Computing Vertical Concepts

Computer Science

Year 1
Algorithm is a set of instructions used to solve a problem or achieve an objective. Computer program turns an algorithm into code that the computer. Understand what is wrong with a simple algorithm when the steps are out of order. An unexpected outcome is due to the code and can make logical attempts to fix the code. Read code one line at a time and make good attempts to envision the bigger picture of the overall effect of the program. Interpret where the turtle in 2Go challenges will end up at the end of the program.

Year 2
Algorithm is a set of instructions to complete a task. Awareness of the need to be precise with their algorithms so that they can be successfully converted into code. Create a simple program that achieves a specific purpose. 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. Identify the parts of a program that respond to specific events and initiate specific actions. Write a cause and effect sentence of what will happen in a program.

Year 3
Turn a simple real-life situation into an algorithm for a program by deconstructing it into manageable parts. To think about the desired task and how this translates into code. Identify an error within their program that prevents it following the desired algorithm and then fix it. To design and code a program that follows a simple sequence. Experiment with timers to achieve repetition effects in their programs. Beginning to understand the difference in the effect of using a timer command rather than a repeat command when creating repetition effects. 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, respond to and attach files to emails using 2Email. Describe appropriate email conventions when communicating in this way.

Year 4
Turn a real-life situation into an algorithm, the children's design shows that they are thinking of the required task and how to accomplish this in code using coding structures for selection and repetition. Intuitive attempts to debug their own programs. Timers to achieve repetition effects are becoming more logical and are integrated into their program designs. 'IF statements' for selection and attempt to combine these with other coding structures including variables to achieve the effects that they design in their programs. Variables can be used to store information while a program is executing, they are able to use and manipulate the value of variables. User inputs and outputs such as 'print to screen'. e.g. 2Code. 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 variables. They can trace code and use step-through methods to identify errors in code and make logical attempts to correct this. In programs such as Logo, they can 'read' programs with several steps and predict the outcome accurately. Recognise the main component parts of hardware which allow computers to join and form a network. Online safety implications associated with the ways the Internet can be used to provide different methods of communication is improving.

Salisbury Manor Primary School
The best in everyone™
Part of United Learning

Computing Vertical Concepts

Computer Science

Year 5
Attempt to turn more complex real-life situations into algorithms for a program by deconstructing it into manageable parts. 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. 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 structures. Combining sequence, selection and repetition with other coding structures to achieve their algorithm design. 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. Value of computer networks but are also aware of the main dangers. Personal information is and can explain how this can be kept safe. Select the most appropriate form of online communications contingent on audience and digital content, e.g. 2Blog, 2Email, Display Boards

Year 6
Turn a more complex programming task into an algorithm by 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. 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. Translate algorithms that include sequence, 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. Interpret a program in parts and can make logical attempts to put the separate parts of a complex algorithm together to explain the program as a whole. Understand and can explain in some depth the difference between the internet and the World Wide Web. What a WAN and LAN are and can describe how they access the internet in school.