Anchoring Computational Thinking in Today's Curriculum

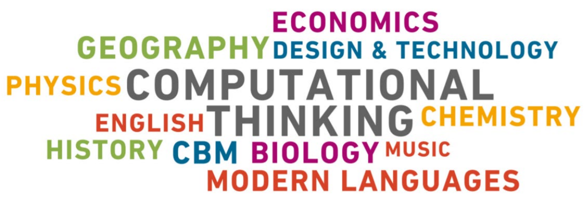
Conrad Wolfram, Wolfram Group

There is a lot of talk of Computational Thinking as a new imperative of education, so I wanted to address a few questions that keep coming up about it. What is it? Is it important? How does it relate to today's school subjects? Is Computer-Based Maths (CBM) a Computational Thinking curriculum?

Firstly, I've got to say, I really like the term.

To my mind, the overriding purpose of education is to enrich life (yours, your society's, not just in terms of 'riches' but in meaning)—and having different ways of thinking about how you look at ideas, challenges and opportunities seems crucial to achieving that.

Therefore, using a term of the form 'X Thinking' that cuts across boundaries but can support traditional school subjects (e.g. history, English, maths) and that emphasises an approach to thinking is important to improving education.



Now, we've had widespread use of the term 'Critical Thinking' for some time, but to me it has much less power of actuality than 'Computational Thinking'. Computation is a highly definitive set of methodologies—a system for getting answers from questions, and one rapidly gaining in power and applicability each year. There is no parallel, definitive 'critic' system; and even the related 'critiquing' is a rather vague skill bucket, not a systemic—and highly successful—roadmap. As a result, Critical Thinking often becomes more of an aspiration of student capability, not a definable, definite, life-enriching set of problem-solving abilities.

To be specific, I'd argue that Computational Thinking is a mode of thinking about life in which you creatively and cleverly apply a four-step problem-solving process to ideas, challenges and opportunities you encounter, to make progress with them.

Here's how it works.

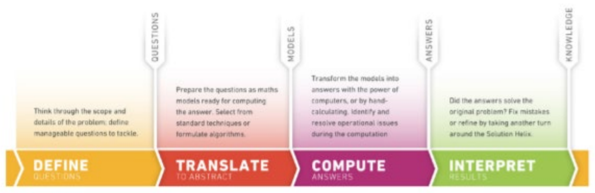

QUESTIONS
MODELS
ANSWERS
KNOWLEDGE
Think through the scope and details of the problem: define manageable questions to tackle.
Prepare the questions as maths models ready for computing the answer. Select from standard techniques or formulate algorithms.
Transform the models into answers with the power of computers, or by hand-calculating. Identify and resolve operational issues during the computation
Did the answers solve the original problem? Fix mistakes or refine by taking another turn around the Solution Helix.
DEFINE
QUESTIONS
TRANSLATE
TO ABSTRACT
COMPUTE
ANSWERS
INTERPRET
RESULTS

Figure 1: CBM problem solving process

You start by **defining the question** that you really want to address—a step shared with most definitions of Critical Thinking.

But computational thinking follows this with a crucial transitional step 2 in which you take these questions and **translate into abstract** computational language—be that code, diagrams or algorithms. This has several purposes. It means that hundreds of years' worth of figured-out concepts and tools can be brought to bear on the question (usually by computer), because you've turned the question into a form ready for this high fidelity machinery to do its work. Another purpose of step 2 is to force a more precise definition of the question. In many cases this abstraction step is the one that demands the highest conceptual understanding, creativity, experience and insight.

After abstraction comes step 3, the **computation** itself—where the question is transformed into an abstract answer, usually by computer.

In step 4 we take this abstract answer and **interpret** the results, re-contextualising them in the scope of our original questions and sceptically verifying them.

The process rarely stops at that point, because it can be applied over and over again with output informing the next input until you deem the answers sufficiently good. This might take just a minute for a simple estimation, or a whole lifetime for a scientific discovery.

