Cambridgeshire Progression in Computing Capability

Programming:

Purpose of study ~ Computing programmes of study: Key stages 1 and 2

A high-quality computing education equips pupils to use computational thinking and creativity to understand and change the world. Computing has deep links with mathematics, science, and design and technology, and provides insights into both natural and artificial systems. The core of computing is computer science, in which pupils are taught the principles of information and computation, how digital systems work, and how to put this knowledge to use through programming.

Building on this knowledge and understanding, pupils are equipped to use information technology to create programs, systems and a range of content. Computing also ensures that pupils become digitally literate – able to use, and express themselves and develop their ideas through, information and communication technology – at a level suitable for the future workplace and as active participants in a digital world.

Aims:

- The national curriculum for computing aims to ensure that all pupils:
- can understand and apply the fundamental principles and concepts of computer science, including abstraction, logic, algorithms and data representation
- can analyse problems in computational terms, and have repeated practical experience of writing computer programs in order to solve such problems
- can evaluate and apply information technology, including new or unfamiliar technologies, analytically to solve problems
- are responsible, competent, confident and creative users of information and communication technology.

Theme Overview: Programming

It's worth noting that computer science aims to cover two distinct, but related, aspects. There's a focus on computer science itself (the ideas and principles that underpin how digital technology works) but this sits alongside the practical experience of programming, almost certainly the best way for primary pupils to learn about computer science.

Computer Science is more than programming, but programming is an absolutely central process for Computer Science. In an educational context, programming encourages creativity, logical thought, precision and problem-solving, and helps foster the personal, learning and thinking skills required in the modern school curriculum. Programming gives concrete, tangible form to the idea of "abstraction", and repeatedly shows how useful it is.

Computing at School (CAS) March 2012: Computing in the National Curriculum – A Guide for Primary Teachers

Cambridgeshire Progression in Computing Capability

	Early Capability		Middle Capability		Later Capability	
	Year 1	Year 2	Year 3	Year 4	Year 5	Year 6
National Curriculum	▪ Understand what algorithms are; how they are implemented as programs on digital devices; and that programs execute by following precise and unambiguous instructions ▪ Create and debug simple programs ▪ Use logical reasoning to predict the behaviour of simple programs		▪ Design, write and debug programs that accomplish specific goals, including controlling or simulating physical systems; solve problems by decomposing them into smaller parts ▪ Use sequence, selection, and repetition in programs; work with variables and various forms of input and output ▪ Use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs			
Cambridgeshire Capability Statements	Pupils create, **debug** and implement instruction (simple **algorithms**) as **programs** on a range of digital devices. Pupils understand that **digital devices** follow precise and unambiguous instructions. Pupils understand that digital devices **simulate** real situations.	Pupils understand that **algorithms** are implemented as **programs** on **digital devices.** Pupils create and **debug programs** to achieve specific goals. Pupils use the **principles of logical reasoning** to plan and predict the behaviour of simple **programs**. Pupils solve real and imaginary problems on and off screen.	Pupils create **programs** to accomplish specific goals: - using an increasing range of **digital devices** and **applications**. - exploring and understanding the impact of changing instructions. - using **sequence** and **repetition** - **decomposing** problems both on and off screen - using the **principles of logical reasoning** in order to resolve problems.	Pupils create and debug **programs**: - using **sequence** and **repetition**. - refining **algorithms** to improve efficiency - controlling or simulating **physical systems**. Pupils begin to explore and notice the similarities and differences between **programming languages** and use this knowledge to help them create and **debug programs** efficiently.	Pupils create, **deconstruct** and refine **programs** to accomplish specific goals. They can: - improve efficiency - use **selection** within programs - use a range of simple **inputs** and **outputs** to control or simulate **physical systems**. Pupils use **logical reasoning** to explain how some **algorithms** work and to detect and correct errors in **programs**. They independently employ strategies to solve problems.	Pupils deconstruct, improve and create **programs** including: - using selection and working with variables. - using the principles of logical reasoning - challenging themselves by making simple programs increasingly complex and employ a variety of strategies to solve problems. Pupils can explain why they have structured algorithms as they have and describe the effect this has on a program.

