Computing Skills Development

NC link & key concept	Skill development across year groups	Year 1	Year 2	Year 3	Year 4	Year 5	Year 6
Communication **To communicate**	Mouse skills	☑ Move mouse ☑ Left & right click	☑ Double click ☑ Show an awareness of vertical scrolling	☑ Use vertical scrolling	☑ Drag and drop using a mouse	☑ Select text with support	☑ Choose areas of text to select and apply a function ☑ Drag and drop text
	Key board skills	☑ Begin to use some type writer keys with support (letters, space bar & backspace)	☑ Know the general position of some type writer keys ☑ Use enter	☑ Begin to learn which keys press with left or right hand ☑ Use curser control keys ☑ Use numeric keypad	☑ Place hands in an appropriate position on a key board to type ☑ Type with developing pace using left and right hand with growing accuracy ☑ Recognise application and system keys	☑ Consistently place hands in an appropriate position to type ☑ Use correct fingers appropriately to type at a developing speed ☑ Use application and system keys when/where appropriate ☑ Recognise function keys	
	Text and images Word PowerPoint	☑ Carry out a simple 'save' ☑ Open work from an easily accessible folder	☑ Save work in an easily accessible folder ☑ Retrieve work in an easily	☑ Save work in a given location with support i.e step by step guidance –	☑ Accurately save work in a given location ☑ Use save and save as functions with prompt	☑ Save work in a more complex location ☑ Recognise and appropriately use save and save as functions ☑ Retrieve and open work from a more complex location	

*Some communication skills appropriately interwoven with application and enquiry skills for KS2			accessible folder & 'open' ☑ Select and carry out a simple 'print' of work	verbal or picture prompts ☑ Understand how to 'overwrite' a save ☑ Retrieve and open work in a given programme ☑ Print work selecting the correct colour option ☑ Copy and paste a simple image or text with support ☑ Alter the design of a document by changing the page colour or adding a border	☑ Retrieve and open work in a given programme ☑ Edit some print functions with support to print work ☑ Move between programmes (e.g Word & Google Chrome) to copy and paste images or text ☑ Format (position) an image, text box or shape with support ☑ Add a simple shape or text box to a document	☑ Know how to find and replace words ☑ Edit print functions to print work ☑ Recognise how and when to format (wrap, move forward/back) an image, text box or shape ☑ Use tabs along the top of a document to edit format, features and additional features (e.g Smart art or a chart)
	Computer Networks	☑ Know that the internet is a network of connected computers	☑ Know that a group of computers that are connected are known	☑ Learn that a hub is a device that joins a group of computers together ☑ With prompts and support, begin to navigate way around different computer networks on the school IT system.		☑ Understand the terms LAN and WAN in the context of computer networks Local Area Network (LAN): Computers connected together

			as a Network ☑ Understand that some computers have different ‘networks’			that are geographically close to each other. Wide Area Network (WAN) computers connected together that are geographically far apart, even in different countries. ☑ Identify and locate different networks on a computer and follow instructions to use and access.
	Using the Internet	☑ Know what an online search engine is	☑ Follow simple step instructions to locate a search engine e.g find Google Chrome on a computer desktop	☑ Know how to access a search engine ☑ Carry out a simple search with some guidance ☑ Select an appropriate web page with guidance ☑ Navigate way around a webpage to retrieve information with support	☑ Use a search engine to access and retrieve information online ☑ Recognise & understand the purpose of different features e.g images, videos, news etc ☑ Understand that some search results may be more reliable than others ☑ Navigate a web page with growing confidence	☑ Edit the settings on a search engine to carry out a more complex search ☑ Use ‘tools’ to narrow or widen a search ☑ Recognise which search results will be most reliable ☑ Know how to search the internet safely ☑ Confidently locate web pages, navigate and retrieve appropriate information from them
Online Safety (Digital Literacy)	Use technology safely, respectfully and responsibly	☑ Ask before you watch ☑ How to make an Avatar		☑ The SMART rules ☑ Play like Share		☑ Google Internet Legends ▪ Protect your stuff ▪ Respect each other

