Computer Science							
	Foundation	Y1	Y2	Y3	Y4	Y5	Y6
Procedural Knowledge	Separate document	1a Understand what algorithms are; how they are implemented as programs on digital devices; and that programs execute by following precise and unambiguous instructions. 1b Create and debug simple programs. 1c Use logical reasoning to predict the behaviour of simple programs.	2a Understand what algorithms are; how they are implemented as programs on digital devices; and that programs execute by following precise and unambiguous instructions. 2b Create and debug simple programs. 2c Use logical reasoning to predict the behaviour of simple programs.	3a Design, write and debug programs that accomplish specific goals, including controlling or simulating physical systems; solve problems by decomposing them into smaller parts. 3b Use sequence, selection and repetition in programs; work with variables and various forms of input and output.	4a Use sequence, selection and repetition in programs; work with variables and various forms of input and output. 4b Use sequence, selection and repetition in programs; work with variables and various forms of input and output.	5a Design, write and debug programs that accomplish specific goals, including controlling or simulating physical systems; solve problems by decomposing them into smaller parts. 5b Use sequence, selection and repetition in programs; work with variables and various forms of input and output.	6a Design, write and debug programs that accomplish specific goals, including controlling or simulating physical systems; solve problems by decomposing them. 6b Use sequence, selection and repetition in programs; work with variables and various forms of input and output.

				3c Use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs. 3d Understand computer networks, including the internet; how they can provide multiple services, such as the World Wide Web, and the opportunities they offer for communication and collaboration.	4c Use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs. 4d Understand computer networks, including the internet; how they can provide multiple services, such as the World Wide Web, and the opportunities they offer for communication and collaboration	5c Use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs. 5d Understand computer networks, including the internet; how they can provide multiple services, such as the World Wide Web, and the opportunities they offer for communication and collaboration.	6c Use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs. 6d Understand computer networks, including the internet; how they can provide multiple services, such as the World Wide Web, and the opportunities they offer for communication and collaboration.
Conceptual Knowledge		1a Children understand that an algorithm is a set of instructions used to solve a problem or	2a Children can explain that an algorithm is a set of instructions to complete a task. When designing	3a Children can turn a simple real-life situation into an algorithm for a program by	4a When turning a real life situation into an algorithm, the children's design shows that they are	5a Children may attempt to turn more complex real-life situations into algorithms for a program by deconstructing	6a Children are able to turn a more complex programming task into an algorithm by

		achieve an objective. They know that an algorithm written for a computer is called a program. 1b Children can work out what is wrong with a simple algorithm when the steps are out of order, e.g. The Wrong Sandwich in Purple Mash and can write their own simple algorithm, e.g. Colouring in a Bird activity. Children know that an unexpected outcome is due to the code they have created and can make logical attempts to fix	simple programs, children show an awareness of the need to be precise with their algorithms so that they can be successfully converted into code. 2b Children can create a simple program that achieves a specific purpose. They can also identify and correct some errors, e.g. Debug Challenges: Chimp. Children's program designs display a growing awareness of the need for logical, programmable steps.	deconstructing it into manageable parts. Their design shows that they are thinking of the desired task and how this translates into code. Children can identify an error within their program that prevents it following the desired algorithm and then fix it. 3b Children demonstrate the ability to design and code a program that follows a simple sequence. They experiment with timers to achieve repetition effects in their programs. Children are beginning to understand the	thinking of the required task and how to accomplish this in code using coding structures for selection and repetition. Children make more intuitive attempts to debug their own programs. 4b Children's use of timers to achieve repetition effects are becoming more logical and are integrated into their program designs. They understand 'if statements' for selection and attempt to combine these with other coding structures including variables to achieve the effects that	it into manageable parts. Children are able to test and debug their programs as they go and can use logical methods to identify the approximate cause of any bug but may need some support identifying the specific line of **code**. 5b Children can translate algorithms that include sequence, selection and repetition into code with increasing ease and their own designs show that they are thinking of how to accomplish the set task in code utilising such	identifying the important aspects of the task (abstraction) and then decomposing them in a logical way using their knowledge of possible coding structures and applying skills from previous programs. Children test and debug their program as they go and use logical methods to identify the cause of bugs, demonstrating a systematic approach to try to identify a particular line of code causing a problem. 6b Children translate algorithms that include sequence,

		the code, e.g. Bubbles activity in 2Code. 1 1c When looking at a program, children can read code one line at a time and make good attempts to envision the bigger picture of the overall effect of the program. Children can, for example, interpret where the turtle in 2Go challenges will end up at the end of the program.	2C Children can identify the parts of a program that respond to specific events and initiate specific actions. For example, they can write a cause and effect sentence of what will happen in a program.	difference in the effect of using a timer command rather than a repeat command when creating repetition effects. Children understand how variables can be used to store information while a program is executing. 3c Children's designs for their programs show that they are thinking of the structure of a program in logical, achievable steps and absorbing some new knowledge of coding structures. For example, 'if' statements, repetition and	they design in their programs. As well as understanding how variables can be used to store information while a program is executing, they are able to use and manipulate the value of variables. Children can make use of user inputs and outputs such as 'print to screen'. e.g. 2Code. 4c Children's designs for their programs show that they are thinking of the structure of a program in logical, achievable steps and absorbing some new knowledge of coding structures. For	structures. They are combining sequence, selection and repetition with other coding structures to achieve their algorithm design. 5c When children code, they are beginning to think about their code structure in terms of the ability to debug and interpret the code later, e.g. the use of tabs to organise code and the naming of variables. 5d Children understand the value of computer networks but are also aware of the main	selection and repetition into code and their own designs show that they are thinking of how to accomplish the set task in code utilising such structures, including nesting structures within each other. Coding displays an improving understanding of variables in coding, outputs such as sound and movement, inputs from the user of the program such as button clicks and the value of functions. 6c Children are able to interpret a program in parts and can make logical attempts to put