the ict service. education
info@theictservice.org.uk

DRAFT (September 2014)

Cambridgeshire Progression in Computing Capability

	Early Capability		Middle Capability		Later Capability	
	Year 1	Year 2	Year 3	Year 4	Year 5	Year 6
Resource Description	▪ Simple programmable robot (e.g. BeeBot) ▪ Problem solving / early logo apps (Kodable, Daisy the Dinosaur) ▪ Simple on screen control software (2Go, Textease Turtle) ▪ Logic problems / puzzles – (e.g…http://nrich.maths.org) **Weblinks** ▪ CS4FN (Computer Science for Fun) ▪ CS unplugged (Computer Science Unplugged)		▪ More complex floor robots (e.g. Probots, roamers) ▪ Logo software (e.g. Textease Turtle, Imagine Logo) ▪ Programming apps / software (e.g. Hopscotch, Scratch) ▪ Logic problems / maths puzzles ▪ Physical systems: models, buzzers, bulbs, control boxes (e.g. Flowgo boxes)and leads, building kits (e.g. Lego, K'nex) **Weblinks** ▪ CS4FN (Computer Science for Fun) ▪ CS unplugged (Computer Science Unplugged) ▪ http://www.code-it.co.uk/		▪ Physical systems: models, buzzers, bulbs, control boxes (e.g. Flowgo boxes) and leads, building kits (e.g. Lego, K'nex) ▪ Flowchart based control software (e.g. Flowol4, CoCo) ▪ Programming apps / software (e.g. Hopscotch, Scratch) **Weblinks** ▪ CS4FN (Computer Science for Fun) ▪ CS unplugged (Computer Science Unplugged) ▪ www.webmaker.org. (for teacher use and possible demonstration) ▪ http://www.code-it.co.uk/	
	NB – none of these products are endorsed by The ICT Service, nor is this list of resources in any way exhaustive. This simply reflects some of the resources we know are readily available in schools already or are free to use.					
Example Activities (Plugged / unplugged)	Role-play acting as 'human robots', predicting what will happen and finding and fixing errors (bugs) along the way. Implement algorithms as programs on Beebots to, for example, find their way to a given point on a map.	Create visual algorithms using flashcards for e.g. BeeBots. Challenge pupils to create increasingly complex programs such as enacting some of the Beebot app levels. Program on-screen simulations through online resources / early block languages (e.g. Daisy the Dinosaur app or 2Go).	Use an on-screen logo program to draw simple shapes such as a rocket ship or house. Find and correct errors (debug) to achieve each goal efficiently. Create simple animations using move, say and sound commands in Scratch e.g. launch a rocket ship using a drum roll and a countdown.	Draw simple 2D shapes using repeating instructions (e.g. draw a square by drawing a line, turning 90° and then repeating this 4 times). Replicate this on screen using logo software or a block programming app such as Hopscotch. Create a 'rolling ball' maze game in Scratch, thinking carefully about how the ball will move and what obstacles players will face.	Program on screen simulations such as traffic lights or fairground rides using software such as Flowol4. Begin to control models with bulbs, buzzers and motors and include switches to control the sequence of actions. Create a greetings card in Scratch using the 'broadcast message' command and a variable where users can add their own name.	Create more complex, 2 player 'bumper cars' game with consequences built in for crashing into each other or the 'barriers'. How many players can you accommodate? 'Pitch' your coding skills in an 'apprentice' style scenario, explaining how you've been efficient and why your game is better than anyone else's! Use constructions kits to build physical models incorporating bulbs, buzzers, motors and a selection of switches.