Application and enquiry **To connect** **To communicate** Detailed overview of each resource within separate Digital Literacy documents				☑ What is email and how can we use it safely? ☑ Google Internet Legends ▪ Think before you share ▪ Check it's for real	☑ Screen time and healthy balance ☑ Keeping personal information personal
	Know where to go for help & how to raise concerns about inappropriate content on the internet	☑ Lee & Kim's adventures ☑ Jessie & friends ☑ Smartie the Penguin **Books** ☑ DigiDuck ☑ Penguinpig ☑ Monkeycow ☑ Chickenclickin ☑ Webster's Bedtime			☑ Fake news ☑ Band runner ☑ Caught in the web
E-safety content should also be taught as/when the teacher thinks it is relevant or necessary for example, when common issues arise or when appropriate to a topic in school or in the news. They should also be continually revisited throughout other areas of the computing curriculum (i.e recap of SMART rules)					
APPLICATION AND ENQUIRY (Information Technology- IT) **To connect** **To collect** **To communicate** *Additional elements linked with	Recognise common uses of information technology beyond school	☑ Spot digital technology in school	☑ Show an awareness of digital technology's differing purposes in school ☑ Have some knowledge of digital technology outside of school e.g a barcode scanner in a shop	☑ Have knowledge of digital technology used outside of school ☑ Be able to give some examples of uses of digital technology outside of school	☑ Have knowledge of a range of different purposes of digital technology outside of school ☑ Recognise the role digital technology plays alongside its purpose and function in today's society

communication aspect and recorded there	Select, use and combine a variety of software	Skills progression and suggested activities in separate LTP document					
Programming (Computer Science) **To code**	Coding	**Apps**					
		Bee Bot Daisy the Dinosaur		ALEX Light Bot hour of code		Cargo Bot Light Bot hour of code	
		☑ To follow simple step instructions within a programme ☑ To know that programs work by following precise instructions ☑ To know how to use the arrow keys to accomplish a specific goal **Purple Mash 2code** **<u>Chimp level</u>** activities T user guide on PM	**Studio Code Course B** ☑ To know how algorithms are implemented as programs on digital devices ☑ I can create and debug simple programs ☑ I can use logical reasoning to predict the behaviour of simple programs **An 'hour of code' project** can be used a supplementing unit if necessary. **Purple Mash 2code**	**Studio Code Course C** ☑ I can create and debug simple programs ☑ I can use logical reasoning to predict the behaviour of simple programs ☑ I can use sequence, selection, and repetition in programs. **An 'hour of code' project** can be used a supplementing unit if necessary. **Purple Mash 2code** **<u>Chimp level</u>** activities T user guide on PM	**Studio Code Course D** ☑ I can design, write and debug programs that accomplish specific goals. ☑ I can use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs. ☑ I can use sequence, selection, and repetition in programs. **An 'hour of code' project** can be used a supplementing unit if necessary. **Purple Mash 2code** **<u>Gibbon level</u>** activities T user guide on PM	**Studio Code Course E ramp up revision** ☑ I can design, write and debug programs that accomplish specific goals. ☑ I can use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs. ☑ I can use sequence, selection, and repetition in programs. **An 'hour of code' project** can be used a supplementing unit if necessary. **Purple Mash 2code**	**Studio Code Course E content** ☑ I can design, write and debug programs that accomplish specific goals. ☑ I can use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and

			Chimp level activities T user guide on PM	**Scratch project** c to revisit and reapply skills in a different context. - This can be linked to a topic e.g the Romans	**Scratch project** to revisit and reapply skills in a different context. This can be linked to a topic e.g the Ancient Egyptians	**Gibbon level** activities T user guide on PM **Scratch project** to revisit and reapply skills in a different context. This can be linked to a topic e.g the Mayans	*programs.* ☑ *I can use sequence, selection, and repetition in programs.* **An 'hour of code' project** can be used a supplementing unit if necessary. **Purple Mash 2code** **Gorilla level** activities T user guide on PM **Scratch project** to revisit and reapply skills in a different context. This can be linked to a topic e.g Shang Dynasty
	Terminology Progression	• Code • program • search • goal • error • character	• Algorithm • Debug • Debugger • Debugging • repetition • selection	• Condition • Data • Input • Output • Pattern • Software	• Generalisation • Logical reasoning/thinking • Program control	• Detect • Correct • Packet (small pieces of data) • String	Children should be confident to use all previous vocabulary in the correct context.

