Suffolk's Computer Science Progression Years 1 – 7

In this progression you will see repeated statements across the key stages. The Learning Outcomes (written as 'I can' statements and highlighted in purple) provide more detail and reflect the increasingly complexity of the problems and range of languages children are expected to encounter.

As a Year 1 I can...	As a Year 2 I can...	As a Year 3 I can...	As a Year 4 I can...	As a Year 5 I can...	As a Year 6 I can...	As a Year 7 I can...
Objective #1: Understand what algorithms are; how they are implemented as programs on digital devices; and that programs execute by following precise and unambiguous instructions.		*Objective #1: design, write and debug programs that accomplish specific goals, including controlling or simulating physical systems; solve problems by decomposing them into smaller parts.*				
Objective #2: Create and debug simple programs.		*Objective #2: use sequence, selection, and repetition in programs; work with variables and various forms of input and output.*				
Objective #3: Use logical reasoning to predict the behaviour of simple programs.		*Objective #3: use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs.*				
Recognise that many everyday devices respond to instructions.	**Recognise that a computer carries out the instructions contained in a program.**					
I can name everyday programmable devices and say what they can do. I can sort everyday devices into programmable and non-programmable.	I can tell my friend to do something in the right order and say that this is an algorithm. I can control a toy by programming a sequence of instructions and say that this is an algorithm. I can explain what an algorithm is. I can give an example of an algorithm. I understand that programs are algorithms working on computers.					
Describe storyboards of a programmable toy.	**Draw my own storyboards for a sprite.**	**Put the parts of a simple process in order.**	**Recognise similarities between storyboards of processes.**	**Plan a task by drawing a diagram of the process.**	**Plan a task by drawing a standardised diagram of the process.**	**Plan a task by drawing a standardised diagram of the process.**
I can follow a given sequence of instructions to program a programmable toy. I can use predictions to select the correct set of instructions for a programmable toy to follow. I can create a sequence of	I can create a set of instructions for my sprite to follow. I can predict what will happen when I run my program.	I can physically carry out a simple process (e.g., make a sandwich). I am able to visualise the process and explain the process orally (e.g., give directions to someone else to make a sandwich). I understand the importance of chronology.	I can analyse and identify the similarities between storyboards of processes (link to repetition and reusing and remixing).	I can identify the key parts of the problem. I can plan a task using my own notation.	I can plan a task using my standard notation. I can decide on the best programming language to use. I can solve a complex problem by decomposing it into smaller parts.	I can create a flowchart to solve a problem. I can follow a flowchart that solves a problem. I can explain the expected outcomes of a flowchart.

instructions for a programmable toy to follow.						
						Analyse algorithms and select the most efficient for given problem.
						I can demonstrate that there might be more than one algorithm to solve a problem.
Create programs to make people or programmable toys do things.	**Write a program to control a virtual output.**	**Write a program to carry out a simple task.**	**Write a program that solves a problem by using repetition.**	**Write a program that solves a problem by using selection.**	**Write a program that solves a problem by using variables.**	**Use two or more programming languages.**
I can give sequential instructions to move another child around obstacles. I can accurately follow instructions from another child. I can give sequential instructions to a programmable toy to tell it where I want it to go. I can identify several ways of moving my programmable toy.	I can give sequential multi-step instructions to a sprite to tell it where I want it to go. I can identify several ways of moving my sprite.	I understand the purpose of a program relating to its output (what is it going to do)? I understand the chronology of a sequence. I can explain orally the sequence involving a number of steps and appropriate detail. I can use computing language (could be standard English using imperatives and adverbs or specific formats within a programming language). I understand that there might be more than one algorithm to solve a problem.	I understand that programs can contain multiple sequences that can be performed in any order. I can identify real-world events that have an element of repetition. I understand why it is necessary to repeat things. I can locate elements of the sequence that could use repetition (e.g. square on logo). I understand that there might be more than one algorithm to solve a problem.	I can identify the conditionals that need to be used. I can identify real-world events that use conditionals. I understand that there might be more than one algorithm to solve a problem.	I can identify the variables that need to be used. I can identify real-world events that use variables. I understand that there might be more than one algorithm to solve a problem.	I can create the solution to a problem in a block-based language. I can relate the computational concepts in a text-based language to those in a block-based language.
						Understand how numbers can be represented in binary, and be able to carry out simple operations on binary numbers.
						I can convert from 8-bit binary to decimal. I can count in binary. I can add two binary numbers together.
Debug my program.	**Debug my program.**	**Debug my program.**	**Debug my program.**	**Debug my program.**	**Debug my program.**	**Debug my program.**
I can test my instructions for my programmable toy	I can test my instructions for my sprite and fix	I am able to do things in small steps and check my	I can identify what each element of the sequence	I can identify what each element of the sequence	I can choose from a range of methods to	I am iterative and incremental in my

and fix mistakes if I need to, with adult support.	mistakes if I need to, with adult support. I can fix mistakes from a given set of instructions and test them. I am able to do things in small steps and check my work as I go.	work as I go. I can identify what the problem is.	does. I can identify what and where the problem is. I can analyse a program to locate a problem. I know how to fix my program.	does. I can identify what and where the problem is. I can analyse a sequence of code to locate a problem. I know how to fix my code.	independently debug my program. I can explain the choices I took to debug my program and why. I can interpret others' programs and explain where the problem occurs.	development of a program, recognising the colour-coding in the Integrated Development Environment (IDE) and solving errors in code as I build my program.