is meant by 'sampling frequency'?



Binary for the sound could be: _____ bits.

The quality has: _____

U1L8: Compression

Why is compression used?

What is meant by 'lossy' compression?

What is meant by 'lossless' compression?

Mid-Unit Review

Unit 2.6 Review Data – Sound/ Compression

Key Terms

Analogue –

Sampling –

Sampling Intervals –

Sampling Frequency –

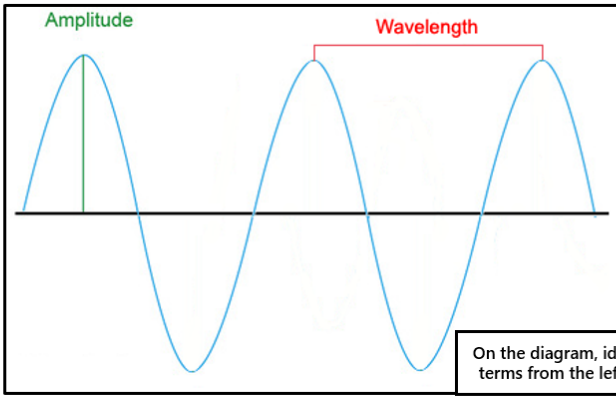
Sample Size –

Bit Rate –

Lossy –

Lossless –

Compression –



On the diagram, identify key terms from the left column

Compression

	Lossy	Lossless
Pros		
Cons		
E.g.		