		• coder • bug • button	• sequence • sprite • stage • evaluation • actions • command • object	• Controls • Script • Motion • Conditional operator • Variables • Design mode • Events • Functions • Method • Block		• User Interface	

Coding Glossary

Actions: Actions are types of commands which are run on a particular object, and cause it to alter its behaviour. Actions could be used to move an object, for example "UP" "DOWN" or "STOP". Actions are often called "methods". See also (object)

Algorithm: A precise step by step set of instructions used to solve a problem or achieve an objective.

Assignment operator: A type of operator that is used to assign or reassign (or change) the value of a Variable. Examples are "set to" which changes a Variable to a new value. For example the code:"A SET TO 2" will change the value of the Variable A to 2. See also (Variable)

Block: A group of commands that are grouped together and are run when a specific condition is met or when an event occurs. For example one could have a "WHEN CLICKED" command and the commands in the "WHEN CLICKED" block would be run when the mouse click occurs. In 2Code, commands in a block are given the same indentation and background shading to indicate they are part of the same block. In real code mode and in many other computer languages blocks of code are indicated with the use of curly brackets { } . See also (events)

Bug: A mistake in computer code that prevents the computer program from behaving in the way the coder intended. See also (debugging,debugger)

Button: An element on the screen that the user can click on. Usually click on a button generates an event that runs some code. See also (user interface)

Character: See: Object

Coder: A person who writes computer code.

Collision detection: Detecting when two sprites on the screen bump into each other. They are often used in a game to detect when a character hits a 'baddy'. See also (sprite)

Command: A command is a single instruction within a computer program. A computer program usually contains several commands. Sometimes commands are called 'statements'.

Concatenation: Adding text together is called concatenation. For example one could add two pieces of text 'The cat '+ 'sat on the mat' to create a single piece of text 'The cat sat on the mat.'

Condition: This is the 'trigger' for an 'IF' or 'REPEAT UNTIL' command, and is the test that must be fulfilled to trigger the next stage. The test result determines whether or not to run the "IF" or "ELSE" block in an "IF/ELSE" command or whether to keep repeating in a "REPEAT UNTIL" command. For example in the snippet:
If A EQUALS B THEN PRINT...
"A EQUALS B" is the condition See also (selection,IF/ELSE)

Conditional operator: An operator (symbol) which evaluates to either true or false depending on the values either side of it. It is used as part of a conditon. Examples are "equals" (as in: 'IF A=B') which will evaluate to true if the values eitherr side of the operator are the same. Other examples include "not equals" or "less than" or "greater than" See also (operator,condition,selection,if/else,repeat until)

Console log: The console log is an output 'window' for the computer program that used purely for debugging purposes. It typically is a scrolling list of messages. The messages could contain information about what the program is doing or they could be notification of errors or problems within the program. See also (debugging,debugger)

Control: See: Program Control

Debugger: A tool that helps coders fix problems in their code. Debuggers often contain a console log, the ability to pause a program, step through a program line by line and the ability to inspect Variables. See also (debugging,coder)

Debugging: Fixing problems in code. Often computer programmers spend as much time debugging code as writing code. See also (debugger,coder)

Design Mode: In 2Code, Design Mode is used to create the user interface of a computer program (the part of the computer program that the user sees). Using Design Mode a coder can create objects and move them around on the screen. See also (user interface)

Events: An event is an occurrence that causes a block of code to be run. The event could be time related (see timer) or could be some kind of user input such as the user pressing a key or clicking the screen. In 2Code, the event commands are used to create blocks of code that are run when events happen. See also (block)

Functions: Sometimes a coder wants a group of commands to be run many times within the same program. To save having to repeat the group of commands the coder can put the commands into a 'function' and can give the function a name. Then the coder can 'call' the function (use its name within the program), which will run all the commands in that function.
IF/ELSE: An "IF/ELSE" command tests a condition. If the condition is true then the commands inside the "IF" block are run. If the condition is not true and there is an "ELSE" block then the commands inside the "ELSE" block are run. See also (condition,block)

Input: Input is information going into the computer. An input could be user the moving or clicking the mouse, or the user entering characters on the keyboard. On tablets there are other forms of input such as finger swipes, touch gestures and tilting the device.

