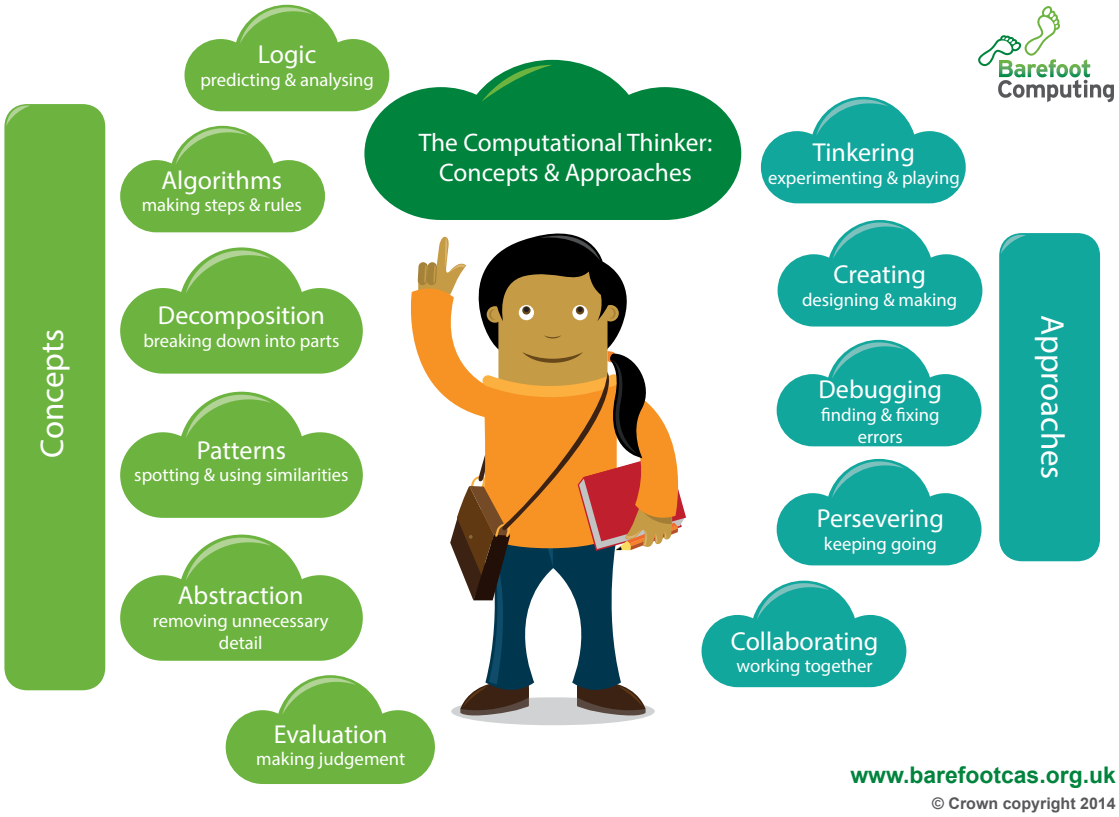
Computational Thinking

What is computational thinking?

> *A high quality computing education equips pupils to use computational thinking and creativity to understand and change the world.*
> (Computing National Curriculum)

Computational thinking allows us to develop skills and techniques to help us solve problems effectively, with or without the aid of a computer. Computational thinking is not 'thinking like a computer' – computers are not capable of thought. Rather, it is learning to think in ways which allow us, as humans, to solve problems more effectively and, when appropriate, use computers to help us do so.

Computational thinking involves 6 different concepts and 5 approaches to working:



The Barefoot Computing computational thinking concepts and approaches

Logic – predicting and analysing

Logic is the study of reasoning. The purpose of logic is to help us try and make sense of things: it helps us establish and check facts.

For example, KS1 pupils might predict the outcome of Bee-Bot programs. At KS2, pupils may use logical reasoning to explain what their algorithms or code does or to help them debug Scratch programs.

Algorithms – making steps and rules

An algorithm is a sequence of instructions, or set of rules, for performing a task.

For example, KS1 pupils might work out the steps a Bee-Bot needs to make to travel a route and note these down as a series of arrow drawings. KS2 pupils, when designing an animation, might label the a storyboard with numbers and notes to show the sequence of the animation action.

Decomposition – breaking down into parts

Decomposition is breaking a problem or system down into its parts. It sometimes involves breaking those parts down further. Decomposition helps us solve complex problems and manage large projects.