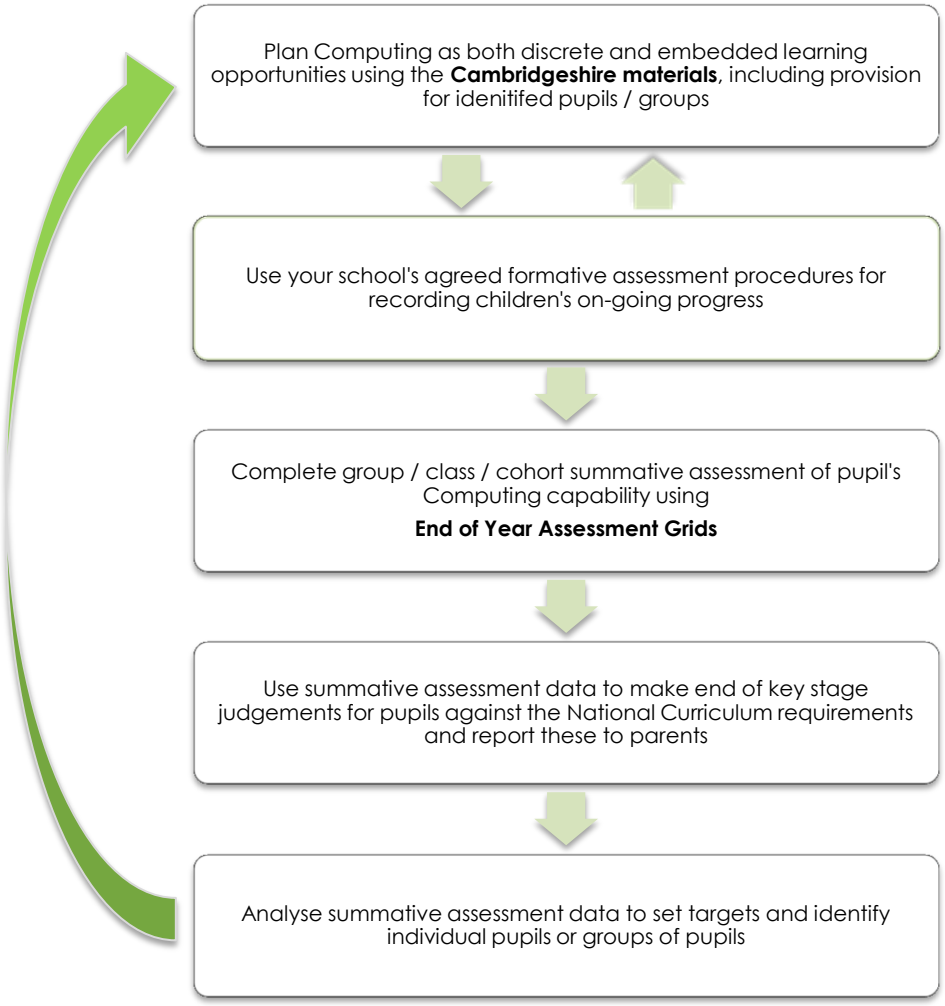
the ict service. education
info@theictservice.org.uk

DRAFT (September 2014)

Cambridgeshire Progression in Computing Capability

In September 2012, the DfE disapplied the 'Programmes of Study, associated attainment targets and statutory assessment arrangements for ICT'. Cambridgeshire suggests the following approach to assessing Computing capability and we will continue to update our guidance as further information is available nationally.

The Assessment Process:



The DfE document '**Primary assessment and accountability under the new national curriculum**' (July 2013) clearly states that '*schools will be able to introduce their own approaches to formative assessment*'. Whichever approach schools choose to adopt, appropriate, targeted questioning should continue to form an essential part of the assessment process in helping pupils to articulate their learning. The following sample questions and statements are designed to support teachers in using effective, open ended questions to collect evidence about what their pupils have learned.

the ict service.
education
info@theictservice.org.uk

DRAFT (September 2014)