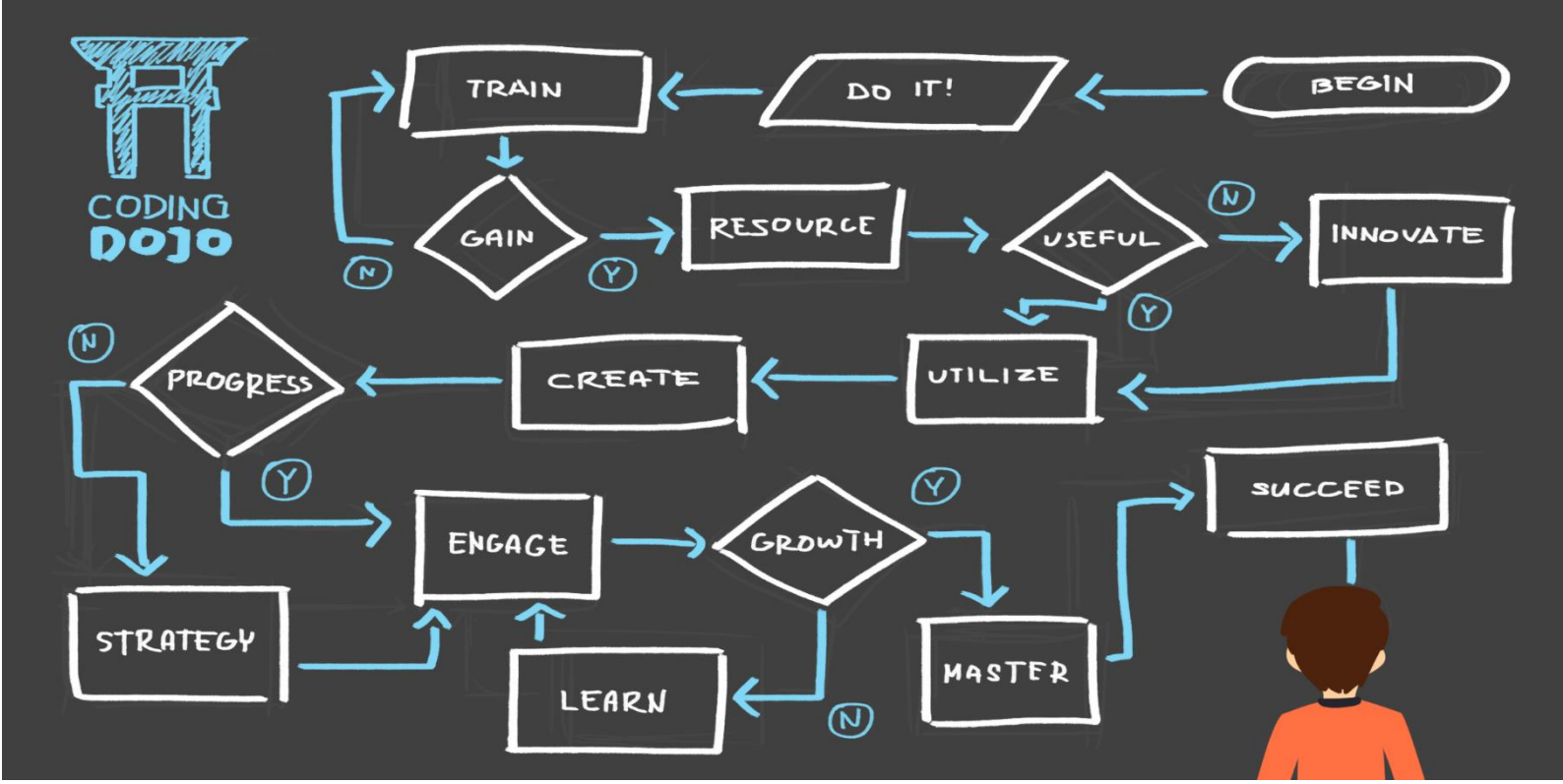
Sound – Exam Questions

Jade records herself reading two extracts from a novel to use in an audiobook. The bit rates and sampling frequencies of each recording are shown in the table.

a) Explain which extract would have the better sound quality [2].

b) Give one drawback of using extract 2 rather than extract 1 for the audiobook [1].

	Length	Bit Rate	Sampling Frequency
Extract 1	2 minutes	128 kbit/s	32 kHz
Extract 2	2 minutes	320 kbit/s	44.1 kHz



Unit 2: Algorithms

Paper 2: Computational Thinking, Algorithms & Programming

This section covers:

- 2.1 Algorithms
  Computational Thinking, Searching, Sorting, Writing and Interpreting Algorithms

U2L1: Computational Thinking

What is Computational Thinking?

What is Abstraction?

What is Decomposition?

What is Algorithmic Thinking?

Exam Questions

1. **A file uploading service won't allow two files with the same file name to be uploaded. If a file name already exists, it will ask the user to change the file name.**
    a. **Describe, with examples, how abstraction can help decide how to compare the files.**
    b. **Describe, with examples, how decomposition could be used to help program this task.**

U2L2: Searching Algorithms

What is a Binary Search?

What is a Linear Search?

U2L3: Sorting Algorithms

How does a Bubble Sort work?

How does a Merge Sort work?

How does an Insertion Sort work?

U2L4: Writing Algorithms (using Pseudocode)

This page is intentionally blank. You should use this to practice developing your algorithmic thinking skills, writing in pseudocode.