Logical operators: Logical operators are used for combining conditions, allowing for complex tests to be created. The most common examples of logical operators are "AND" and "OR". An example of a condition using a logical operator could be "IF A GREATER THAN 0 **AND** A LESS THAN 10 THEN A IS BETWEEN 1 AND 9". For completeness, logical operators have been included in 2Code, but they are beyond the primary school curriculum. Some of the Gorilla (advanced) debug challenges touch upon them. See also (operator,conditon)

Mathematical operator: An operator which functions as a typical mathematical statement, eg "+","-","x" or "/". The mathematical operator "+" in the code "A+2" will evaluate to 4, if the Variable A is equal to 2. See also (operator,assignment operator)

Method: Another word for an action. See also (action)

Object: An object is an element in a computer program that can be created and manipulated using the object's actions or properties. In 2Code all the elements on the screen are objects. See also (action,properties)

Operator: A symbol that represents a process to apply to the objects on either side, for example "+" , "=", or "AND". For example, "a+b" or "IF a=b". In 2Code there are four types of operators: assignment operators, conditional operators, mathematical operators and logical operators. See also (assignment operator,conditional operator,mathematical operator,logical operator)

Output: Output is information that comes out of the computer. This could be items that appear on the screen or sound that comes out of the speakers. Examples of output are "Print to screen" and "Sound".

Packet: Small pieces of data (text, pictures, sound)

Program Control: How parts of the program will run, how often and in some cases, when.

Programmer: See: coder

Properties: Properties are qualities that are associated with an object. Examples include colour, speed or angle. Properties of an object can be changed in a similar way to Variables using assignment operators. See also (operators,actions,Variable)

Repeat: In 2Code a "repeat" command can be used to make a block of commands run a set number of times or to repeat a block of commands forever. See also (sequencing)

Repeat Until: in 2Code a "repeat until" command will repeat a block of commands until a condition is met. See also (condition,sequencing)

Selection: A decision command, where a program chooses a different outcome depending on a condition, such as "REPEAT...UNTIL" or "IF...ELSE".

Sequencing: When a computer program repeats a sequence of commands. In 2Code this could be done using "REPEAT", "REPEAT UNTIL" or using a "Timer". See also (repeat,repeat until,timer)

Sprite: An element on the screen that is typically an image. Sprites are often animated and they can be set to move around the screen and can be used, for instance, to represent characters within a game. See also (user interface)

Statement: Another word for a command. See also (command)

Strings: In code, pieces of text are often called strings.

Timer: The timer is a command in 2Code that allows a block of commands to be run either after a timed delay or at regular timed intervals. Real programming languages often have a similar type of command. See also (sequencing,block)

User interface: The part of the computer program that the user sees on the screen. Usually this consists of various visible elements such as buttons, sprites or input fields. See also (design mode)

Variable: Variables are used in programming to keep track of the things that can change while a program is running, for example, the on/off state of a switch (see SWITCHING BACKGROUND lesson), or to count the number of swipes before changing a lamp into a genie (see GENIE lesson) or the numbers changing in a timer (see NIGHT & DAY gibbon lesson). Variables are like boxes that the computer can use to store information. In order to find the information in the box, each box should be labelled. Therefore, each variable (each of our boxes) needs to have a name. The name should be something that helps you remember what it is. The information inside the box is called the variable value. The user, the program or another variable can change this variable value. In 2Code, variables can be either numbers or text (words, phrases or even whole sentences). See also (operators,object,properties,assignment operators)

Variable watch: The variable watch is a debugging tool that allows us to watch the values of our variables as our programs run. See also (variable)