				variables. They make good attempts to 'step through' more complex code in order to identify errors in algorithms and can correct this. e.g. traffic light algorithm in 2Code. In programs such as Logo, they can 'read' programs with several steps and predict the outcome accurately. 3d Children can list a range of ways that the internet can be used to provide different methods of communication. They can use some of these methods of communication, e.g. being able to open,	example, 'if' statements, repetition and variables. They can trace code and use step-through methods to identify errors in code and make logical attempts to correct this. e.g. traffic light algorithm in 2Code. In programs such as Logo, they can 'read' programs with several steps and predict the outcome accurately. 4d Children recognise the main component parts of hardware which allow computers to join and form a network. Their ability to understand the	dangers. They recognise what personal information is and can explain how this can be kept safe. Children can select the most appropriate form of online communications contingent on audience and digital content, e.g. 2Blog, 2Email, Display Boards.	the separate parts of a complex algorithm together to explain the program as a whole. 6d Children understand and can explain in some depth the difference between the internet and the World Wide Web. Children know what a WAN and LAN are and can describe how they access the internet in school.

				respond to and attach files to emails using 2Email. They can describe appropriate email conventions when communicating in this way.	online safety implications associated with the ways the internet can be used to provide different methods of communication is improving.		
Vocabulary (Vocabulary definitions can be found on T:\Curriculum\Computing\Vocabulary)		1.4- Lego builders instruction, algorithm, computer, program, debug. 1.5- Maze explorers Direction, challenge, arrow, undo, rewind, forward, backwards, right turn, left turn, debug, instruction, algorithm. 1.7 Coding Action, background, button, character,	2.1 coding Action, algorithm, bug, character, code block, code design, command, debugging, design mode, input, object, properties, repeat, scale, when key, when clicked, timer.	3.1 coding Action, algorithm, bug, code block, code design, command, debugging, design mode, event, if, input, output, repeat, object, properties, timer, computer simulation, selection, variable. 3.5 Email Communication, email, compose, send, CC, attachment, formatting, report to the	4.1 Coding Action, algorithm, bug, code block, code design, command, debugging, design mode, event, if, input, output, repeat, object, properties, timer, computer simulation, selection, variable. 4.5 Logo Logo, BK, FD, RT, LT, repeat, SETPC, SETPS, PU, P. 4.7 Effective searching	5.1 Coding Action, algorithm, bug, code block, code design, command, debugging, design mode, event, if, input, output, repeat, object, properties, timer, computer simulation, sequence, selection, variable. 5.5 Game creator Animation, computer game, customise, evaluation, image,	6.1 Coding Action, alert, algorithm, code design, command, debugging, design mode, event, if, input, output, repeat, object, properties, timer, tabs, computer simulation, sequence, selection, variable. 6.4 Databases Audience, blog, blog page, blog post, collaborative, icon.

		code block, code design, coder, coding, collision detection, command, design mode, object, program, properties, scale, stop command, when key, sound, when clicked.		teacher, password, address book, save to draft.	Easter egg, internet, internet browser, search, search engine, spoof website, website.	instructions, interactive, screenshot, texture, perspective, playability.	6.6 Networks Internet, world wide web, network, router, Local area network (LAN), wide area network (WAN), network cables, wireless. 6.7 Quizzing Audience, collaboration, concept map, database, quiz.
POSSIBLE END POINTS		Create a maze for the superhero to save the people in danger.	2a links to blended learning and following instructions through class dojo.	3a links to using class dojo and communicating with class teacher. Class assembly via dojo.	Design and create a game or problem-solving activity using 2 code. Can also be linked to topic being taught at that time.	Use the skills taught to create a class workshop to help younger children to navigate parts of the internet or turning real life situations into algorithms	Begin to use 2blog to communicate within class. Design and create their own blogs.
SMSC/GLOBAL GOALS/ BRITISH V./COMMUNITY LINKS		Moral and social development	Moral and social development	Moral and social development	Moral and social development	Moral and social development	Moral and social development

		British value link to enable students to develop their self-knowledge, self-esteem and self-confidence.	British value link to enable students to develop their self-knowledge, self-esteem and self-confidence.	British value link to enable students to develop their self-knowledge, self-esteem and self-confidence.	British value link to enable students to develop their self-knowledge, self-esteem and self-confidence.	British value link to enable students to develop their self-knowledge, self-esteem and self-confidence.	British value link to enable students to develop their self-knowledge, self-esteem and self-confidence.