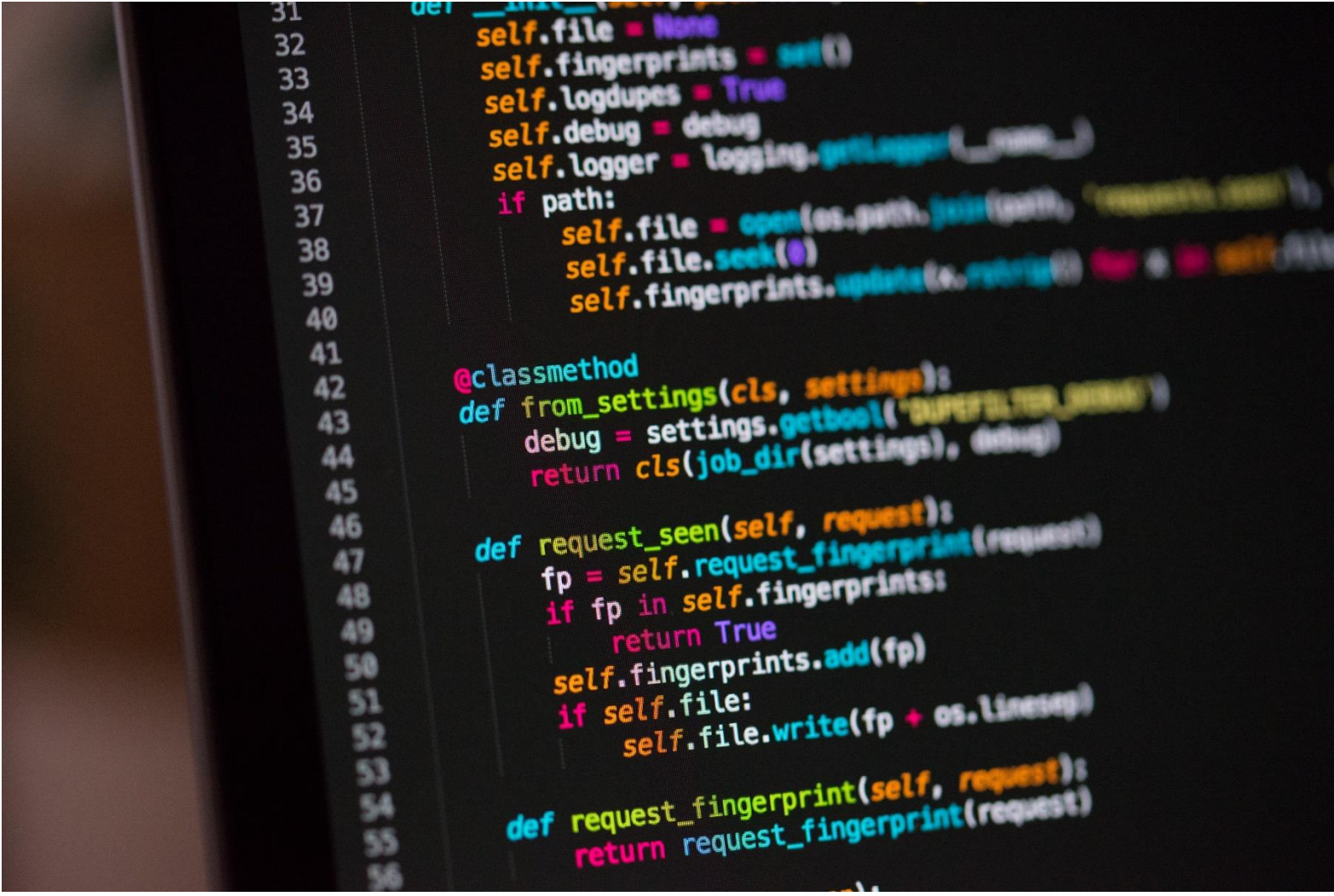
U2L5: Writing Algorithms (using Flowcharts)

Draw and label the key flowchart symbols.

Draw a flowchart to check if a new username is valid. Usernames should be at least five characters long and unique. If it's invalid, the algorithm should give the reason why and get the user to enter another username.

U2L6: Interpreting, Correcting & Completing Algorithms

This page is intentionally blank. You should use this to practice developing your algorithmic thinking skills.



Unit 3: Programming

Paper 2: Computational Thinking, Algorithms & Programming

This section covers:

- 2.2 Programming Techniques
- 2.3 Producing Robust Programs
- 2.5 Translators and Facilities of Languages

U3L1: Programming Basics

What is a variable?

What is a constant?

Identify the key operators in Python. What do they do?

What is an input?

What is an output?

What is meant by 'assignment'?

Identify the main data types, providing examples for each.

Data Type	Description	Example

Identify common arithmetic and Boolean operations.

U3L2: Sequence

What is meant by the term 'sequence'?

Write examples of sequenced Python programs below.

U3L3: Selection

What is meant by the term 'selection'?

Write examples of selection Python programs below.

U3L4: Iteration

What is meant by the term 'iteration'?

What is meant by a 'count-controlled' loop?

What is meant by a 'condition-controlled' loop?

Write examples of iterative Python programs below.

U3L5: String Manipulation

What is meant by string manipulation?

What is concatenation?

Identify some of the key string manipulation commands below, writing them as Python programs.

U3L6: Sub-Programs

What is a sub-program?

Why are sub-programs used?

What are the differences between a function and a procedure?

Write some sub-programs in Python below, including at least one function and procedure.

U3L7: File Handling

What is file handling?

What are the key file handling commands?

Write some example Python programs using file handling below.

U3L8: Arrays

What is an array?

Identify some operations that can be performed on an array.

Write some Python programs using arrays below.

U3L9: SQL

What is a database?

What is SQL?

Create a database table below to store some student data.

Write some SQL commands to query this table, and show the output when this query is run.

U3L10: Defensive Design

Why should programs be carefully designed?

What is input sanitisation?

What is input validation?

Why should we anticipate misuse?

What is authentication?

Identify some examples of authentication.

What is meant by 'program maintainability'?

How can we maintain programs?

U3L11: Testing

What is the purpose of testing?

What is iterative testing?