For example, if KS1 pupils are asked to program a Bee-Bot to navigate a long route, they might decompose the route into sections. KS2 pupils, when designing a quiz, might decompose the project into parts such as the questions, the artwork for the background and the score.

Patterns – spotting and using similarities

Using patterns means spotting similarities and common differences. By identifying patterns we can make predictions,create rules and solve more general problems. This is called generalisation.

For example, KS1 pupils might tinker with a Bee-Bot to find out what happens, as they do this they notice what the Bee-Bot does as they press the different command keys; they build up rules about how the programming language works. KS2 pupils, when designing a game, might compare a number of example games to work out the common features.

Abstraction – removing unnecessary detail

Abstraction is simplifying things. Abstraction is identifying what is important without worrying too much about the detail. Abstraction allows us to manage complexity. Abstracting leads to a simple view of the main idea of a thing. It's about seeing the wood for the trees.

For example, KS1 pupils might summarise the action for a simple animation of a joke as a storyboard. KS2 pupils, when creating a computer game might model how a ball bounces, but not the effect of air resistance

Evaluation – making judgement

Evaluation is concerned with making judgements, in an objective and systematic way whenever possible. Evaluation is something we do every day: we make judgements about what to do and what we think based on a range of factors

For example, KS1 pupils might think about how their work could be improved and KS2 pupils might consider the views of others to improve their work. Pupils can evaluate finished work and also make evaluative judgements as they work on a project.

Why is computational thinking important?

Computational thinking and the concepts behind it, form the basis for much of computer science. Computer scientists are interested in finding the most efficient way to solve problems. They want to find the best solution that solves a problem correctly in the fastest way and using the least amount of resources (time / space).

- Is this the most efficient way to solve the problem?
- Is this the fastest way?
- Does it require the least amount of resources?
- Does it solve the problem and give the right answer?
- Can it be used to solve other problems?

The pupils from Henley Primary School in the image below are doing a great deal of computational thinking naturally as they consider the best way to build a robot. Considering whether a context or a problem can be computable and how this can happen is an important element of computational thinking. In this example, this could be how they could program their robot to perform a variety of tasks.

Pupils from Henley Primary school developing their computational thinking skills

What does computational thinking look like the primary curriculum?

EYFS

There is lots of opportunity to encourage the building blocks of computational thinking. For example, with support, pupils can work collaboratively to build the highest tower, or to work out the best way to negotiate climbing equipment.

At this level pupils can, with support, understand algorithms, decomposition, and abstraction, for example working out how to 'get dressed for winter' involves decomposing the problem and sequencing instructions (algorithms).

They are also focussing only on the important elements, rather than the detail (abstraction).

Please see the individual concept and approaches pages for further examples of how to include the different elements of computational thinking.

KS1

Pupils can sequence simple algorithms using decomposition of simple problems, such as how to grow a plant from seed. They may label the parts of a flower (decomposition), and check with a partner to see if their work is correct (collaborative debugging and evaluation). Pupils can consider whether a problem is best solved with or without a computer.

For example, when finding out about a particular topic, say how methods of transport have changed over time, would it be best to look in a book, or look on the internet? How can they conduct the internet search in the most efficient way?

Similarly, if they wanted to take a picture of something so others could see it, which device, if you have a choice, would be best to use and why?

Please see the individual concept and approaches pages for further examples of how to include the different elements of computational thinking.

KS2

As pupils progress through key stage 2 they can demonstrate increasing levels of computational thinking as their cognitive ability develops; decomposing to an increasing number of levels, designing algorithms and implementing programs with increasing confidence.

They can spot patterns and abstract more readily, focussing on relevant detail only; for example in maths working out that if they complete a multi-step problem in a particular way, are more likely to reach the correct answer (or vice versa!).

Computational thinking approaches become more familiar, for example, pupils can debug a problem more effectively, whether that is finding and correcting grammatical errors in a piece of text or code, either independently or collaboratively.

A classroom culture in which collaboration and 'trying-things-out' (tinkering) is actively encouraged, while over-reliance on the teacher is discouraged can help to build pupils' confidence alongside their computational thinking.

Please see the individual concept and approaches pages for further examples of how to include the different elements of computational thinking.

Find out more about computational thinking

More information on computational thinking in primary
Kiki Computational Thinking materials
Google Education computational thinking micro-site
Miles Berry on computational thinking in primary schools