Modern technology has dramatically shifted the effective process because you don't get stuck on your helix roadway at step 3, so you may as well zoom up more turns of the track faster.

I think it's helpful to represent this iteration as ascending a helix made up of a roadway of the four steps, repeating in sequence until you can declare success.

While I've emphasised the process end of Computational Thinking, the power of its application comes from (what are today!) very human qualities of creativity and conceptual understanding. The magic is in optimising how process, computer and human can be put together to solve increasingly tough problems.

The Computational Thinking process

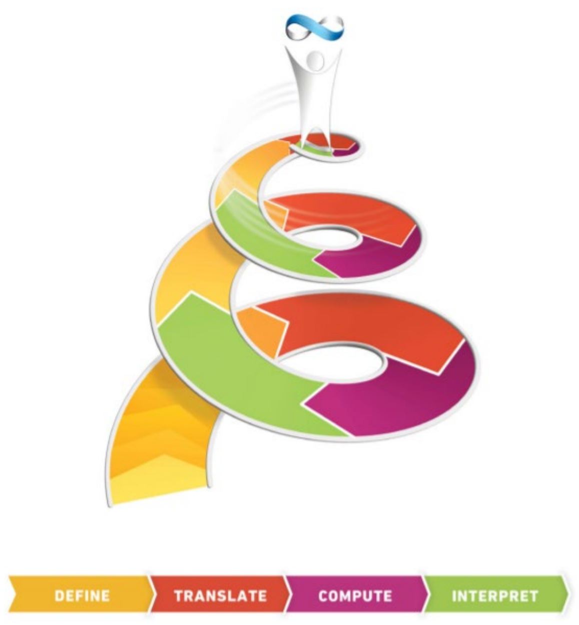


Figure 2: the Computational Thinking process

Is this process of Computational Thinking that I describe connected with maths—are they even one and the same? And what about coding?

There is very heavy overlap with the Computer-Based Maths approach, and much less with today's traditional maths education; coding is an important element, particularly as the main way in which you manifest abstraction.

Real-world maths—defining it and its applications broadly, as I do—absolutely relies on Computational Thinking. There are also specific areas of knowledge that maths is considered to contain (e.g. particular concepts and algorithms), which are often important in applying Computational Thinking to different areas of life. Maths is a domain of factual knowledge as well as the skills knowledge of how to process it.

Even in the real world this broad definition of maths may be alien to engineers or scientists, who would consider what I'm describing simply as part of engineering or science respectively.

There's another key difference, too, between a traditional maths way of thinking about a problem and a modern Computational Thinking approach, and it has to do with the cost–benefit analysis between the four steps of the helix.

Before modern computers, step 3—computation—was very expensive because it had to be done manually. Therefore, in real life you'd try very hard to minimise the amount of computation at the expense of much more upfront deliberation in steps 1 (defining the question) and 2 (abstracting). It was a very deliberate process. Now you might have a much more scientific or experimental approach, with a looser initial question for step 1 ('Can I find something interesting in this data?'), and an abstraction in step 2 leading to a multiplicity of computations ('Let me try plotting correlation of all the pairs of data')—because computation (step 3) is so cheap and effective you can try it lots and not worry if there's wastage at that step. Modern technology has dramatically shifted the effective process because you don't get stuck on your helix roadway at step 3, so you may as well zoom up more turns of the track faster.

DEFINE
TRANSLATE
COMPUTE
INTERPRET

Figure 3: Computational Thinking process bar

A useful analogy is the change that digital photography has brought. Taking photos on film was relatively costly (though cheap compared with the chemical-coated glass plates it replaced). You didn't want to waste film, so you'd be more meticulous at setting the shot before you took it. Now you may as well take the photo; it's cheap. That doesn't mean you shouldn't be careful to set up (abstract it) to get good results, but it does mean the cost of misfires, wrong light exposure and so forth is less. It also opens up new fields of ad-hoc photography to a far wider range of people. Both meticulous and ad-hoc modes can be useful; the latter has added a whole new toolset, though it doesn't always replace the original approach.