What is final testing?

What is a syntax error?

What is a logic error?

Draw a test plan to test a login system that uses a username and password.

U3L12: Translators

What are the different levels of programming language?

Type of Language	Description

What is a translator?

What is an assembler?

What is a compiler?

What is an interpreter?

Identify features of an IDE that help with writing programs.

Python Knowledge Organiser

What is Python? Python is a programming language used by large companies to create applications like parts of Google's search engine, Yahoo! Discussion groups and features in YouTube. Python is a high level language and requires the user to type code rather than selecting blocks like in Scratch.

Interacting with the user:

Print a message
```
print('Hello, world!')
```
Print multiple values (of different types)
```
ndays = 365
print('There are', ndays, 'in a year')
```
Asking the user for a string
```
name = input('What is your name? ')
```
Asking the user for a whole number (an integer)
```
num = int(input('Enter a number: '))
```

Deciding between options:

Decide to run a block (or not)
```
x = 3
if x == 3:
  print('x is 3')
```
Decide between two blocks
```
mark = 80
if mark >= 50:
  print('pass')
else:
  print('fail')
```
Decide between many blocks
```
mark = 80
if mark >= 65:
  print('credit')
elif mark >= 50:
  print('pass')
else:
  print('fail')
```
- `elif` can be used without `else`
- `elif` can be used many times

Are two values equal?
```
x == 3
```
⚠ two equals signs, not one

Are two values not equal?
```
x != 3
```
Less than another?
```
x < 3
```
Greater than another?
```
x > 3
```
Less than or equal to?
```
x <= 3
```
Greater than or equal to?
```
x >= 3
```
The answer is a *Boolean*: `True` or `False`

Repeating (Loops/Iteration)

Repeat a block 10 times
```
for i in range(10):
  print(i)
```
Sum the numbers 0 to 9
```
total = 0
for i in range(10):
  total = total + i
print(total)
```
Repeat a block over a string
```
for c in 'Hello':
  print(c)
```
Keep printing on one line
```
for c in 'Hello':
  print(c, end=' ')
print('!')
```
Repeat a block over list (or string) indices
```
msg = 'I grok Python!'
for i in range(len(msg)):
  print(i, msg[i])
```

Count from 0 to 9
```
range(10)
```
⚠ range starts from 0 and goes up to, *but not including*, 10

Count from 1 to 10
```
range(1, 11)
```
Count from 10 down to 1
```
range(10, 0, -1)
```
Count 2 at a time to 10
```
range(0, 11, 2)
```
Count down 2 at a time
```
range(10, 0, -2)
```

String manipulation:

Compare two strings
```
msg = 'hello'
if msg == 'hello':
  print('howdy')
```
Less than another string?
```
if msg < 'n':
  print('a-m')
else:
  print('n-z')
```
⚠ strings are compared character at a time (*lexicographic order*)

Is a character in a string?
```
'e' in msg
```
Is a string in another string?
```
'ell' in msg
```

Convert to uppercase
```
msg.upper()
```
also `lower` and `title`

Count a character in a string
```
msg.count('l')
```
Replace a character or string
```
msg.replace('l','X')
```
Delete a character or string
```
msg.replace('l','')
```
Is the string all lowercase?
```
msg.islower()
```
also `isupper` and `istitle`

Variables:

Creating a variable
```
celsius = 25
```
Using a variable
```
celsius*9/5 + 32
```

Whole numbers (integers):

Addition and subtraction
```
365 + 1 - 2
```
Multiplication and division
```
25*9/5 + 32
```
Powers (*2 to the power of 8*)
```
2**8
```
Convert integer to string
```
str(365)
```

Text (strings):

Single quoted
```
'perfect'
```
Double quoted
```
"credit"
```
Multi-line
```
'''Hello,
World!'''
```
Add (*concatenate*) strings
```
'Hello' + 'World'
```

Pseudocode Keywords

PRINT	INPUT	OUTPUT
NEXT	WHILE	END
DO	UNTIL	IF
ELSE	ELSEIF	ENDIF
GET	REPEAT	FOR
SELECT	STORE	SWITCH
FUNCTION	ENDFUNCTION	PROCEDURE
ENDPROCEDURE	WHERE	AND
OR	START	STOP
ADD	MULTIPLY	DIVIDE
MOD	DIV	SUBTRACT

**NB Not an exhaustive list*

My CS: HUB