Back to maths. What's sadly all too clear is that today's mainstream educational subject in this space of 'maths' isn't meeting the real-world need of Computational Thinking that could be addressed by Computer-Based Maths. Its focus on teaching students how to do step 3 manually might have made sense when that was the sticking point in applying maths in life: because if you couldn't do the calculating, you couldn't use maths (or Computational Thinking). Conversely, providing experience primarily in a very deliberate, meticulous, uncontextualised, pre-computer application of the computational process—rather than in a faster-paced, computer-based, experimental, scientific-style application to real problems—cannot continue to be maths' chief purpose if the subject is to remain mainstream. Instead, its primary purpose ought to be Computational Thinking—as it is in our CBM manifestation.

Like real-world maths, coding relies on Computational Thinking but it isn't the same subject or (by most definitions) anything like a complete route to it. You need Computational Thinking for figuring out how to extract problems to code and get the computer to do what you want, but coding is the art of instructing a computer what to do; it's the expertise you need in order to be the sophisticated manager of your computing technology, and that includes speaking a sensible coding language, or several, to your computer.

What of other school subjects? Computational Thinking should be applicable to a very wide range. After all, it's a way of thinking—not the only way of thinking, but an important perspective across life. Whether it's design ('How can I design a streamlined cycle helmet?'), or history ('What was the key message each US president's inaugural address delivered?'), or music ('How did Bach's use of motifs change over his career?'), every subject should envelop a Computational Thinking approach.

The Computational Thinking approach needs knowledge of what's possible, experience of how you can apply it, and knowledge of today's machinery to perform it.

An important practical question is whether this wider application can happen without there being a core educational subject whose essence is Computational Thinking? I don't think so. Not at school levels, anyway. That's because the Computational Thinking approach needs knowledge of what's possible, experience of how you can apply it, and knowledge of today's machinery to perform it. You need to know which concepts and tools there are to translate and abstract to in step 2. I don't think you can learn this only as part of other subjects; there needs to be an anchor where these modern-day basics (learnt in a contextualised way) can be fostered.

Politically, there are two primary ways to achieve this: introduce a new core subject, or transform an existing one. Either is a major undertaking. Maths and coding are the only existing school subject contenders for the transformational route. Maths of course is ubiquitous, well-resourced and occupies a big part of the curriculum—but today's subject largely misses the mark. Coding is the new kid on the block, too narrow, not fully established; it has far less time or money but has a zeal to go to new places.

How does CBM relate? For the very short term, simply as the start of today's best structured program for engendering Computational Thinking—one that's principally around maths but is applied to problems and projects from all subjects.

Ultimately our aim is to build the anchor Computational Thinking school subject as we explicitly broaden CBM beyond being based in maths (and, just as importantly, beyond the perception of it being based only in maths). Look out for modules of CBM geography and CBM history!

Make no mistake: whatever the politics or how it's labelled, whoever wins or loses—someday a core, ubiquitous school subject in the space I'm describing will emerge. The first countries, regions and schools that manage this new core and its cross-curricular application will win big time.

This article was prepared for *Horizon: Thought Leadership*, a publication of the Bastow Institute of Educational Leadership, Department of Education and Training, Melbourne, Victoria, Australia.

This was first published on 4 October 2016 as a blog post at www.conradwolfram.com/home/anchoring-computational-thinking-in-todays-curriculum and is reproduced here with the permission of the author.

Conrad Wolfram, physicist, mathematician and technologist, is Strategic Director and European Co-Founder/CEO of the Wolfram group of companies. Described as the ‘Computation Company’ and uniquely operating at the intersection of computation, data and knowledge, the Wolfram Group is driving innovation across data science, modelling and maths through technology and solutions. Conrad is also widely known for his thought leadership in reforming education using modern technology. He founded computerbasedmath.org to fundamentally rethink and rebuild the mainstream maths education curriculum, introduce computational thinking and combine with coding now computers can be assumed. The resulting major change is now a worldwide force in re-engineering STEM, with early projects in Estonia, Sweden and Africa.